Italian Institute of Technology
University of Genoa

Doctoral School on Humanoid Technologies
XXI cycle

'Cognitive Robots'
From Affordance to Action & back

Thesis submitted for the degree of
"Doctor of Philosophy"
by

Vishwanathan Mohan

Supervisors: Prof. Pietro G. Morasso
Prof. Giorgio.Metta
Prof.Vittorio. Sanguineti

Genova, Italy
A.A 2005/2008

The research described in this book has been supported by the grants from the Italian Institute of Technology, a foundation established jointly by the Italian Ministry of Education, Universities and Research and the Ministry of Economy and Finance. The research was also supported by the European Union through the FP6 project GNOSYS.

Thirst drove me down to the water, where I drank the reflections of the moon.

Jalaladdin Muhammad 'Rumi'

12th century Sufi Mystic, Poet

To
all the accidental forces (physical and psychological)
agents (natural/artificial)
that have shaped my ‘doing’
and hence shaped my ‘being’,
and helped me grow
‘from potential to actual’

Acknowledgements

Passion for science is often infectious! and during this journey I have been extremely fortunate that my paths crossed several such people who have directly and indirectly contributed towards the development of the work presented in this manuscript. Particularly, I would like to thank Prof.Pietro Morasso and Prof.Giorgio Metta for several things, most importantly, having the passion to play around with important problems in science, often with a smile that speaks much more than what few blobs of ink in this paper can ever express. I thank Prof.Giulio Sandini for providing me the opportunity to play with expensive toys (robots) and also for all the freedom and encouragement he gave me during the last three years. Thanks to Prof. David Vernon for that brilliant course on cognitive systems he conducted! Thanks to Ingrid and Anastasia for often going out of the way to help me with the beurocratic issues, an issue I am not very 'Cognitive' about! I also extend my gratitude to Prof.Vittorio Sanguineti and all the colleagues of the neurolab for creating a simulating environment to work in.

I am profoundly indebted to Stathis Kasderidis for all the 'space, time, energy and tireless efforts' he put to make GNOSYS functional. We have often seen failures together many times during the creation of GNOSYS. Our several discussions in 'Heaven' has greatly influenced the development of this work. (Heaven is an exotic cafeteria in the island of Crete). I also thank Ioanna for taking care of me whenever I was in Greece. This work is a result of past four years of collaborative research under the framework of the EU funded project GNOSYS. I thank our inspirational leader Prof.John.G.Taylor for most importantly asking the right and often difficult questions, and sometimes permitting me to bite much more than I could chew! I also take this opportunity to express my gratitude to the other collaborating teams of GNOSYS, Prof.Hans Peter Mallot, WolfGang, Christo for creating the visual perception system (the main inputs to my reasoning architecture), Vassilis Spais and Nikos Chatzinikoalu for assembling the robot and allowing us to create an additional playground for the robot at ZENON, Harris Baltzakis for creating the excellent ORCA simulator that I used for some of the simulations related to spatial navigation. I would like to thank Prof.V.Srinivas Chakravarthy of the Indian Institute of Technology Madras, for introducing me to neural networks, associative memories and thermodynamics of computation, a field that still deeply interests me, even though I could not find the much needed time to wonder about it in the past few years. I would to thank Luca, Boris and Mehmet for some really interesting discussions we have had on non linear dynamics and chaos. Thanks to my very old friends Indrajit, Ravi, Sushant, Rajesh, Krishnan, Rahul, Winnieza and Anisa for the support they extended all these years.

I would like to thank the several authors whose works have often inspired and influenced my thinking mainly, the Indian Mystic OSHO Rajneesh, Douglas Hofstadter, VS Ramachandran, Gerald Edelman, Peter Atkins, Richard Dawkins, Antonio Damasio, Richard Feynman, Murray Gell Mann, Walter Freeman, Helen Fisher, Ekhart Tolle and George Lakoff.

Finally, I express my deep gratitude towards the members of my family for the support and love they have given me all along these years.

Abstract

Affordances are the seeds of action. Being able to identify and exploit them opportunistically in the 'context' of an otherwise unrealizable goal is a sign of cognition. Being able to do this in the mind by performing virtual actions, further allows an agent to mentally evaluate 'what additional affordances' it can create in its world as a consequence of its real/simulated actions, hence most importantly enabling it to reason about how the world must 'change' such that it becomes a little bit more conducive towards realization of its internal goals. What is the nature of the computational substrate that could drive embodied robotic agents to exhibit such levels of cognitive control over their perceptions actions and imaginations? The problem is difficult and the solution often elusive. However, there are several pressures to provide an answer, both from the intrinsic viewpoint of better understanding ourselves, to creating robots that can flexibly deal with our needs and in the environments we inhabit. In this thesis, we propose an 'internal models' based computational architecture for reasoning and action generation with an aim to empower robotic embodiments with some preliminary ability to virtually manipulate neural activity in their mental space in order to exhibit flexible goal directed behavior in their physical space. Both the developmental process (learning) and the seamless flexibility of the proposed architecture is expressed through the life of a moderately complex robot 'acting, learning and performing' in a moderately complex playground, attempting to use its 'perceptions actions and imaginations' to intelligently and resourcefully cater 'rewarding' user goals. A passive motion paradigm (Mussa Ivaldi et al, 1988) based forward inverse model for mental simulation / real execution of goal directed arm (and arm+tool) movements, a spatial mental map, an internal model for pushing objects and an abstract sensorimotor space (reasoning system) are progressively created in the behavioral repertoire of the robot. From a global view point, the complete computational architecture can be visualized as a loosely coupled network of dynamical systems, with the power of sensorimotor exploration, self organization, field computing and value dependent learning exploited at all levels of hierarchy (and in all the internal models). Additionally, a high level of abstraction (to manage/coordinate complexity emerging at different scales, dynamical systems, changing world), approximate self similarity in structure, learning and dynamics of the different internal models acquired by the robot, heterogeneous optimality (learning when to optimize what), circularity (closure between perception and action at all scales, with inconsistencies at one level becoming an affordance to 'act, reason or explore' at some other level) and recursivity (self referential nature) form the cornerstones of the proposed architecture. A resulting side effect, as demonstrated by the results presented in this manuscript, is the scalability, portability, open-endedness and effortless up gradation of the computational architecture while dealing with more complex 'bodies, worlds and goals'.

Contents

Introduction

> Like the entomologist in search of brightly colored butterflies, my attention hunted, in the garden of grey matter, the mysterious butterflies of the soul.
>
> SANTIAGO RAMÒN Y CAJAL

The world we inhabit is an amalgamation of structure and chaos. There are regularities that could be exploited. Species biological or artificial, which do this best have the greatest chances of survival. We may not have the power of an ox or the mobility of an antelope but still our species surpasses all the rest in our flair by inventing new ways to think, new ways to functionally couple our bodies with the structure afforded by our worlds. Simply stating, it is this ability to 'explore, identify, internalize and exploit' the possibilities afforded by the structure in one's immediate environment to counteract limitations 'of perceptions, actions and movements' imposed by one's embodied physical structure, and to do this in accordance with one's 'internal goals', that forms the hallmark of any kind of cognitive behavior. In addition, natural/artificial systems that are capable of utilizing 'thoughts' at the service of their 'actions' are gifted with the profound opportunity to mentally manipulate the causal structure of their physical interactions with the environment. Complex bodies can in this way decouple behavior from direct control of the environment and react to situations that 'do not really exist' but 'could exist' as a result of their actions on the world. However, the computational basis of such cognitive processes have still remained elusive.

This is a difficult problem, but there are many pressures to provide a solution – from the intrinsic viewpoint of better understanding ourselves to creating artificial agents, robots, smart devices and machines that can reason and deal autonomously with our needs and with the peculiarities of the environments we inhabit and construct. This has led researchers towards several deep questions regarding the nature of the computational substrate that could drive an artificial agent to exhibit flexible, purposeful and adaptive behavior in complex, novel and sometimes hostile environments. How do goals, constraints and choices 'at multiple scales' meet dynamically to give rise to the seemingly infinite fabric of reason and action? Is there an internal world model (of situations, actions, forces, causality, abstract concepts)? If yes, 'How' and 'What' is modeled, represented and connected? How are they

invoked? What are the planning mechanisms? How are multiple internal models coordinated to generate (real/mental) sequences of behaviors 'at appropriate times' so as to realize valued goals? How should a robot respond to novelty and how can a robot exhibit novelty ? What kind of search spaces (physical and mental) are involved and how are they constrained? This thesis is in many ways an exploration into some of these questions expressed through the life of a moderately complex robot playing around in a moderately complex playground (which implicitly hosts artificially reconstructed scenarios inspired from animal cognition), trying to use its perceptions, actions and imaginations 'flexibly and resourcefully' so as to cater 'rewarding' user goals.

So why a robot ? At least for the human species, tinkering around with possibilities afforded by nature has undoubtedly led to progress. Form stones, anvils and hand axes, we have moved on to create civilizations, we have moved on to create art, and we have moved on to create a wide gamut of sophisticated tools, machines and robots that aid us in our daily lives by performing amazingly sophisticated tasks (some of which are definitely beyond our physical capabilities). Several researchers (Metta, 2000, Lungarella et al, 2003) have realized the additional affordances provided by the robots other than their regular applications in industry, household, hospitals defense, space technology etc. This additional affordance comes from the fact that other than its usual use in a variety of straightforward applications, a robot can also be used as a tool to better understand our own selves, to understand how interactions between body and world shapes the mind, shapes reason and shapes action. This stems from the fact that unlike the range of direct problems common in conventional physics that involve computing effects of forces on objects, brains have to deal exactly with the inverse problems of reasoning, choosing and determining the motor commands that would permit the intended, goal directed mechanical interaction with the world. Strikingly, many of the inverse problems faced by the brain to control movements and drive intelligent behavior are indeed similar to the ones roboticians must solve to make their robots move intelligently and act flexibly in the world. Hence, while the field of neuroscience benefits from the theories of construction and control of manmade bodies, roboticians on the other hand have the profound parallel opportunity to learn about the structural and functional organization of the central nervous system.

In this introductory chapter, we will concisely present an executive summary of some of the important problems investigated in this manuscript. Writing a lengthy literature survey is not the goal of this chapter and we will restrict ourselves to a general synopsis summarizing some recent progress in the area, the scientific and practical interests related to the work presented in this thesis. In fact every chapter has its own related literature that is concisely presented in a local fashion in order to facilitate a context dependent reading/understanding of the central issues. To make things bit more interesting, we will begin with some cartoons of the main results presented in this thesis (that we will encounter in the following chapters 2-6), and try to build the primary 'operational' objectives (the 'What' part) around those

cartoons. The rest of the thesis is of course about the 'Why' and the 'How' parts of the problem.

1.1 Central Themes through Pictures

Figure 1 shows a robotic embodiment 'G' (GNOSYS) 'thinking about' realizing simple user goals like grasping different objects (blue stick in panel A, and red ball in panels B-D) while perceiving four different environmental scenarios in a constantly changing world 'W' and powered with a reasoning-action generation architecture 'R' that drives its physical/mental behavior 'P' (plan descriptor).

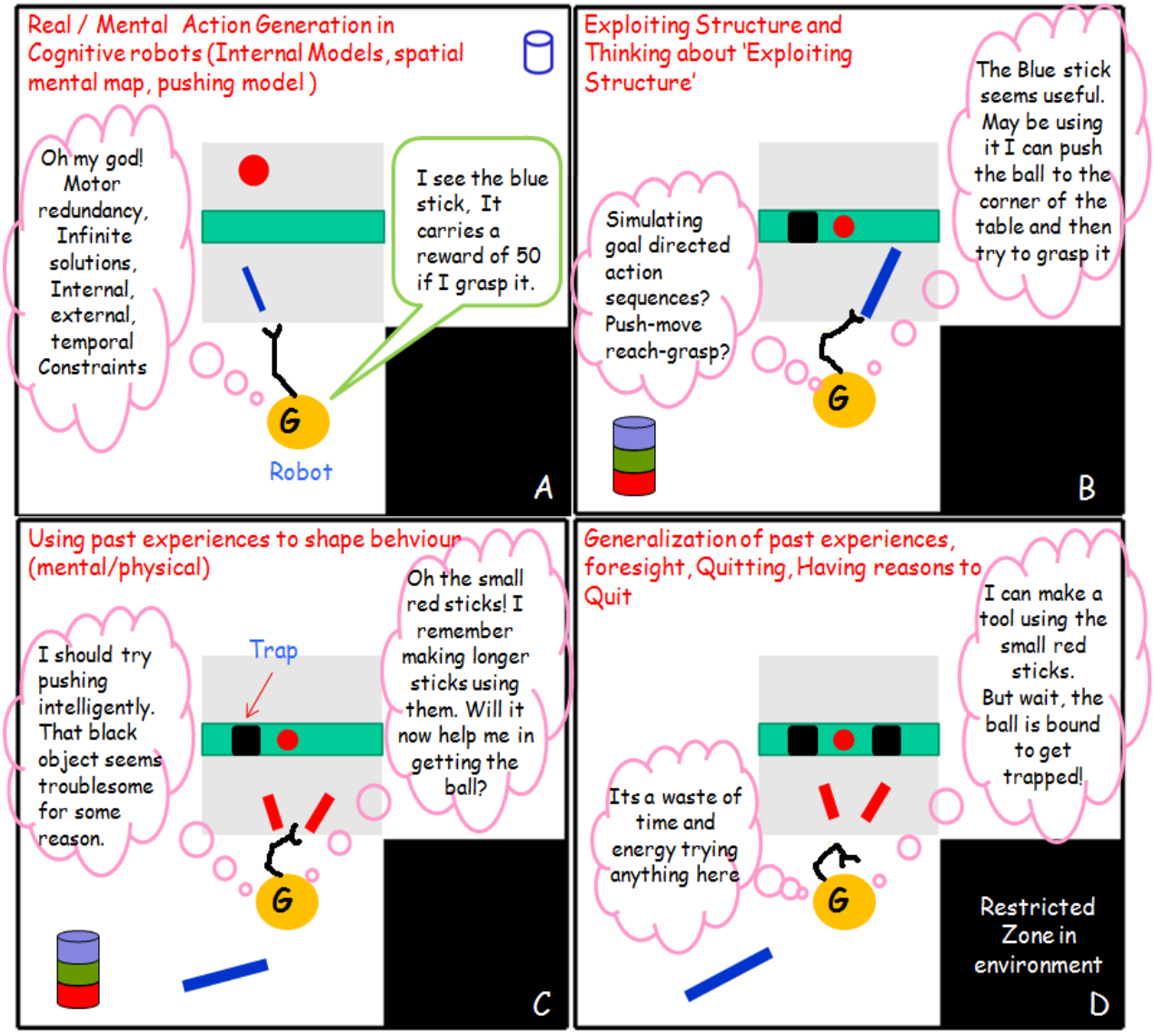


Figure 1. Central themes through pictures

Cognition is a concept with a spectrum of interpretations, going from those involving a range of complex sensory, motor, memory, attention and reasoning related processes to those having conscious awareness of the processing inside the cognitive system. It must be quite clear from figure 1 that the work presented in this thesis relates to the development of computational architectures driving higher level cognitive functions in embodied robotic platforms that 'act, learn and perform' in moderately complex, changing environments. Of course the sequence of panels shown in figure 1 (A-D) was a result of the natural evolution of the work that we began starting with a simplistic reaching scenario using a couple of goal objects. The environmental complexity was gradually enhanced with an aim to make things more interesting, at the same time attacking extremely challenging computational problems with a simplified collection objects like few sticks, cylinders and balls (and the robot). As a starting point we defined the following set of primary 'operational' objectives:

- P1. The development of the reasoning-action generation system 'R' that drives the physical/mental behaviour of the GNOSYS robot. Additionally, on the course of the journey, the aim was also to develop and better understand the fundamental computational principles necessary to drive *any* artificial agent/robot to exhibit some level of resourcefulness, purposefulness, flexibility and adaptability in its 'reasons and actions' while inhabiting unstructured worlds like 'W'.
- P2. The goal behind development of the proposed architecture was not just to make the GNOSYS robot reason and act intelligently in its little world 'W' but also to implement the architecture in ways such that it can

  a) scale up with the complexity of newer worlds and goals;
  b) allow portability of the architecture on more complex robotic platforms;
  c) allow quick upgrading/relearning of the existing internal models when ported to other robots;
  d) allow learning of new internal models (primitives) without disrupting already existing interactions;
  e) allow hassle-free interfaces with computational models (developed by other groups) that support the reasoning architecture (mainly visual perception).

  The aim was also to focus on creating an architecture that is domain-agnostic and implemented with a relatively high level of abstraction.

These operational objectives can be further decomposed into a set of 'specific' sub objectives

- S1. Progressive construction of a moderately complex playground that both enables the robot to gain rich sensorimotor experience and facilitates the training/validation of the reasoning-action generation architecture in a diverse set of ecologically

realistic scenarios. Snapshots of four example environments with increasing levels of complexity (A-D) are presented in figure 1. *(Rich sensorimotor worlds for robots).*

- S2. Development of the forward/inverse model for real/mental action generation in the GNOSYS robot. As a consequence of the development of the internal model for coordinating arm/body movements, the aim was to be in a situation to attempt the scenario shown in panel A of figure 1, taking into account a range of task specific constraints 'at runtime'. 'I *see* the blue stick' implies that a fair amount of visual complexity is also involved, in addition to dealing with problems like motor redundancy in different motor spaces and accounting for a diverse set of internal, external, task-specific and temporal constraints. *(Forward/Inverse models).*
- S3. Learning an internal mental map of the GNOSYS playground (to deal with spatial goals and for the joint coordination the forward inverse models developed in S2 with space related planning/ anticipation). An additional objective was to create an internal model for pushing objects (i.e. being able to both anticipate its sensory and motor consequences and to execute goal directed pushing) taking into account traps placed at different locations along the trapping groove. Traps (black objects in the figure) were introduced suddenly at the later stages of development as a 'surprise' element for the robot. The main aim at this stage was to empower the robot with a primitive (but non trivial) ability to perform goal-directed mental simulations of sequences of actions. An example scenario is illustrated in panel B that involves a simulation of a sequence of actions in order to grasp the otherwise unreachable red ball. *(Mental maps, causality, value fields).*
- S4. Development of an abstract reasoning system, with a special focus on the crucial problems of organizing the massive amount of information arising in a changing world (mental/real). The complexity of realizing a task like 'Grasp a Red Ball' in changing environments like those shown in figure 1 (mainly panels C and D) results from the fact that before grasping the red ball itself with end-effector, there may be several intermediate sequences of real/virtual 'Reaching', 'Grasping', 'Pushing', and 'Moving' actions directed at 'potentially useful' environmental objects, information regarding which is not specified by the root goal itself (which was just 'Grasp the red ball'). So before realizing the root goal, the robot has to 'track down' and 'realize' a set of useful sub goals that 'transform' the world in ways that would then make the successful execution of the root goal possible. *(Past experiences, reasoning, goal space).* The scenario shown in Panel D is the ultimate case of exploiting the power of mental simulation i.e. to Quit without doing anything.
- S5. Demonstration of the effectiveness of the architecture in a physical instantiation seeking to realize user goals in diverse environmental scenarios (inspired by experiments related to physical cognition conducted in animals). A related objective

was to implement either parts or the complete computational architecture on other robotic embodiments (for example, the humanoid 'iCub' and the SCORBOT platform). This was addressed during the later stages of this thesis, mainly to demonstrate the portability, scalability and flexibility of the architecture. *(Modularity, abstraction, scalability).*

1.2. Practical and Scientific Interests : 'A General Synopsis'

The motivation behind the various objectives formulated in the previous section was the understanding that one needs to establish important progress in the development of computational frameworks that enable robotic embodiments to handle user goals (related to daily activities of humans) in changing environments. Further, it should enable them to exhibit a high level of 'mobile intelligence' in their behaviors while manipulating objects, moving around and interacting with (or learning from) their environments. These objectives are also clearly reflected in the recently released draft of the European commission (WP9) identifying important challenges for research in the field of cognitive robotics under the seventh framework programme (2007-2013) that quotes:-

"Challenge 2 aims to extend systems engineering to the design of systems that can carry out useful tasks (eg, *manipulation* and grasping, *exploration* and *navigation*, *monitoring* and *control*, *situation assessment*, communication and interaction), autonomously or in cooperation with people, in circumstances that were not planned for explicitly at design time. Specifically, such systems should be:

- more *robust*: performance should not degrade when they are presented with unexpected data;
- more *adaptive*: performance should be open (within reasonable constraints) to changing service requirements, without the need for extensive human intervention;
- more effective: performance should improve because they can *predict* or *anticipate* what might happen at some point in the future, near or far;
- more natural: performance should be tolerant to the ambiguity and uncertainty that is a consequence of dealing with humans, and performance should improve with time.

System capabilities in dimensions such as *deliberation* and learning, and innovation and *creativity*, would appear to be necessary to meet this aim.

This clearly calls for design that shares some characteristics with the higher-level cognitive processes of the brain."

It is fair to say that in spite of extensive research in multiple fields, scattered across multiple scientific disciplines, the present artificial agents still lack much of the resourcefulness, purposefulness, flexibility and adaptability that humans so effortlessly exhibit. Fundamental problems for generating flexible motor behavior under the presence of multiple task specific constraints (without predefined optimizations and cost functions) are still open as are the following ones, to mention a few: autonomous/progressive learning of sensorimotor

dependencies through exploration (bootstrapping); development and use of internal models for goal-directed visual and motor imagery (foresight, prediction); identification of affordances in the environments through intervention and observation (affordance as seeds of action); reasoning about actions in ecologically realistic scenarios (in contrast to puzzle solving, question answer sessions on induction and deduction in software); representation of physical causality; robust behavior in open ended environments; acquisition of new motor skills; integration of new conceptual knowledge.

Cognitive agent architectures are found in the current literature, ranging from purely reactive ones implementing the cycle of perception and action in a simplistic hardwired way to more advanced models of perception, state estimation and action generation (Brooks, 1986; Georgeff, 1999, 1987; Toussaint, 2006; Shanahan, 2005, 2006), architectures for analogy making (Hopfstader, 1984; French, 1995; Kokinov and Petrov 2001), causal learning (Pearl, 1998; Geffner, 1992), probabilistic/ statistical inference (Yuille et al, 2006, Pearl 1988) and brain based devices (DARWIN Series). Even though symbols and symbol manipulation have been the main stay of cognitive sciences (Newell and Simon, 1976) ever since the days of its early incarnations as AI, the disembodied nature of traditional symbolic systems, the need to presuppose explicit representations, symbol grounding and all other associated problems discussed in Sun (2000) have been troubling many cognitive scientists (Varela and Maturana, 1974; Churchland ,1986).

This led to the realization of the need for *experience* to precede *representation*, in other words the emergence of representational content as a consequence of sensory-motor interactions of the agent with its environment, a view that can be traced back to many different contributions spanning the previous decades, e.g. Wiener's Cybernetics (1948), Gibson's ecological psychology (1966), Maturana and Varela's autopoesis (1974), Beer's neuroethology (1990), Clark's situatedness (1997). In this view, adaptive behaviour can best be understood within the context of the (biomechanics of the) body, the (structure of the organism's) environment, and the continuous exchange of signals/energy between the nervous system, the body and the environment. Hence the appropriate question to ask is not what the neural basis of adaptive behaviour is, but what the contributions of all components of the coupled system to adaptive behaviour and their mutual interactions are (Morasso 2006c).

In other words, the ability to autonomously explore, identify, internalize and exploit possibilities afforded by the structure in one's immediate environment is critical for an artificial agent to exercise intelligent behaviour in a messy world of objects, choices, relationships. Intelligent agents during the course of their lifetimes gradually master this ability of coherently integrating the information from the bottom (sensory, perceptual, conceptual) with the drives from the top (user goals, self goals, reward expectancy), thereby initiating actions that are maximally rewarding. A major part of this process of transformation takes place in the mental space (Holland and Goodman, 2003) where in the

agent, with the help of an acquired internal model, executes virtual actions and simulates the usefulness of their consequences towards achieving the active goal. Hence, unlike a purely reactive system where the motor output is exclusively controlled by the actual sensory input, the idea that a cognitive system must be capable of mentally simulating action sequences aimed at achieving a goal has been gaining prominence in literature. This also resonates very well with emerging biological evidence in support of the simulation hypothesis towards generation of cognitive behaviour, mainly *simulation of action*: we are able to activate motor structures of the brain in a way that resembles activity during a normal action but does not cause any overt movement (Metzinger and Gallese, 2003; Grush 2004); *simulation of perception*: imagining perceiving something is actually similar to the perceiving it in reality, only difference being that, the perceptual activity is generated by the brain itself rather than by external stimuli (Grush, 1995); *anticipation*: there exist associative mechanisms that enable both behavioural and perceptual activity to elicit other perceptual activity in the sensory areas of the brain. Most important, a simulated action can elicit perceptual activity that resembles the activity that would have occurred if the action had actually been performed (Hesslow, 2002).

Computationally this implies the need to have two different kinds of loops in the agent architecture, firstly a situation-action-consequence loop or forward model that allows contemplated decision making (without actual execution of action) and secondly a Situation-Goal-Action loop to solve the inverse problem of finding action sets which map the transformation from initial condition to active goal. That such forward models of the motor system occur in the brain has been demonstrated by numerous authors. For example Shadmehr (1999) has shown how adaptation to novel force fields by humans is only explicable in terms of both an inverse controller and a learnable forward model. More recent work has proposed methods by which such forward models can be used in planning (where actual motor action is inhibited during the running of the forward model) or in developing a model of the actions of another person (Oztop, Wolpert and Kawato, 2004). Engineering Control frameworks of attention, using modules of control theory (Taylor, 2000) extended so as to be implemented using neural networks, have been extensively applied to modeling motor control in the brain (Morasso, 1981; Wolpert, Ghahramani & Jordan 1994, Morasso & Sanguineti 1997; Gribble et al 1998; Desmurget & Grafton, 2000; Miall & Wolpert, 1996; Kawato, 1999; Wolpert & Kawato, 1998; Imamizu, 2000), with considerable explanatory success. Such planning has been analysed in these and numerous other publications for motor control and actions but not for more general thinking, especially including reasoning. Nor has the increasingly extensive literature on imagining motor actions been appealed to: it is important to incorporate how motor actions are imagined as taking place on imagined objects, so as to 'reason' what objects and actions are optimally rewarding. Others have also emphasized the need to combine working memory modules for imagining future events with forward models, for example the process termed 'prospection'

in (Emery & Clayton, 2004). Guided by the experimental results from functional imaging and neuropsychology, computational architectures have recently begun to emerge in the literature for open-ended, goal-directed reasoning in artificial agents, most importantly incorporating the creation and use of internal models and motor imagery. A variety of computational architectures incorporating these ideas have been proposed recently, for example an architecture that combines internal simulation with a global workspace (Shanahan 2005), IAM (Internal Agent Model) theory of consciousness Holland (2003), learning a world model using interacting self-organizing maps (Toussiant, 2006, 2004), learning motor sequences using recurrent neural networks with parametric bias (Tani et al, 2007).

The idea of using internal models to aid generation of intelligent behavior also resonates very well with compelling evidence from several neuropsychological, electrophysiological and functional imaging studies, which suggest that much of the same neural substrates underlying modality perception are also used in imagery; and imagery, in many ways, can 'stand in' for (re-present, if you will) a perceptual stimulus or situation (Zattore et al, 2007; Berhmann ,2000; Fuster 2003). Studies show that imagining a visual stimulus or performing a task that requires visualization is accompanied by increased activity in the primary visual cortex (Kosslyn et al, 2006; Klein and Le Bihan, 2000). The same seems to be true for specialized secondary visual areas like fusiform gyrus, an area in the occipito-temporal cortex which is activated both when we see faces (Op de Beeck and Kanwisher 2008) and also when we imagine them (O'Craven et al, 2000). Lesions that include this area impair both face recognition (Damasio et al, 1990) and the ability to imagine faces. Brain imaging studies also illustrate heavy engagement of the motor system in mental imagery i.e. we are able to activate motor structures of the brain in a ways that resembles activity during a normal action but does not cause any overt movement (Parsons et al, 2005; Rizzolati et al 2001, Grush, 2004). EEG recordings on subjects performing mental rotation tasks have revealed activation of premotor and parietal cortical areas, indicating that they may be performing covert mental simulation of actions by engaging the same motor cortical areas that are used for real action execution (Williams et al, 1995). FMRI studies have similarly found activation of the supplementary motor area as well as of the parietal cortex during mental rotation (Cohen et al, 1996). Similar results have also been obtained from experiments that involve auditory imagery of melodies that activates both the superior temporal gyrus (an area crucial for auditory perception) and the supplementary motor areas. Further, metallization also affects the autonomic nervous system, the emotional centers and the body in same ways as actual perceptual experiences (Damasio, 2000).

To summarize, the increasing complexity of our society and economy places great emphasis on developing artificial agents, robots, smart devices and machines that can reason and deal autonomously with our needs and with the peculiarities of the environments we inhabit and construct. On the other hand, considerable progress in brain

science, emergence of internal model based theories of cognition and experimental results from animal reasoning has resulted in tremendous interest of the scientific community towards investigation of higher level cognitive functions using autonomous robots as tools. Rapid increase in robots' computing capabilities, quality of their mechanical components and subsequent development of several interesting (and complicated) robotic platforms, for example, Cog (Brooks, 1997) with 21 DOFs (Degrees of Freedom), DB (Atkeson et al, 2000) with 30 DoFs, Asimo (Hirose and Ogawa, 2007) with 34 DoFs, H7 (Nishiwaki et al, 2007) with 35 DoFs, iCub (Metta, Natale et al 2007) with 53 DoFs raise the challenge to propose concrete computational models for reasoning and action generation capable of driving these systems to exhibit purposeful, intelligent response and develop new skills for structural coupling with their environments. The computational machinery driving the physical/mental behavior of the GNOSYS robot proposed in this thesis contributes solutions to a number of issues that need to be solved to realize these competences.

Table 1 presents a concise summary of recent (some ongoing) efforts closely related and relevant to the nature of work described in this thesis. The columns represent the different projects surveyed and the rows represent the variety of topics under study. Since the areas of investigation are highly interdisciplinary, the presence of the tick mark merely indicates that the concerned topic is a major focus of investigation. Of course other interesting pieces of work (not surveyed in table 1) mainly the DARWIN series of BBD's (Edelman, 2006; Krichmar et al, 2005a,b) and the CLARION architecture (Sun, 2007) have also influenced the development of the work described in this thesis.

Projects	CoSY	GNOSYS	RobotCub	JAST	COSPAL	MindRaces	SPARK
Chief focus	A multi-disciplinary investigation of *requirements, design options* and *trade-offs* for human-like, autonomous, integrated, physical (eg., robot) systems.	A Domain Agnostic, goal directed, brain based cognitive architecture for integration of perception-reasoning-action generation cycle in embodied robotic agents.	Creation of a 54 degree of freedom cognitive humanoid robot (iCub) and investigation of the development of its cognitive skills.	Extend cognitive systems to multi-agent scenarios, mutual cooperation and development, with dialogue supported perception and action.	Combining symbolic reasoning with artificial neural networks	Moving from reactive to anticipatory cognitive systems, with expectation driven attention and analogical reasoning.	Nonlinear dynamical systems based architecture for action oriented perception with applications to mobile robotics.
Application /Validation	Explorer robot, Philosopher agent, helper robots	Animal reasoning inspired scenarios of physical cognition (tool use, tool making etc) in an unstructured world	Baby Humanoid of the size of a two and half year old child, embodied developmental approach to cognition	Two robots cooperating in a construction task, robot- human, robot-robot joint actions	Emulate capabilities of small children solving puzzles, problem solving etc	Proactive planning, Anticipatory emotions, Prediction by analogy	Dynamically emerging patterns, synergetics, emergent computing
Robot based exploration	✓	✓	✓		✓	✓	✓
Manipulation	✓	✓	✓	✓	✓		
Progressive learning	✓	✓	✓	✓	✓		✓
Attention Control		✓		✓	✓	✓	
Language or Gesture	✓		✓	✓			
Affordance	✓	✓	✓	✓			
Multiple Agents			✓	✓			
Self Adaptation	✓	✓	✓	✓			
Self organization		✓	✓	✓			✓
Symbolic Architetures	✓	✓	✓	✓	✓		
Subsymbolic architectures		✓			✓	✓	✓
Dynamical Systems		✓	✓				✓
Website for detailed information	http://www.cognitivesystems.org/	www.ics.forth.gr/gnosys	www.robotcub.org	http://www.euprojects-jast.net/	www.cospal.org	www.mindraces.org	www.spark.diees.unict.it

1.3 A few words about this manuscript

This thesis is organized into six chapters, the first being the current introductory chapter. The rest of the thesis is composed as follows.

Chapter 2 presents a general overview of the environmental set up (playground) we constructed for training/ validating the reasoning-action generation system of the GNOSYS robot, experiments from animal reasoning that inspired the design of the playground and the intricacies involved in different scenarios that the environment implicitly affords to the robot during phases of user goal/curiosity driven explorative play. A concise description of the functional specification, various perceptual/motor capabilities and low level control/software information flows in the GNOSYS robotic embodiment (the learner and performer) is presented in the initial sections. Some insight into the environmental complexity, the resulting behavioural complexity and the resulting computational complexity needed to generate these behaviours and finally visualizing the internal dynamics of a robot that is generating these behaviours is presented. Chapter 2 makes the point that we indeed deal with complex systems.
(key words: GNOSYS playground, experience precedes representation, animal reasoning experiments, environmental complexity, behavioral complexity and computational complexity)

In Chapter 3 we try to seek a simple intuitive explanation of how goals, constraints and choices can meet in a nonlinear dynamical system to give rise to spatio-temporal coordination of the various degrees of freedom in two different robotic bodies 1) the GNOSYS robot that we introduced in the previous chapter and 2) the 53 degrees of freedom baby humanoid 'iCub'. General formulation of the forward/inverse models for simulating/executing reaching movements towards a goal object using networks of task relevant parts of the body / internal body model, dealing with multiple task specific constraints at runtime by representing them as superimposed force fields and dealing with temporal constraints are presented along with several flow graphs, simulation and implementation results. After fine tuning the motor system of the GNOSYS robot, we describe the extension of the forward/inverse motor control architecture for whole upper body coordination (two arms and waist) in the baby humanoid 'iCub'. Extraction of general principles involved in creation of goal specific composite forward/inverse models and upstream integration of the proposed computational framework with higher level cognitive layers like reasoning are presented as we gradually navigate through this chapter.
(*key words: passive motion paradigm, terminal attractors, heterogeneous optimality, composite forward/inverse models*)

In Chapter 4, using powerful field computing principles applied on a growing neural gas, we describe how internal models for spatial topology and pushing can be learnt by GNOSYS through self organization of randomly generated sequences of sensorimotor data. During

the course of the discussion we also present experimental results from the robot's behavior that touch several important issues like representations, goals, sub goals, optimality, contradictions, reinforcements, causality, foresight and rationality.
(*key words: sensorimotor space, action space, activation dynamics, value fields, experience, contradictions, motor modulated anticipatory shifts in activity*)

Chapter 5: How can the robot reduce/distribute a high level goal into temporally chunked atomic goals for the different internal models? How can the robot do this flexibly for a large set of environmental configurations each having its own affordances and constraints (and manage the information generated due to interactions between different sub systems, virtual execution of different actions, resultant changes in the world, different functional goals)? What happens if the constraints in some environments do not allow the goal to be realised ? Can the robot mentally evaluate the fact that it is in fact impossible to realize the goal in that scenario ? Will it Quit without executing any physical action at all? If yes, does it have a reason to Quit ? and Can we see the reasons that caused the Quitting by analysing the field structure? In chapter 5, we seek intuitive, algorithmic and computational answers to some of the questions above, in addition to presenting four reasoning tasks that reflect different aspects of the performance of the reasoning-action generation system on the GNOSYS robot.
(Keywords: Abstract sensorimotor space, place map, goal space, Betty's task, Nonexistent objects, Quitting)

In chapter 6 we move from the inner most core of an 'atomic action' where fields compete to the outer most core of a 'WITS enabled world' where atomic cognitive agents compete, seeking to find unifying principles.
(Keywords: Abstraction, Circularity, Atomic Actions, Atomic Goals, Atomic Cognitive agents)

The Arena of Action

> You do not need something more to explain something more, all you need are the fundamental laws and **a large set of accidents.**
>
> MURRAY GELL-MANN
> The Quark and the Jaguar: Adventures in the Simple and the Complex

Toys have a history as old as human civilization itself. Little toy birds, carts, monkey's that slide across a string were excavated from the Indus valley civilization around 3000 BC. Dolls made of terra cotta with moveable limbs, sticks, bows and arrows were common toys used to play by children in ancient Greece and Rome. The functional role played by explorative sensorimotor experience acquired during play towards the overall cognitive development of an agent (natural/artificial) is now well appreciated by experts from diverse disciplines like child psychology, neuroscience, motor control, machine learning, linguistics, cognitive robotics among others. In simple words, play helps organize neural systems in the brain that ultimately mediate vital functions like vestibulo-motor capabilities, gross and fine motor skills, goal directed behaviour, problem solving, social interaction, abstract thought, negotiation and team work. No wonder, playing is the most natural thing we do and there is much more to it than just having fun. This chapter presents a general overview of the environmental set up (playground) we constructed for training/validating the reasoning-action generation system of the GNOSYS robot, experiments from animal reasoning that inspired the design of the playground and the intricacies involved in different scenarios that the environment implicitly affords to the robot during phases of user goal/curiosity-driven explorative play. A concise description of the functional specification, various perceptual/motor capabilities and low level control/software information flows in the GNOSYS robotic embodiment is presented in the initial section.

2.1 'GNOSYS Robot': The Acting Body

The GNOSYS robotic platform comprises of a number of sensors and actuators that allow the robot to perceive, learn, represent and act on the world as an embodied agent rather than a cognitive software abstraction. Keeping in mind important engineering issues like mechanical complexity, hardware complexity, software complexity, robustness, extensibility and flexibility of the system, a modular approach was followed throughout the architectural design of both the hardware and the software elements of the platform (Designed and

Assembled at ZENON Automation Systems, Athens). This implies that, for each type of sensor and actuator present on the platform, an independent hardware module is used, which is in turn controlled by an individual low-level software module. Integration of the entire range of the platform's functionalities takes place at a higher, more abstract level such that when seen from the level of reasoning, the body is coordinated as an unified entity (and not as a disembodied set of sensors and actuators).

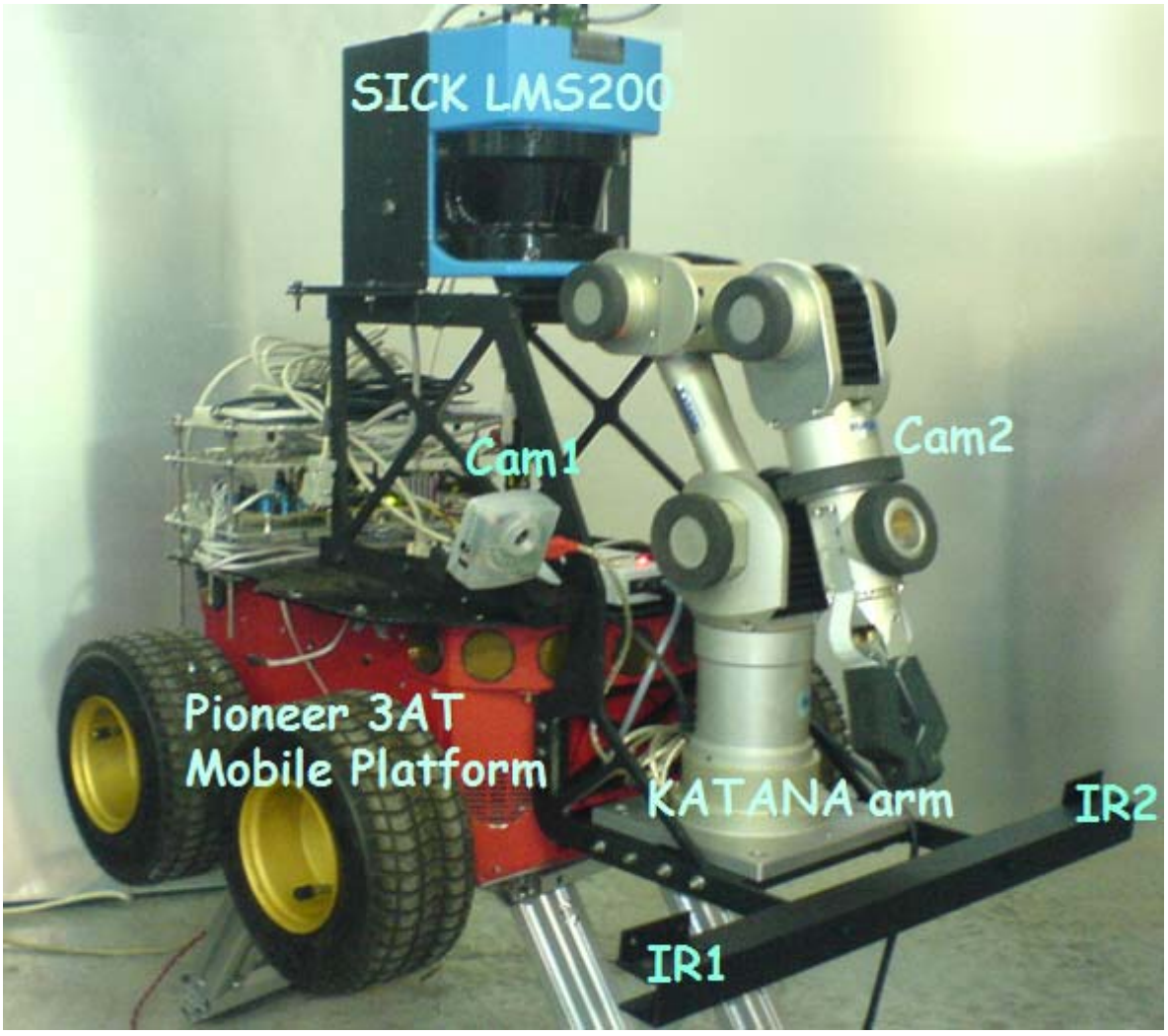


Figure 1. GNOSYS robotic platform: mobile base (Pioneer 3AT); Katana arm (5 degrees of freedom); stereo camera system (Cam1+Cam2); laser range scanner (Sick LMS200); proximity sensors (IR1+IR2). Additional sensors are not shown in the picture.

In order to illustrate how the various modules are integrated together as a complete robotic body, capabilities of all the modules (sensors and actuators) present on the robot and a brief overview of the underlying hardware and software architecture is given below. The basic robotic platform is an electric autonomous ground vehicle with slip (or friction) steering. It is wheeled but the wheels on each side are mechanically interlocked and act as a low friction

track. It is capable of turning in place when the commanded translational speed is zero. The robot fits within a 700 by 700 mm square envelop and has an effective payload capacity on level ground of approximately 30 kg which includes any additional non basic platform hardware such as sensors and manipulator arm. The following set of sensors are presently integrated on the robotic platform.

a) 8 Infrared Proximity Sensors

These sensors use infrared light triangulation to determine the distance to any obstacle directly in front of them. They are mainly used to detect the tables (which are about 20 cm in height and not hence detected by the laser scanner) on which different objects are placed to test reasoning scenarios. IR sensors have an effective range of 5 – 25 cm, with a precision of about 2-3 mm, depending on lighting and electrical noise conditions in the environment. They also act as contactless bumpers around the robot, in order to prevent collisions to walls as the robot moves.

b) Magnetic Compass

This Devantech CMPS03 compass provides an indication of the location of the magnetic North relative to the robot's orientation. It is intended to be used in orientation-related tasks, world mapping etc.

c) Inclinometers

In order to recognise potentially hazardous terrain situations, these sensors provide an indication of the robot's inclination in the horizontal plane (ground), by measuring the vertical gravitational force acting on them.

d) Thermopile

This Devantech TPA81 thermopile sensor can detect infrared radiation in the 2–22 μm wavelength range. It provides both an ambient temperature sensor, and an 8-pixel array of readings, arranged in a line directly in front of the sensor. This sensor is intended for detecting hazardous environments, as well as pin-pointing the location of a source of intense heat – e.g. a fire. Reasoning scenarios where this sensory information could be useful would be like a task of using a long stick to push an otherwise unreachable button (fire alarm) to alert the user, when high ambient temperature is sensed.

e) Sound Direction Sensor

This sensor module employs four directional microphones to measure the noise levels around the robot. It can be used for detecting and locating loud noises in the robot's environment.

f) Sound Frequency Spectrum Sensor

An additional microphone is used to gather audio information from the environment, and an FFT algorithm (Oppenheim, Schafer and Buck, 1999) is employed to provide the amplitude of discrete frequency bands in the audio spectrum.

g) Laser Scanner

SICK LMS200 laser scanner was used to extract global localization information of the robot situated in the GNOSYS playground. Discrete Markov models handle the topological aspects of the localization problem, i.e. perform coarse (topological) localization, and Kalman filters are used to filter the metric aspects, that is, elaborate on the previous discrete result and provide accurate localization. The LMS200 laser scanner uses the time-of-flight measurement principle to estimate distances up to 150m. It has a resolution of 10mm with ±4cm statistical error in the 8-20m range, and can scan a range of 180º at an angular resolution of 0.5º.

h) Cameras

Three Unibrain Fire-I cameras are mounted on the robot. Two of them are placed on either side of the robotic arm, and can be used for stereoscopic vision tasks inside the arm's working envelope. The third one is mounted on top of the robot, and is intended to provide scene images of the environment.

The following set of actuators enable the GNOSYS robot to move, manipulate and interact with objects in the environment.

a) Pioneer 3AT Mobile Platform

The Pioneer 3AT by ActivMedia serves as the basic chassis and wheels of the robot. It provides a sturdy platform, with enough carrying capacity to accommodate all of the components of the system. The basic platform actuators are the four wheel motors which are 120 Watt DC servomotors with software implemented as PID controllers at a 100 Hz rate. A set of four large motorized wheels ensure that the robot can perform in demanding outdoors environments.

b) Katana 6M Robotic Arm

The Katana 6M DOF robotic arm by Neuronics is a state-of-the-art articulated arm, offering a positional repeatability of ±0.1 mm in the workspace. Its light aluminium construction, combined with 6 harmonic drives, ensure optimal precision and stability with payloads of up to 500g. The operational degrees of freedom is shown in figure 2. The link lengths and the permitted ranges of movements for various joints is presented in table 1.

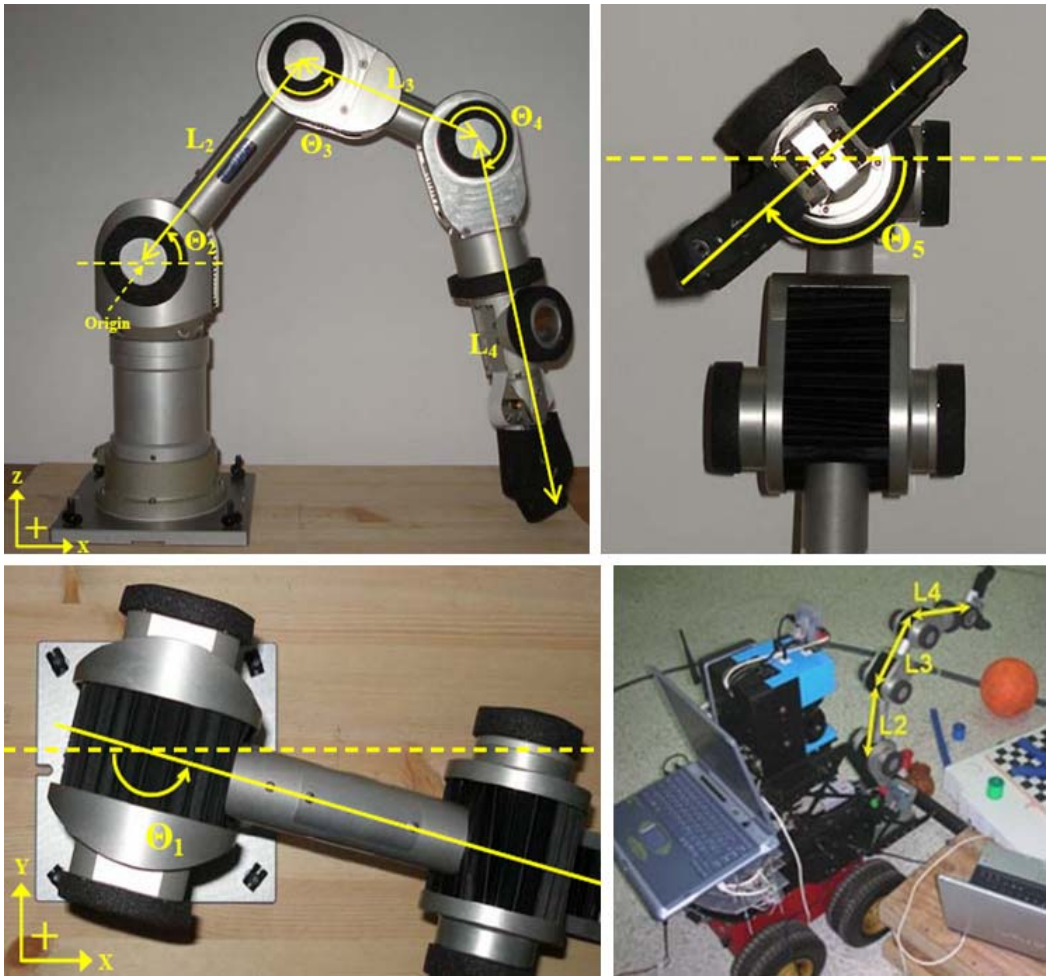


Figure 2. The KATANA Arm

Parameter	Value / Range
L_2	190 mm
L_3	139 mm
L_4	313.3 mm
Θ_1	+6.7° → +353.3°
Θ_2	-18.7° → +124.2°
Θ_3	+52.8° → +302.7°
Θ_4	+63.5° → +296.4°
Θ_5	+8.5° → +352.4°

Table 1. Link Lengths and Range of motion for the different DOF in the KATANA arm.

The GNOSYS onboard computational facility comprises of two individual computer units, each responsible for different tasks. The first of these, the Gumstix computer, is an embedded Linux computer that provides access to the custom sensory modules, through an I^2C data bus. Figure 3 illustrates the sensory devices that are connected to the Gumstix computer. The second onboard computer is a laptop PC operating on windows that handles

communication with the robotic arm, the Pioneer 3AT platform, the laser scanner and the cameras. In addition, the laptop PC offers ample computational power to run the middleware applications that implement the low-level functionalities, protocol translations etc. Figure 3 shows a block diagram how the various hardware devices (sensors and actuators)are interfaced with the onboard computers that run the low level software servers that handle bidirectional communications between the hardware and more abstract computational models situated in the brain of the robot.

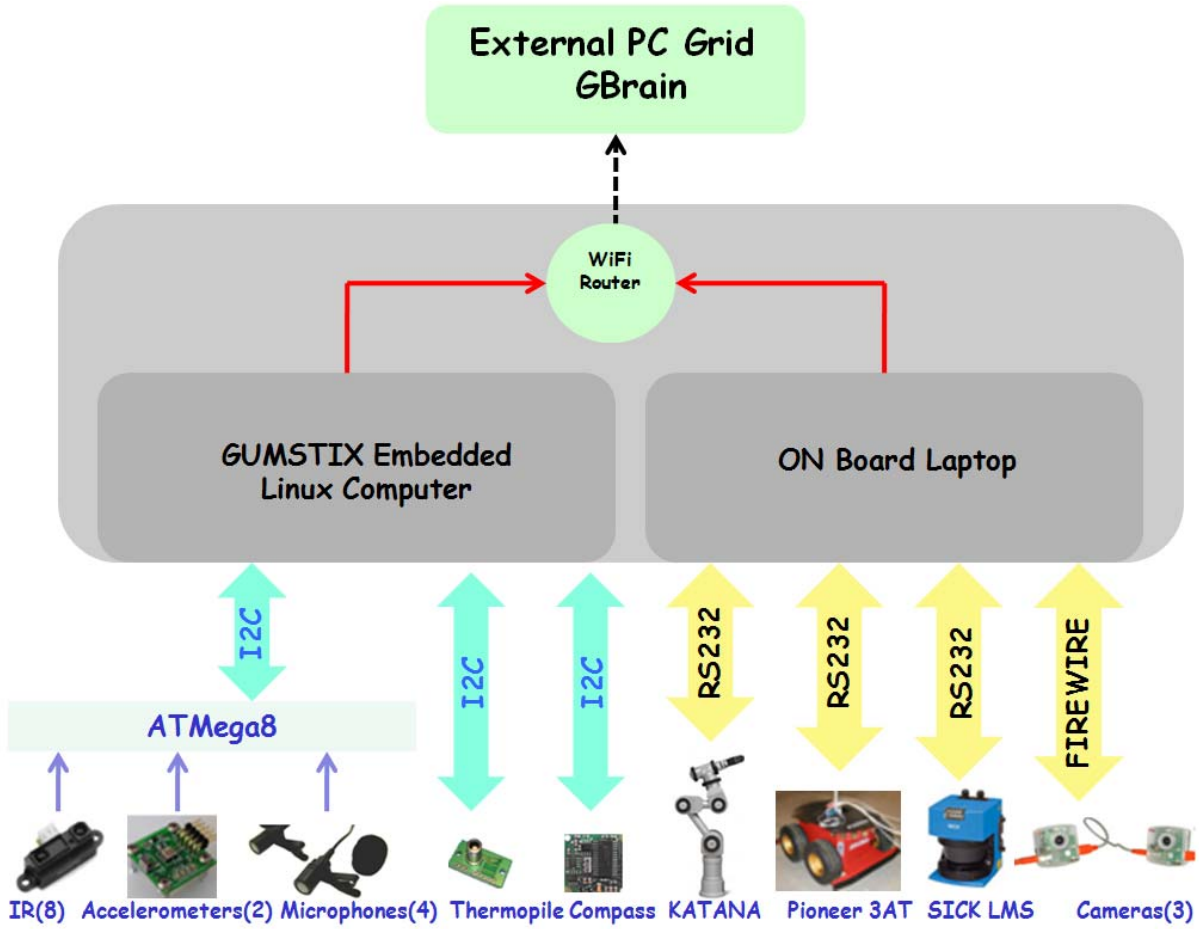


Figure 3. Hardware interfaces and lower level communication scheme

Low Level software integration

The different layers present in the overall software architecture of the system is shown in figure 4. At the lowest level there are six servers, each responsible for providing access to a distinct sensory-motor module. These servers do not communicate directly with the outside world, but instead they are linked through the middleware application called the GnosysAgent shown in figure 5. The middleware application basically aggregates the functionality of the six servers and provides access to them under a single entry point. The

GnosysAgent is also responsible for streaming data from the low-level servers to the layer above it called the Gnosys DLL. The Gnosys DLL is responsible for providing a high-level Application Programming Interface (API) for abstract software modules seeking to access and control the robot. In this way, any implementation details such as TCP/IP communications, multithreaded event pumps etc are hidden from the more abstract computational models.

The obvious advantage is the power of abstraction, a very powerful idea to tame complexity: hardware, computational and cognitive. In simple terms, a person working at the level of reasoning system of the robot need not worry about details like who made the motors that is fitted in the wheels of the robot. All he needs to know is the format of speed set (and rotation) commands he needs to communicate in order to make the robot move.

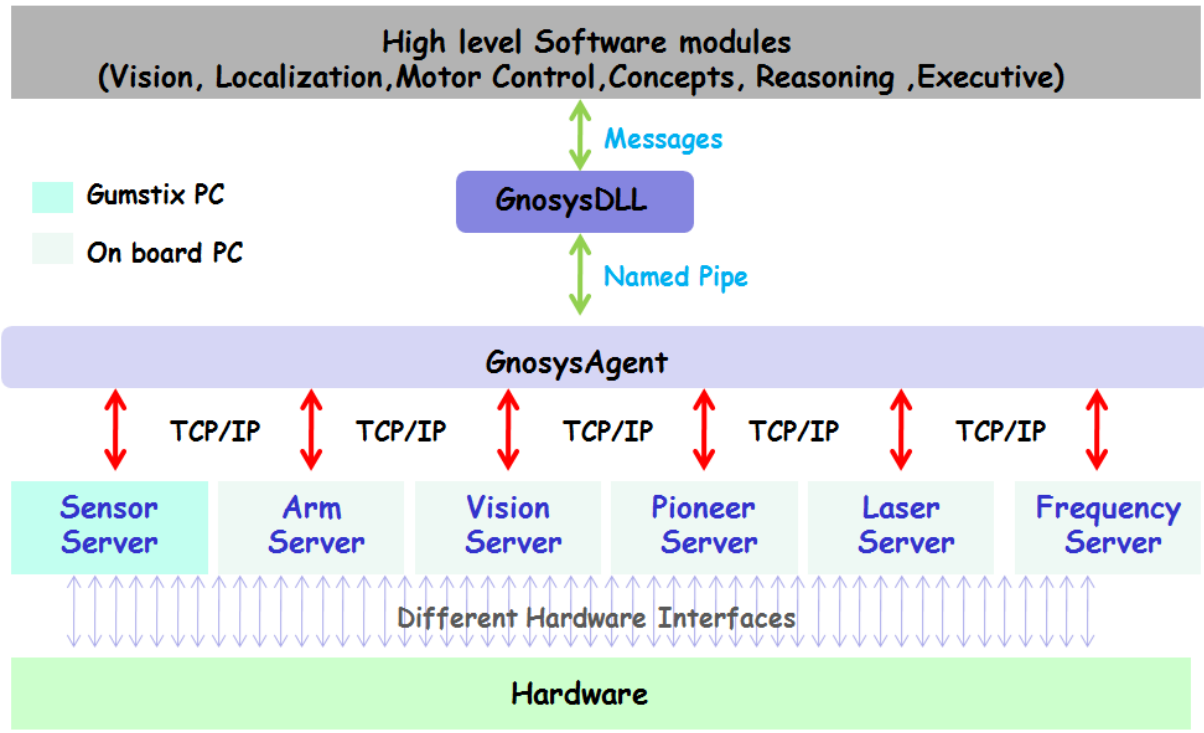


Figure 4. Layers in the Software Architecture

The GnosysAgent also has a built-in vetoing scheme to arbitrate the operation of the different functionalities in case of conflicting requests. For example, a "move forward" command issued through the Portal functionality will be vetoed by the "Bumper" functionality if an obstacle is detected in front of the robot. Moreover, if the new obstacle was placed dynamically when the

The obvious advantage is the power of abstraction, a very powerful idea to tame complexity: hardware, computational and cognitive. In simple terms, a person working at the level of reasoning system of the robot need not worry about details like who made the motors that is fitted in the wheels of the robot. All he needs to know is the format of speed set (and rotation) commands he needs to communicate in order to make the robot move.

robot was executing some goal, this veto mechanism at lower level can effect the

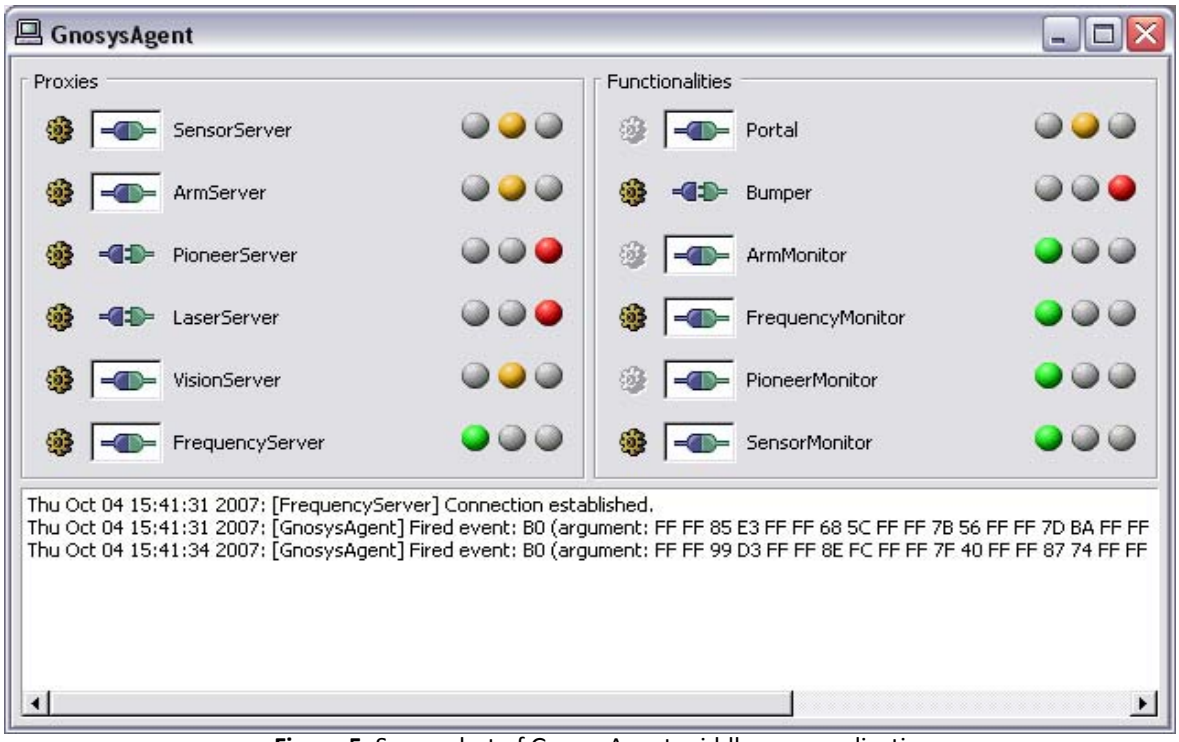


Figure 5. Screenshot of GnosysAgent middleware application

internal model representing the spatial topology of the playground in the brain (reasoning system) of the robot. As we will see in the forthcoming chapters, contradiction resolution mechanisms are a fundamental feature of the developed architecture and exist at all levels from concrete to abstract. In addition, the measure of inconsistency is directly used to trigger higher levels of reasoning.

2.2 'Animal Reasoning Experiments': Creating rich sensorimotor world's for robots

The earliest origins of rapid development in manipulative tactics, mental swiftness, memory capacity and social sophistication, paralleled with rapid neocortical enlargement can be dated back to the simian primates about 25 million years ago. Common chimps with binocular vision and fingered dexterous hands are certainly investigative animals, with the ability to quickly learn environmental correlates (Pavlovian conditioning) and use the results of their exploration to guide future behavior (Skinnerian conditioning). They are routine tool-makers with elaborate social customs of tool use, effective group hunters of relatively large prey, living in loose communities somewhat dominated by coalitions of males, with occasional inter-community lethal aggression (Bryne, 2007; McGrew,1992).In the past few years, there have been many studies of animal cognition that reveal interesting behaviors demonstrated by animals that have all the hallmarks of similar brain processing in humans. Such experiments generally focus on many open problems that are of great interest to the

cognitive robotics community, mainly attention, categorization, memory, spatial cognition, tool use, problem solving, reasoning, language and consciousness.

Even though studies of complex cognition have been traditionally focused on monkeys and apes, recent results show that a larger circle of animals appear to be involved in different forms of cognitive behavior: a) crows, wood peckers and other birds can fashion tools appropriate for extracting food from receptacles by means of bending pieces of wire into suitable hook-like shapes; b) corvids exhibit social reasoning behaviors such as caching food[1] and reburying it in a different site if they see that they have been observed by another individual of their species (Emery and Clayton, 2004); c) Egyptian vultures crack open ostrich eggs by selecting and using appropriate stones as tools; d) herons are skilled in bait-fishing; e) ravens can solve difficult puzzles such as untangling a knotted string to free up a tasty treat[1].

On the other hand, considerable progress in brain science coupled with rapid increase in robots' computing capabilities and quality of their mechanical components has resulted in tremendous interest of the scientific community towards development of artifacts that autonomously evolve their cognition and develop new skills in structural/functional coupling with their environments. Further, the growing body of robot-based research has now begun to make serious claims to be a new tool for studying biology. The general idea is to develop an understanding of natural systems by building a robot that mimics some aspects of their sensory motor systems, their nervous systems and their behavior. So can scenarios related reasoning and complex cognition demonstrated by animals be replicated in robots? By doing so is it possible to gain better insights into the computational basis of these intriguing behaviors' that make us feel that animals are both sensitive and intelligent like us? Is it possible to understand 'how' animals actually apply insight to problem-solving? Is it trial and error based associations due to repeated contact with their environments or is it that they have a more deeper understanding of the causal forces at play? do they understand abstract concepts and are capable of learning higher level symbolic systems ()?

Animal reasoning experiments generally focus on many open problems that are of great interest to the cognitive robotics community, mainly attention, categorization, memory, spatial cognition, tool use, problem solving, reasoning, language and consciousness.

[1] A nice review of latest progress in animal cognition can be found in a recent three part NATURE mini series titled 'Inside the animal mind'.

Guided by experiments from animal reasoning, we constructed a playground for GNOSYS robot that implicitly hosts experimental scenarios of tasks related to physical cognition known to be solved by different species of primates, corvids and children. Special focus was given to tasks that engage the robot goals similar to essential day to day functions of cognitive agents (humans and animals) in their natural environments.

Creating biologically inspired habitats of varying levels of complexity and subtlety for robotic artifacts is both a nontrivial problem and an extremely worthwhile experience. It is a non trivial problem because of the necessity to maintain a balance between simplicity and complexity (in some ways also a balance between science and technology). While over-simplifying the environment (restricting the system to navigation in a maze, reaching targets, tracking red and blue objects etc) can mask the real computational problems, over- complicating the environment presents non trivial problems in dextrous manipulation, visual perception, active sensing, robustness in low level software, communication systems, energetic constraints, computational power, latency among others. It is also necessary to ensure that the environment is complex enough such that for a same user goal, small changes in the environment structure (and combinations of objects present in it) require a significant change in behaviour/strategy employed by the robot to realise the goal. A cleverly designed environment for a robotic artifact can basically serve three important purposes

a) facilitate exploration-driven development of different sensorimotor contingencies of the robot, development of goal-dependent value systems, internal representation of different cause effect relations, outcomes of interventions etc.

b) help revealing to the designer the underlying computational mechanisms that may be at play (and in turn need to be incorporated into the cognitive architecture) based on the amazingly infinite ways by which goals may be realized in different scenarios. This stems from the fact that the problems the human brain faces, while allowing the body to act successfully in unstructured worlds, are very similar to the ones cognitive scientists face to make their robots behave in a similar situations. The only difference is that while trying to make robots do even seemingly trivial things in changing environments, we become more conscious of the enormously complex integrative apparatus that spells synergy among the universe of information, relationships, choices, sensors, and actuators when we effortlessly realize similar goals in similar environments. Many animal psychologists are in agreement with the fact that studies on tool related behaviour may be unusually revealing about the underlying cognitive processes and the level of understanding involved in animals' manipulation of physical objects (Bluff et al, 2007).

c) serve as a test bed to evaluate the performance of the system as a whole and make comparisons of the robot's behaviour with that of real organisms, other cognitive architectures.

Guided by experiments from animal reasoning, we constructed a playground for GNOSYS robot that implicitly hosts experimental scenarios of tasks related to physical cognition known to be solved by different species of primates, corvids and children below 3 years. Instead of focusing on puzzle solving exercises, inductive and deductive reasoning tasks prevalent in literature that require the system to answer questions of inference, special focus was given to ecologically realistic scenarios that engage the robot in tasks similar to essential day to day functions of cognitive agents (humans and animals) in their natural environments as emphasized in (Sun, 2004). In this way, even for simple high level goals like reaching a ball, the complexity of the reasoning process and actions needed to achieve them could increase more than proportionately with the complexity of the environment in which the goal needs to be accomplished. Care was also taken to ensure that the experimental scenarios provide opportunities for the agent to accidentally (or through trial and error) learn to exploit affordances provided by various objects available in the playground. The following subsections present a summary of the major experiments from animal cognition literature that inspired the design of the GNOSYS playground.

2.2.1 The n-sticks paradigm

This is an interesting paradigm related to tool use employed by experimenters working with chimpanzees. It is a slightly more complicated version of the task in which the animal reasons about using a nearby stick as a tool to reach a food reward that was not directly reachable with its end-effector. Visalberghi et al (1993, 1996, 2000) studied the reasoning powers of chimpanzees to determine if they could extract general rules in order to obtain a reward by suitable tool use in a scenario consisting of a clear tube, open at both ends, with a food reward inside it that could be pushed out of either end by means of a stick, which was available to the chimpanzee. The chimps successfully managed to extract the food from the tube by pushing it with a tool of suitable length. If presented with tools of different lengths during a trial , chimps often chose the most appropriate tool directly and did not employ any trial and error based policy of testing with all the available tools. Tool selectivity is critical for animals because, selecting an improper tool incurs costs in terms of time, often resulting in the potential loss of the food to another predator. As reported by Chappell and Kacelnik (2002) and subsequently confirmed by several others, crows are also very selective in choosing the most appropriate tool suitable for a particular task. In a similar task of 'pulling a food basket out of a tube', crows often chose tools that precisely matched the geometry of the tube in which the food was trapped. Both longer (hence difficult to maneuver) and shorter tools were consistently neglected.

The task of using one stick as a tool is itself non trivial (computationally) in the sense that it requires an understanding of the relative discrepancy in distance, impossibility to get the reward by means of the direct use of the end-effector, selection of a tool of suitable length based on the geometry of the tube, and then the crucial process of forming a *sub-goal* to grasp the tool, update the kinematics of the arm and then start all over again. The two-sticks paradigm involves two sorts of sticks: Stk1 (short), and Stk2 (long), one of each being present on a given trial, only the small one being immediately available, and the food reward only being reachable by means of the larger stick. Figure 6 shows a pictorial representation of the two-sticks paradigm. The above scenario can be suitably extended to involve n sticks, however we limited the number of sticks to three in the experiments with the robot.

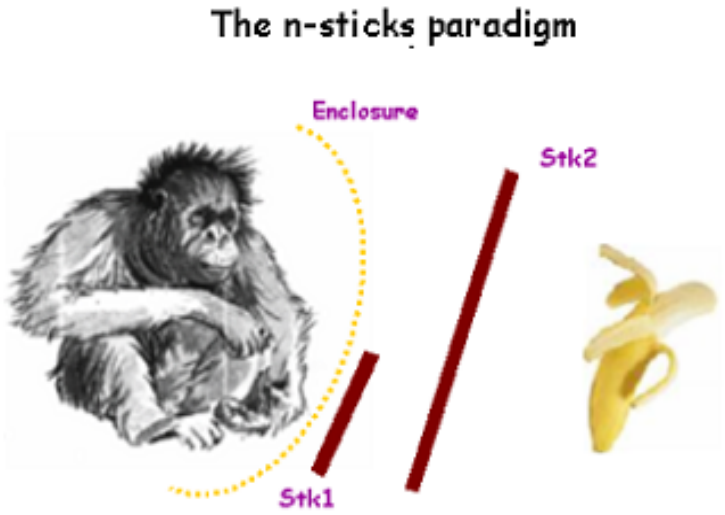


Figure 6. The two-sticks version of the n-sticks paradigm. The chimp is unable to reach directly the food due to an enclosure and learns how to get it via the two sticks.

To obtain the food reward, the animal can go through the following sequence of virtual or real actions:

I. Try to virtually reach the food using the end-effector (which is unsuccessful);
II. Search for tool of suitable length, find stick Stk2;
III. Initiate first sub-goal to grasp the long stick Stk2. Again this leads to lack of success, leading to further search;
IV. Observe the small stick and imagine using it to reach the longer stick. Initiate second sub-goal to grasp the small stick Stk1. This action is successful;
V. Come back to sub-goal-1, now try to reach the long stick using the short one. Since there is already a stick in its hands, it cannot grasp the longer one as of now, so using the short stick, it must manoeuvre the large stick within reach of the end-effector;

VI. Release Stk1 and reach Stk2, which is now directly accessible, and grasp it;

VII. Use the large stick to manoeuvre the food within reach, release Stk 2 and obtain the food.

The task of using two sticks contains in itself various other smaller reach/grasp sequences, also including the task of using one stick (like in the experiments of Visalbergi and Limongelli, 1995,1996). The paradigm is also computationally interesting in the sense that it necessitates integration and use of several concepts like the notion of a goal, goal directed motor imagery using forward/inverse models to realize effectiveness of different virtual actions with/without tool coupled to the end-effector, priorities in scheduling different sub goals and combined use of vision, reasoning and movement.

2.2.2 'The Cognitive Hook maker': Betty's novel tool making task

An interesting case of novelty in behaviour was demonstrated by Betty, the Caledonian crow who lived and "performed" in Oxford under the discrete scrutiny of animal psychologists (Chappell & Kacelnik 2002; Weir et al, 2002, Emery & Clayton 2004), when she faced the problem of extracting a food basket kept at the bottom of a transparent vertical tube, and when her usual tool was not available to her. Instead, she found a straight piece of wire lying nearby. Betty had already had experience of successfully using a bent twig to extract the food from the vertical tube and also the experience of playing with and bending flexible pipe cleaners a year ago. She is now faced with the non-trivial task of determining how to retrieve the food from the tube using a straight wire. After a short hesitation, she pushed one end of the wire into the tape holding the tube and moved the other round with her beak, making a hook, using which she successfully extracted the food basket from the vertical tube. She repeated this performance 90 percent of the times she faced the same situation. A pictorial representation of the problem faced by betty is shown in figure 7.

It is not uncommon to encounter similar situations in our daily life when we have to reason about things that do not yet exist, but could exist as a result of our actions.

Figure 7. Giving a mental look: Tool making New Caledonian Crows.

Even in the wild, the Caledonian crows are known to make hook tools by trimming and sculpting twigs until a functional hook has been fashioned (Kacelnik et al, 2004). Hook tools are then used to poke out insect larvae from holes in trees using slow deliberate movements. The crows also manufacture stepped-cut Pandanus leaves (Emery and Clayton, 2004; Kenward et al , 2005), which are used to probe for prey under leaf detritus, using a series of rapid back-and-forth movements or slow deliberate movements that spear the prey onto the sharpened end. These tools are consistently made to a standardized pattern and are carried around on foraging expeditions. The ability to purposefully fashion and use tools may also be correlated with the fact that that corvids (crows, jays, ravens, jackdaws) have remarkably large brains for their body weight, at about the same ratio of brain volume/body weight as other primates proficient in tool use. In addition to using tools, the New Caledonian crows also display behaviors found in other corvids which are often thought to be associated with high cognitive abilities, such as breaking nuts by dropping them from branches (Hunt et al., 2002; Layard & Layard, 1882), and food-caching.

Rates of success higher than chance level can be obtained by avoiding the trap through a rule of action based on either some associative processes or else by understanding the cause–effects relations between some key features of the task such as the trap, the reward, and the outcome of the pushing action in terms of the displacement of the reward.

It is not uncommon to encounter similar situations in our daily lives when we have to reason about things that do not yet exist but could exist as a result of our actions. In addition to the computational features necessary to tackle the n-sticks paradigm described in the previous section, Betty's novel hook making task further requires scenarios and mechanisms facilitating accidental learning through physical intervention of the robot, opportunistic exploitation of these isolated experiences/memories present in the state space (among many other irrelevant experiences) in the context of present goal, creation of the virtual objects through virtual actions (using the forward/inverse model pair in this case) and finally the ability to see the utility of these virtual objects in the context of the active goal. The mere curiosity to explore computational architectures that could account for the creative process demonstrated by Betty, the types search spaces involved, the means to constrain them, the nature of representations needed, the means to acquire them (reliably in the case of robots), the planning mechanisms needed, memory retrieval mechanisms in play prompted us to develop an artificial version of the paradigm for the GNOSYS robot.

2.2.3 'The trap tube paradigm': Knowing how the tool works

This paradigm was not considered for initial experiments on the robot. It was the result of a natural evolution in the computational architecture of the reasoning system driving the

behaviour of GNOSYS that prompted us to attempt this classic scenario from animal cognition. The trap tube task is an extremely interesting experimental paradigm that has been conducted on several species of monkeys and children (between 24-65 months), with an aim to investigate the level of understanding they have about the solution they employ to succeed in the task. As mentioned in section 2.2.1, chimps faced no problems in successfully extracting food rewards from the transparent tube by pushing it with a tool of suitable length. Now in an extended scenario (Visalberghi ans Tomasello, 1997), a trapping third tube closed at its distant end (shown in figure 8a) was then inserted in the middle of the main tube, with the trap hanging downwards. In this scenario, a subject can be 50% successful in every trial simply by systematically inserting the stick into the same side of the tube or by inserting it into one of the two sides by chance. In contrast, rates of success higher than chance can be obtained by avoiding the trap through a rule of action based on associative processes or else by understanding the cause–effects relations between some key features of the task, such as the trap, and the outcome of the pushing action.

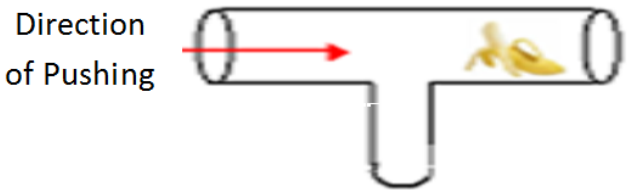


Figure 8a. A trap is positioned in the center of the horizontal tube

In this modified scenario, the animals needed at its simplest, to always insert the stick into the main tube at the end most distant from the reward. This task was solved by the chimps very quickly with a success of around 98.5%. This task was also solved by some capuchin monkeys and careful analysis of their behaviour showed that the capuchin Roberta was actually choosing the side of insertion based on the spatial configuration of the tube, trap and the reward.

In order to probe how much the animals understood as to what they were doing, the apparatus was then modified so that the trapping tube was repositioned close to one end of the main horizontal tube.

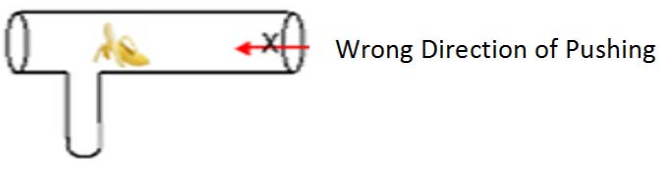


Figure 8b. Modified trap tube task.

The rule ‘always put the stick in at the end farthest from the reward, then push’ would have failed every time in this case, since it would only work if the reward was also closer to the free end (not the distant free end) than the trapping tube. Otherwise applying the rule would always cause the loss of the reward. It was found that in this modified and more difficult paradigm the two initially successful chimpanzees were still successful; however the successful capuchin on the original tests failed totally on the modified paradigm. As noted by

the experimenters, data favor (but do not prove) the hypothesis that attributes success of the chimps to an understanding of the causal relation between the tool using action and its outcome. In other words, this also implies that (see also Visalberghi & Limongelli, 1996) the capuchins were 'able to learn how to use the tool but failed to understand how the tool actually functioned'. Children below three years of age were unable to figure out a successful strategy for the trap tube paradigm, whereas older children succeeded to carry out the task in a few trials. Moreover, they verbally explained the reasons of choosing the direction of pushing and also explained what would have happened otherwise. This classic experiment suggests that different species of primates may be using different strategies in solving a task, even though all of them may be successful at it. Furthermore, simple variations of the same problem can limit the success of different species towards achieving the goal (because of the lack of identifying/understanding all causal relationships involved).

Of course a robot that is capable of solving the n-sticks paradigm is going to fail in the modified trap tube task during the first few trials. This failure contradicts robot's earlier experiences of carrying out the same actions, which were actively rewarded by the user for achieving the goal of grasping the attended object. Can this contradiction at the level of reward values be used to trigger still higher levels of reasoning and/or exploration activities in order to seek the cause of failure? What new computational modules need to be incorporated into the reasoning system to achieve this? The problem in short is to have computational mechanisms that help resolving *cognitive dissonance*, a term derived from the earliest works of Leon Festinger (1957). Dissonance in short is an aversive drive that motivates a cognitive agent to change either its behaviour or its belief in an effort to maintain 'psycho-logical' consistency in its global knowledge space (its neural connectivity structure). To achieve this computationally, the robot must have at least the following three capabilities:

(a) achieving awareness that, for some reason, the physical world works differently from the mental (simulated) world;

(b) identifying the new variables in the environment that determine this inconsistency (in the trap tube case the robot should discover that the essential novelty are the holes/traps introduced by the experimenter;

(c) initiating new actions that can block the effect of this new environmental variable (change the direction of pushing the ball i.e. away from the hole/trap in the simplest case).

To summarize, assuming that the world is consistent, an indication of having incomplete set of cognitions about a behavioral system (trap tube task in our case) should motivate the robot to engage in explorative drive reduction activities in order to infer missing relations, add new knowledge in such a way as to create closure, completeness, or consistency in its global knowledge space. A computational mechanism accounting for cognitive dissonance is critical in taking the giant leap from developing rational to rationalizing agents and this was

the only motivation to include the classic trap tube paradigm as the final scenario to be incorporated in the playground of the GNOSYS robot.

2.3 The GNOSYS Playground

Guided by experimental scenarios from animal reasoning, we set up two different environmental configurations for the GNOSYS robot. Initial experiments mainly related to spatial goals, such as fetching and transporting objects, Grasping, reaching, playing with objects on the table, pushing objects etc, were conducted in an open lab environment shown in figure 9. In later stages, we constructed a closed square room 3×3m in dimensions, inside which more specific experiments related to animal reasoning tasks were conducted (figure 10). As we can observe, unlike the open lab environment, this set up also imposes some special constraints in navigation that we intentionally incorporated to conduct experiments related to learning heterogeneous optimality criteria for navigating towards different spatial goals. In the following three sub-sections we present a general overview of three different aspects related to the experimental set up:

a) Objects present, the complexity levels introduced in the environment based on their different possible combinations, the kind of opportunities they afforded to the robot

b) The range of possible experiments that could be conducted in the playground and the corresponding anticipated target behaviors' of the robot that could be studied under various conditions;

c) The computational complexity involved in different tasks, the set of computational models that need to be developed/learnt by the robot and their coupled interactions.

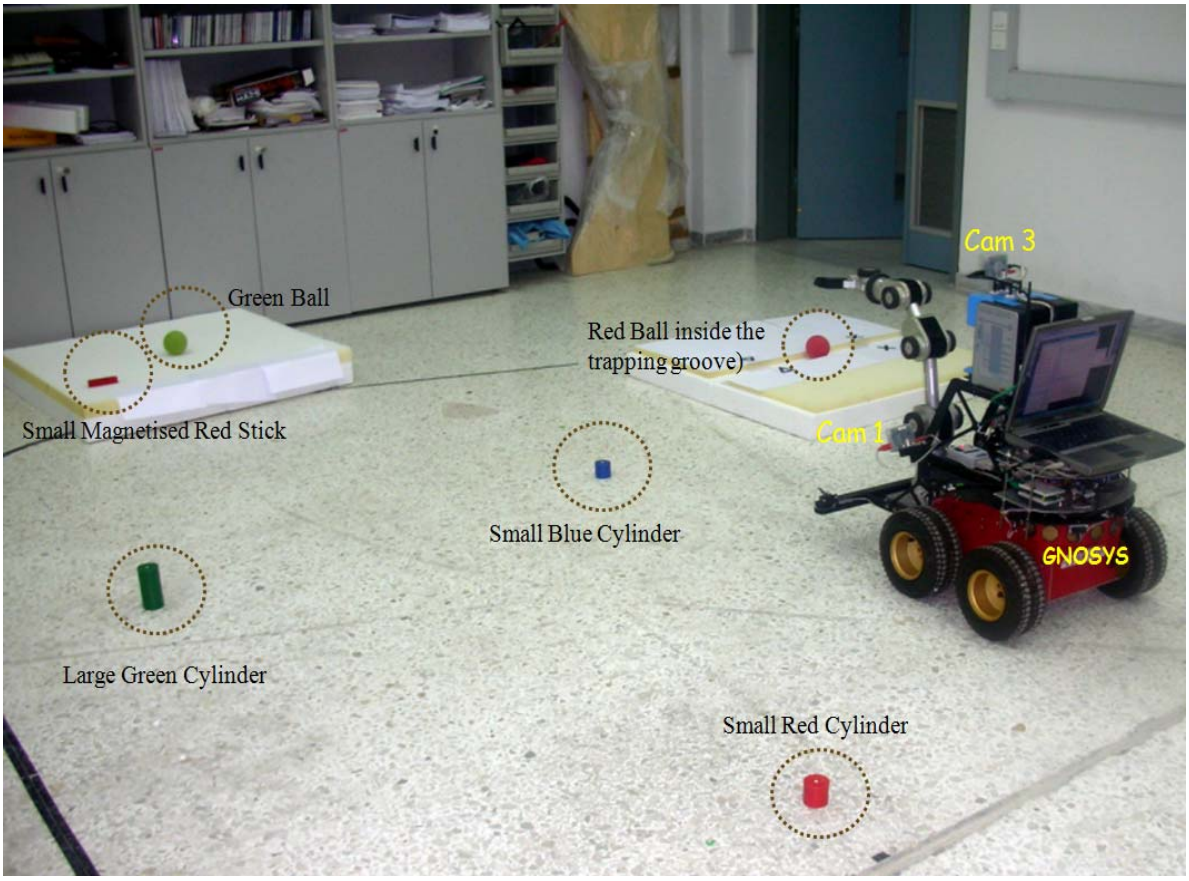


Figure 9. GNOSYS playground in the early phase of the project with objects placed at different locations in the room.

2.3.1 Environmental Complexity

The environment in which the GNOSYS robot is situated consists of the following set of objects placed randomly either on the floor or on the tables.

(a) Cylinders of different colours and variable sizes (small/big) of reasonable weight to facilitate manipulation; in the advanced set up, cylinders were replaced with coffee cups of different sizes. Stacking cylinders was a usual task that we employed at many times to test the overall accuracy of the action generation system, forward/inverse models for reaching, camera to arm calibration, 3d reconstruction of salient points of the goal object in the camera plane into Cartesian coordinates, low level control and message passing schemes. In addition the robot was also given spatial goals to collect cylinders placed in the floor and put them at a common place in the lab.

(b) Sticks of various lengths (10, 15, 20 and 25 cm). Both sticks and longer cylinders could be used as possible tools in scenarios emulating the n-sticks paradigm. Among the collection of tools, the three small red sticks 10 cm in length are magnetized. The basic aim in having small magnetized sticks in the playground was that while playing

with different sticks (using a motor primitive of bringing one stick already grasped by the robot to the edge of another stick placed in the table) the robot can discover and gain experience of an additional affordance provided by a small subset of sticks (i.e. to make even longer sticks), just like the case of Betty having had experience of playing with flexible pipe cleaners a year before she was presented with the task of retrieving the food basket using a straight wire.

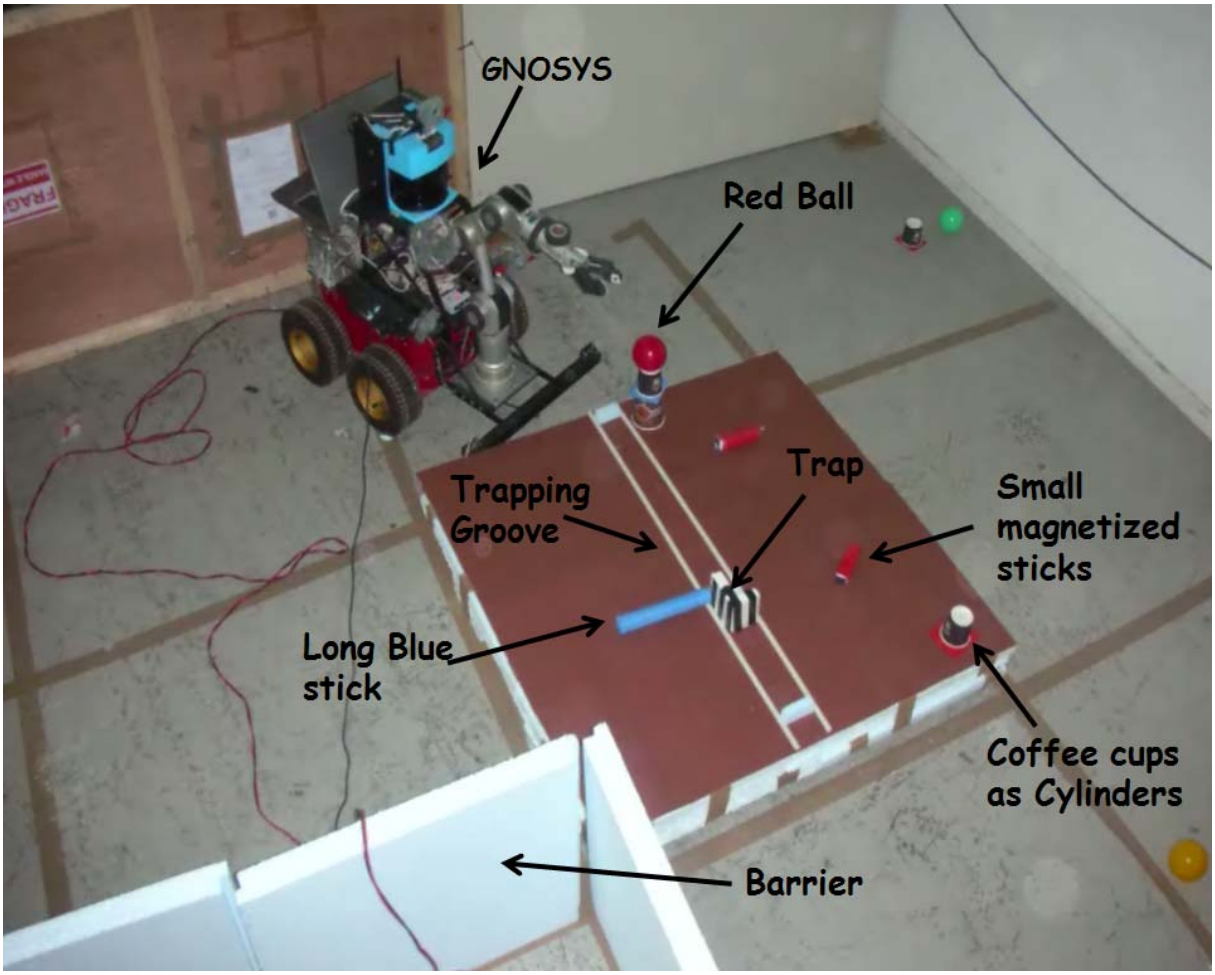


Figure 10. Top view of a typical GNOSYS environmental set up during the advanced stages of the project. It is a 3m by 3m enclosure (every unit square is approximately $1m^2$ in area) with the table placed in the center of the playground. Various tools, goal objects, trapping groove and trap are also seen in the picture.

In this way, the robot is provided with an opportunity to reason and exploit this additional feature to make longer tools for reaching trapped/unreachable objects, in case it encounters a situation in which a proper long enough tool is not available to it (assuming that the robot has had previous experiences of playing with magnetized sticks and has internally represented the related experience as a memory trace).

(c) The environment consisted two smooth tables constructed with most convenient height for the manipulation, clearly visible to the two cameras and detected by the infrared sensors fitted near the wheels of the robot. The tables were large enough such that objects can be placed on them such that they are unreachable when approached by the robot from any side (this makes reasoning about using/making long sticks obligatory). In one of the tables (shown in figure 10), horizontal grooves were cut and run across the table from one side to another, so that the gripper can slide sticks along a groove from either end to push out a rewarding object to the edge of the table (this could eventually result in spatial activations that drive the robot to move to the edge of the table closest to the object). Traps could be placed all along the groove so as to prevent the reward from moving to the edges of the table when pushed by the robot (similar to the trap tube paradigm);

(d) In this environmental layout, the robot is asked to pursue relatively simple high level user goals like reaching, grasping, stacking, pushing and fetching different objects. The interesting fact is that even though the high level goals are simple, the complexity of the reasoning process(and subsequent action generation) needed to successfully realize these goals increases exponentially with the complexity of the environment in which the goal is attempted. For example, let us assume that the task of the robot is 'Grasp the red ball' in the scenario shown pictorially in figure 11. In order to achieve this goal, the robot must be capable of reasoning about and virtually execute the following complex chain of actions (in the proper temporal order) and further be able to continuously foresee their resulting consequences:

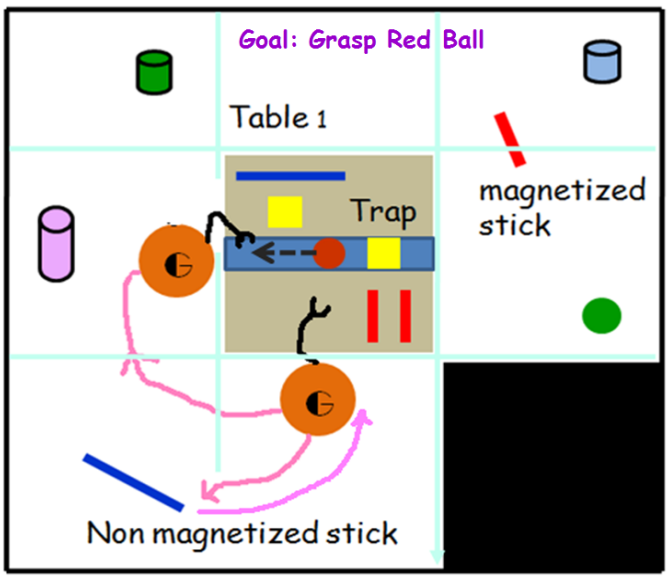


Figure 11. Typical environment in which user goals like reaching, grasping, fetching various objects are issued to the robot.

a) Detecting the possibility of grasping the long blue stick (with the assumption that the robot is aware of this object through the global place map and is able to predict its

possible utility in reaching the otherwise unreachable ball using the reaching forward/inverse model);

b) Verifying the possibility of returning back to its initial position close to the table after grasping the long blue stick;

c) Verifying the possibility of initiating the pushing action (provided that it has learnt that pushing fetches greater reward, in such situation, than any other action;

d) Choosing the proper direction of pushing based on the location of the trap (provided that it has had previous experiences and failures about the presence of the trap and has learnt its effect);

e) Predicting the location of the ball as a result of the pushing action (using the pushing internal model);

f) Discovering the possibility of navigating around the table and reaching closer to the predicted position of the ball after the pushing action (using the internal model representing spatial topology);

g) Verifying the possibility of grasping the ball directly with its end-effector in case it actually executes step 'f' (using the forward/inverse model for the arm);

h) Initiating, accordingly, real actions in the physical space to realize the goal.

The example basically illustrates the difference between 'reaching goals' and 'reaching targets'. The difference is that in the former case, there are several intermediate stages of reaching, grasping, pushing and moving that are not specified by the user goal, and the robot has to discover all the intermediate chunks of situation- action information based on its own knowledge of the way the world works. Further, using a set of few sticks, balls, traps and cylinders and combining/placing them in different ways, an enormous amount of complex environmental situations can be created, the only limitation being the imagination of the experimenter itself.

2.3.2 Behavioral Complexity

An approximate insight on the desired global behavioral space of the robot in response to the corresponding environmental space is of great help while shaping the global computational architecture of any cognitive machine that is expected to behave flexibly and robustly in the world. Moreover, this piece of information is also critical in deciding the basic assumptions to be made in the theory, for example, what built-in primitives to begin with, how experience is gained, how learning takes place, the interactions between different communicating sub-systems, the modular divisions in software structures etc to mention a

few. The only way to estimate the range of behaviors that could possibly be generated by the robot in different environments is to put ourselves in the shoes of the robot and imagine what we would have done in similar situations. Hence, before the development of the computational architecture of the reasoning system, we briefly surveyed the kind of behaviors we expect the robot to generate for a select set of environmental configurations that in themselves cover a broad range of possible user goals, the original scenarios from animal reasoning and their complex combinations.

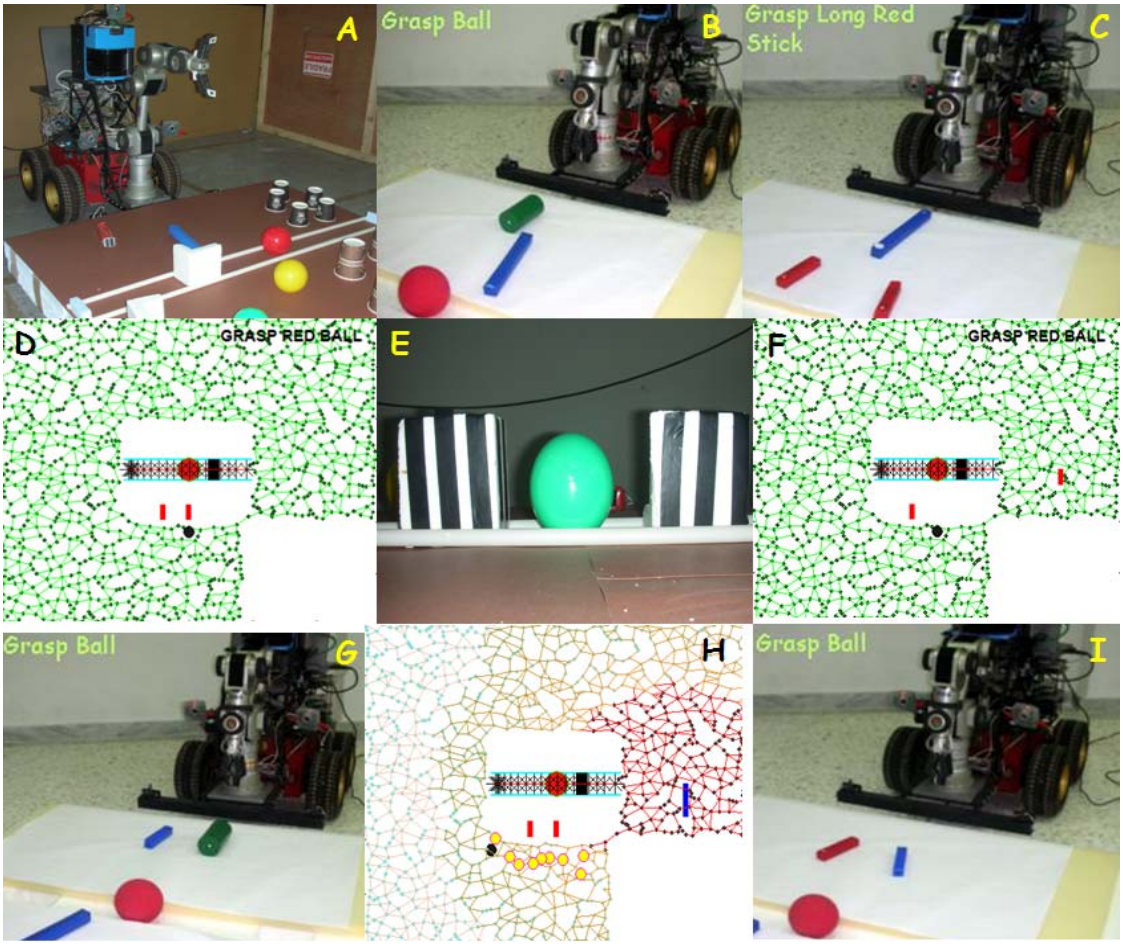


Figure 12. Examples of target scenarios and user goals used to test flexibility in reasoning capabilities of GNOSYS robot .

Figure 12 shows nine possible configurations of the environment in which the robot is asked to execute different user goals (after the completion of learning phase).

The goal in panel 'A' is to grasp the ball and the solution is a simple case of direct tool use, tool coordination and navigation.

Panel 'B' is a case of n-sticks paradigm with the twist that the robot must use the long cylinder as the tool1 to get hold of the long blue stick (tool2). Of course, this implies that it is not an object, but a feature (length in this case) which the object affords that must form the appropriate basis for selection of a tool (at the level of computation). This fact is even

reflected in the writings of the great American psychologist William James (James, 1890) when he poetically illustrated the different uses of a book (a thing that gives knowledge, a paper weight, an instrument to swat a mosquito) and so on.

The goal of the scene in panel 'C' is to grasp a long red stick. As a matter of fact, there are no long red sticks in the environment. The only red sticks present are the small magnetized ones. If the robot has not had previous experience of trying to couple different sticks together, the result of this goal is a loss of motivation. But if the robot has had experience with the magnetized sticks, then it must have a memory trace of having made a long red stick represented sub-symbolically in the abstract reasoning system. Can the robot exploit this experience? This task looks similar to the Betty's tool making task, but has one major difference. The difference is that in the Betty's tool making case, higher levels of reasoning are triggered by the non convergence of the forward inverse model simulation (to reach/grasp a goal object like the green ball), while in the case of attempting a goal to grasp a physically nonexistent object, it's the non convergence of physical search (through vision), that initiates a higher level of reasoning, making the agent reflect on what more could be done, and hence providing the opportunity to act creatively. After figuring out the fact that it can actually make a long red stick and grasp it, the robot further needs to solve the n-sticks paradigm to grasp the two small red sticks in the first place.

The functional role played by explorative sensorimotor experience acquired during play towards the overall cognitive development of an agent (natural/ artificial) is now well appreciated by experts from diverse disciplines like child psychology, neuroscience, motor control, machine learning, linguistics, cognitive robotics among others. In simple words, play helps organize neural systems in the brain that ultimately mediate vital functions like motor-vestibular capabilities, gross and fine motor skills, goal directed behaviour, problem solving, social interaction, abstract thought, negotiation and team work. No wonder, playing is the most natural thing we do and there is much more to it than just having fun.

Panel D is a straightforward artificial reconstruction of Betty's tool making task.

The goal of scenario in panel 'E' is to grasp the green ball. It is an advanced case of trap tube paradigm, in which there are traps at both ends and the reward is in between the traps. This scenario was not pre planned and did not exist in the demonstrable tasks even when the computational architecture for reasoning was implemented and tested on the robot. A mere curiosity to see how the robot behaves in this case and more importantly why it does so (based on the internal representations of various fields that are responsible in shaping the external actions), prompted us to try this task. This interesting scenario will be analyzed in more detail and explained in computational terms in the forthcoming chapters.

The task of panel ‘F’ is similar to betty’s tool making scenario with the difference that there are sequences of navigation involved during the process of creating the long red stick. This can also result in the agent attempting to save energy by not returning to its start position to initiate the pushing. Instead the agent can grasp the stick in front of it, move towards the other small stick placed in the floor, couple it to make the appropriate tool, and now while returning, it need not come all the way back, it can initiate the pushing action from the other side of the table (i.e opposite to its starting position). In computational terms, this implies that the knowledge related to pushing must be internally represented in ways that are independent of the orientation of the robots body.

The scenario in panel ‘G’ is a case of quitting because no tools are immediately available.

The scenario in panel ‘H’ is once again interesting because there are two possible solutions to grasp the red ball. The first one is to use the long blue stick placed in space for which the robot has to spend energy in navigating all around the room. The second solution is to make a long red stick and use it. How should the robot choose between these solutions? Can energetic constraints be incorporated into the reward structure? There can be many other scenarios where more than one tool may be available to the agent which we deal with in the forthcoming chapters.

The scenario in panel ‘I’ shows two small sticks, one red and other blue. Since the blue ones are not magnetized, the possibility of making a long stick does not exist, so one possibility for the robot would be to lose motivation and quit the goal. However, if we have mechanisms of memory retrieval in the computational architecture that need only a small cue to recall/reconstruct (a more correct word often used by Endel Tulving, 2000) a complete past experience (temporal sequence of objects, situations and actions), then the one available small red stick is a sufficient cue for the robot to extrapolate the complete process that eventually leads to the possibility of having a long red stick. Assuming that the architecture for reasoning is powerful enough to achieve this, what would then be the possible behavior of the robot? Our guess was that, under these circumstances, the robot should do what we would have done if we faced the similar situation, i.e go on a spatial search in the physical space to find the missing raw component that is needed to make the tool that is present in its mental space. Before concluding this section, we note that the nine examples presented here are just a small subset of the whole gamut of possible scenarios that could be presented to the robot. By analyzing them we not only get an idea of the various shades and richness in behavioral space that different environments (and goals) elicit, but also get useful hints related to the necessary global computational structure of the cognitive architecture that drives the robot to generate these fascinating behaviors’.

2.3.3 Computational Complexity

In this section, we briefly overview the different computational and software structures that need to be created in the GNOSYS robot to provide the necessary perception-learning-reasoning-action generation powers required to satisfy user goals flexibly in a range of environmental scenarios described in the previous sections. A block diagram of high level interactions between major computational modules implemented in the robot is shown in figure 13. Even though reasoning and action generation (RAS) will be the central focus of discussion of this thesis, in this section we quickly summarize the interface we had with other computational architectures (mainly at the perceptual level) developed by tireless (and timely) efforts put by other collaborating teams in GNOSYS: 1) Foundation for research and technology, Heraklion , Greece; 2)ZENON Automation Systems, Athens, Greece; 3) King's College, London, UK; 4) University of Tubingen, Germany .

The sensorial and conceptual layer is bidirectionally interfaced with the RAS. The main computational models present in this layer are examined in the following.

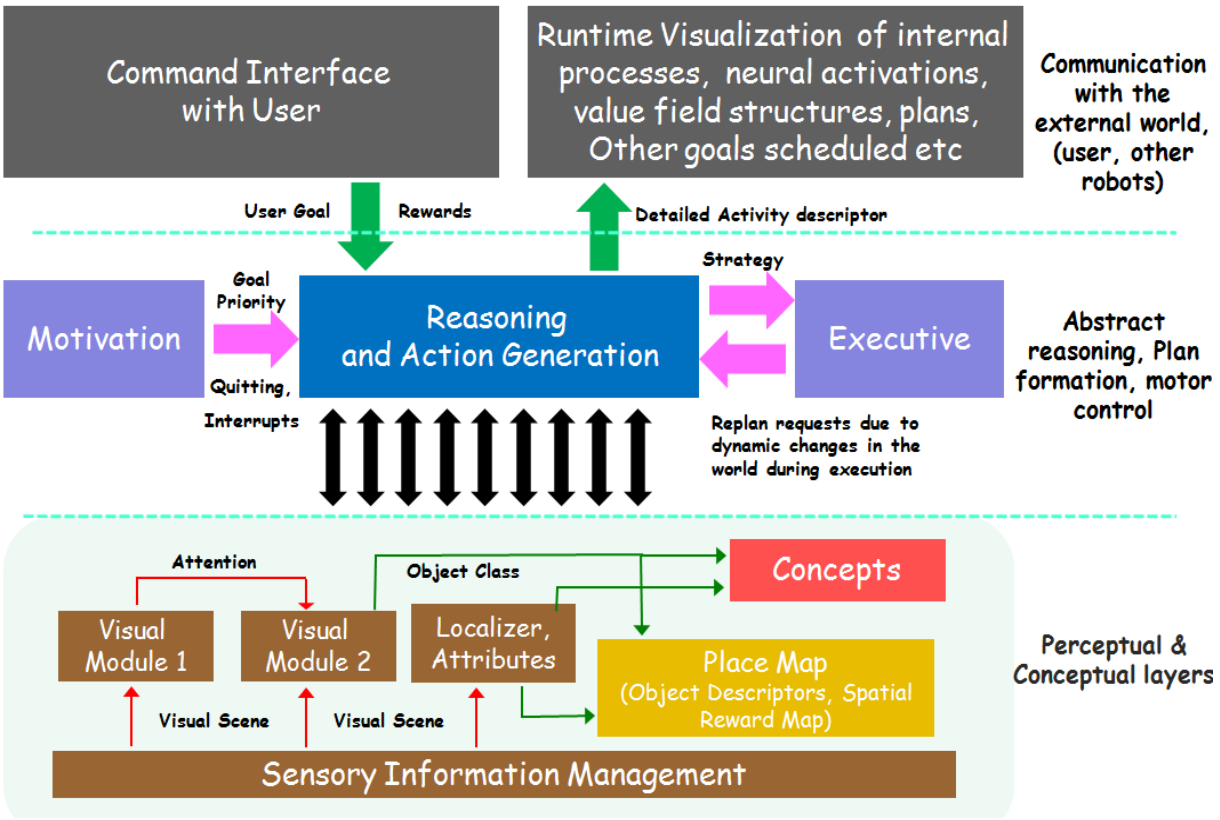


Figure 13. Fundamental interactions and information flows between major computational models driving the behavior of GNOSYS robot

Visual Perception system: The hypothesis regarding various objects present in a visual scene is made based on the scene analysis of two separate vision systems: visual module 1 (VM1) and visual module 2 (VM2). VM1 uses a sparse sampling approach based on Scale-Invariant-Feature-Transform (SIFT: Loncomilla and del Solar, 2005; Lowe, 2004) that in its current state is capable of learning representations (of pre-segmented) objects in real time. Although

there exist methods (Mikolajczyk and Schmid, 2005; Serre et al, 2005) which have been shown to outperform SIFT in recognition performance, the SIFT algorithm still gives the best trade-off between performance and computational complexity. This module is further capable of detecting multiple objects in scenes in real-time. For an image size of 320 x 240 pixels, the achieved frame rate is between 4 and 10 frames per second (including object classification). VM2, on the other hand, employs a dense sampling approach using a hierarchical feed-forward set of neural network modules, each composed of an excitatory and an inhibitory layer, approximately emulating the known modular structure of the human visual system (Taylor, Hartley and Taylor 2005; Taylor, 2006). While the major advantage of the VM1 is the ability of fast scene analysis, sparse sampling can also lead to several problems including ambiguity in the object representations due to effects of non-Gaussian noise, lack of information between key-points etc. On the other side, the full (dense) processing of the input scene using the neural networks in VM2 even though very accurate, is computationally expensive, and hence limits the image size able to be processed in order to achieve almost real time visual scene analysis when requested by the reasoning system. Hence the strategy for fusion of the two visual systems was to exploit the advantages of both approaches. Instead of reducing the image resolution (which we found was a real drawback in complex scenes during reasoning tasks), VM1 was used to calculate and pre-process the regions of interest (ROI) in the visual scene in real-time. These ROIs were then sent to the more accurate VM2 for a dense analysis and subsequent recognition of objects present in the visual scene. Interested readers may refer to (GNOSYS docmentation) for more detailed information regarding the GNOSYS visual perception systems. The output of the process of scene analysis at a computational level is an array of object descriptors. Each object descriptor contains information about the shape, color and location of the center of gravity of the object in the image plane (as seen by both cameras). This information is used by the 3D reconstruction system a module belonging to the action generation system described in detail in the next chapter.

3D reconstruction system: This system reconstructs the location of an object in the camera plane (given by the visual systems) into corresponding reconstructed coordinates in the Euclidian space to be subsequently used by the computational models for coordination of movements of the robot. This system also has the function of estimating other physical attributes of the object like its longitudinal size (a parameter useful in tool selection), orientation of the object with respect to the body of the robot (especially for the sticks that are placed on the table, or floor). In the next chapter, we will enter into greater details regarding the computational architecture of this module.

Localizer: The localizer is responsible for keeping track of the global position and orientation of the robot in the playground using information received from the SICK LMS200 laser scanner. This critical information is directly accessed by both reasoning and action

generation systems to plan/coordinate a range of goal directed movements of the robot. Interested readers may refer to (Baltzakis, 2004) for more detailed information about the global localization system in the GNOSYS architecture.

Place Map: Place map is a memory that keeps track of all objects (and information related to their salient features) detected by the visual system during iterations of scene analysis. In simple terms the place map keeps track of collection of objects available for potential use by the robot in order to realize its goals. During subsequent cycles of visual analysis if objects already present in the place map are detected again, then the information related to them is refreshed based on the latest updates. The important feature of the place map is that it is not static but keeps changing dynamically as a result of several processes taking place in the reasoning system. For example as the reasoning process virtually acts on objects present in the place map, the resultant structure of the world can potentially change after the interventions initiated by the reasoning system. These changes are reflected in the instantaneous informational scope of the place map. We will dwell into greater details about these issues in the chapter related to the reasoning system. The third source who can potentially act on the place map is the user himself. We decided to add this facility so that during reasoning scenarios, the user can prohibit the robot from using certain objects in the world, hence forcing it to change strategy, go into exploration, try using other objects etc.

Motivation: The motivation system influences the priorities of the goals being executed (scheduled for execution) and also couples to the reasoning system (by modulating the instantaneous agent body state) thereby influencing the actions recommended by the reasoning system. A goal's priority is influenced mainly by the following parameters:

a. 'Extrinsic value' which is given either by the user or from experience (built by reinforcement learning);

b. 'Intrinsic Value' which is updated every step by the motivational system and in a way captures the value of the goal to the agent's well being. This enables the same goal to have different values at differ times, irrespective of the same external state;

Normally a Goal is terminated if it converges to its target state, if it is executed a set number of times, or it is stopped by the User. However there can be pathological cases where Goals might not converge to their targets, due to environment changes, interference from other Goals' plans or other cases (like sudden introduction of a trap in the trapping groove). At every time step the motivational system also updates the 'Commitment' and 'Engagement' variables that capture the withdrawal / persistence that the agent places on the goal. Interested reader may refer to Kasderidis et al (2004, 2006) for further details on this topic.

A prolonged period of low importance (intrinsic value) or of suspended execution automatically causes the goal to be eliminated from the system.

Reasoning and Action generation system: This system is the central focus of this thesis. The action generation system is responsible for controlling all the movement related aspects of the robot both in the mental and physical space. It is a collection of closely coupled internal models mainly, 1) a forward inverse model for reaching that coordinates a range of arm (arm+tool) movements taking into account a range of constraints in the intrinsic and extrinsic spaces. 2) an internal model representing the spatial topology of the playground that is concerned with all space related planning, body movements of the pioneer platform 3) An internal model representing the forward inverse dependencies involved in the pushing action 4) as set software processes that handle all intercommunications between these internal models and the whole action generation system with the rest of the computational architecture using an agreed protocol of communication.

The user directly communicates with the reasoning system to present goals and rewards to the robot. The reasoning system basically is responsible for the transformation of an high level user goal into a plan that can be executed by the robot in order to physically realize the goal. In order to achieve this, the reasoning system uses the global knowledge accumulated by the agent as a result of its various motor explorations and the collection of value fields that are learnt based on the rewards presented by the user. This knowledge is represented in the connectivity structure in and between the neural maps representing the sensorimotor space of the robot and influences the dynamics of the reasoning system during the process of transformation of an user goal into a suitable plan to be sent for execution. The dynamics of the reasoning system may also transform a high level user goal into sequences of sub goals (directed towards objects estimated to be of potential use), that need to be accomplished in order to realize the root goal. Seen from the level of reasoning and execution, there is no difference between a user goal and a sub goal other than the fact that the sub goal was initiated by the system itself. Every sub goal (just like the root user goal) instantiate their own processes of reasoning (which may in turn incapsulate even more sub goals) and trigger the various forward inverse models in the action generation system to virtually execute their plans. This implies that the computational architecture is inherently recursive in nature. A tracking process continuously monitors the progress of the reasoning system towards realizing the goal and keeps track of anomalies detected between mentally simulated information and the real sensory information received from the environment. Any inconsistency between the top down and bottom up information flow alters the behavior of the robot from normal dynamics to explorative dynamics. The tracking process is also responsible for terminating the process of reasoning once the goal is realized, reinitializing the system and updating the global knowledge space in case there were explorative actions initiated and new knowledge gained during the complete cycle of reasoning.

Executive: The output of the reasoning system is a temporally arranged sequence of motor actions, directed towards different objects in the environment, henceforth called the plan descriptor. The plan descriptor in itself is self contained in the sense that it holds all the necessary information starting from an abstract chain of actions to the detailed level of motor commands that needs to be sent to different actuators to perform the action. This plan descriptor is the input to the executive system. Since many goals are simultaneously processed by different reasoning threads, the executive may have more than one plan descriptors to be scheduled for execution. The executive system combines different plans together and reorganizes them into mutually exclusive domains of 'action', so that different non overlapping actions may be carried out concurrently, navigation can be minimized so as to save energy and environmental resources can be used cleverly. In sum, while the reasoning system attempts to solve the 'how' part of the problem, the executive deals with the 'When to do what' part of the problem and ensures efficient use of the resources: agent's body and opportunistic possibilities afforded by environment that may enable the agent to gain maximal rewards.

In the next few chapters, we will be dealing in detail with many of the subsystems mentioned in this section so as to better understand their internal dynamics, learning and behavior. The objective of this section was to present a global summary of the overall computational architecture, the major information flows and interactions between different important computational modules, the manner in which goals, perceptions, reasons and actions come together to drive the behavior of the robot in complex environments.

2.4 Visualizing the life of a cognitive machine

Despite being structurally and functionally similar, it is in fact the contributions of the large set of accidental forces 'physical and psychological' that we encounter in our lives that makes each one of us unique in terms of our goals, behaviors, choices, interpretations and actions. No wonder, it often requires an analyzing brain of a cybernetic detective to peep into the mysteries hidden inside the mind of another human, to discover the subtle forces that have shaped his thinking and to discover the reasons behind his actions.

The prime reason for the difficulties involved in interpreting the cognitive space of another agent stems from the fact that when complex bodies interact autonomously with complex environments in different time instances, learning different things, some through accidents, some through observations and some through interventions, a large amount of information is generated that remains local to the knowledge space of the interacting agent, and which in turn influences its behavior. Humans empowered with the ability to create powerful symbol systems and language have a means to make at least some of that local knowledge (that which is consciously accessible) global or easily accessible to others, hence being able to give birth to culture, science and of course the ultimate form of abstraction

'mathematics'. No wonder, children around the age of 3 years when tested on the trap tube paradigm (section 2.2.3) could not only solve the task, but could also tell the experimenters the knowledge of causality that they employed to solve it, and even what would have happened in case they had changed their strategy. Most theorists and experimental psychologists agree that the extent to which non human animals understand their physical world is an extremely controversial issue mainly due to conceptual and empirical difficulties. Even in the example of the trap tube paradigm, 'how' and 'how much' the capuchins understood when they succeed in the simpler version and failed in the more complex version of the same task, what was the strategy that led the chimps to success in both cases? Was it due to associations they had learnt after few trial experiments or do they have a more general understanding of the physical causality? Do different species of animals tested use different strategies even though all of them are successful? All these questions are still objects of debate.

Despite being the creators of the computational architecture, we still have to pay the cost of giving the system autonomy, in terms of creating ways to peep inside the mental space of the robot just like a cybernetic detective analyzing the human mind or an animal psychologist making sense of why the animal does what it does. The only difference is that we have to do this for a time scale that is not limited by the time of the experiment, but by the life of the cognitive machine itself.

Now let us put ourselves in place of the animal psychologist, and the robot in the place of an animal attempting to autonomously learn, act and handle similar experiments, environments inspired from animals reasoning. It is easy to visualize the fact that we ourselves will be facing the massive task of making sense of the various streams of information reaching different parts of the cognitive architecture, the visual scenes being analyzed, the associations being made through time, the actions being executed, the value fields being learnt due to experiences, the plans being developed, the virtual simulations taking place in the mental space in the different internal models, the sub goals being formed etc and all this in a world that itself is changing as a result of the physical interventions made by the robot and the interventions made by ourselves in order to set up the environment for the robot. As we will see in the forth coming chapters, the reasoning and action generation system of the robot has been created in ways such that even though the structure of organization of information is defined by us, the content (of information) is self organized by the robot itself as a result of its pseudo random explorations in the play ground. Further as seen in figure 13, the complete architecture is a collection closely interacting high level computational modules, each of them being networks of dynamical systems themselves. Hence, despite being the creators of the computational architecture, we still have to pay the cost of giving the system autonomy, in terms of creating ways to peep inside the mental

space of the robot, analyze the internal dynamics of all these interacting dynamical systems, to make sense of what is going on inside the architecture, what were the reasons that are driving the actions of the robot, what was the learning that influenced this behavior, what new is being learnt in the system, why does the robot do what it is doing and predict what will be the behavior of the robot if we modify the world, and do all these for a time scale that is limited by the life of the cognitive machine itself. A closely related problem is finding ways to explain in simple terms to an external observer the global dynamics occurring inside the cognitive architecture during the performance of any behavior by the robot i.e the problem of transforming instantaneous, goal specific, world specific, knowledge specific sub symbolic representations in the cognitive architecture in forms that are appropriate to the human sensory, perceptual, and cognitive systems.

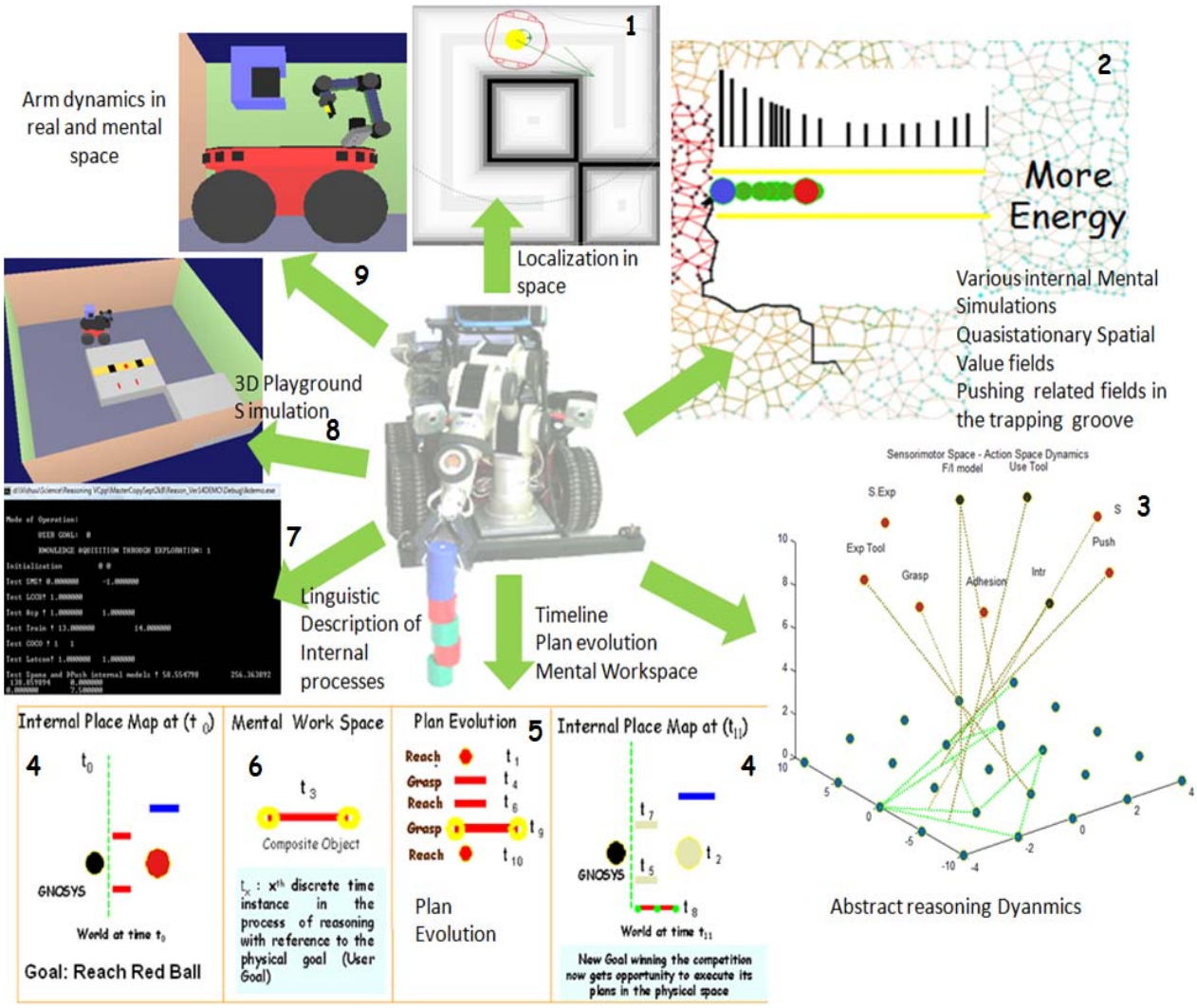


Figure 14. Nine different visualization interfaces currently in operation in the GNOSYS architecture in order to monitor on-line different facets of the internal processing.

Figure 14 shows nine major visualization interfaces currently in operation to present different facets of the behavior of the GNOSYS robot online (and in real time) during

learning, performance of goal directed reasoning tasks and other experiments for testing computational modules etc. The tenth visual interface that is not shown in the figure is a panel that relays the outputs from the cameras, processing of the images and the results of the scene analysis done by the visual perception systems.

However, the objects recognized by the vision systems, automatically enter the place map (2.3.3 D) and the neural representations of the place map are transformed into the objects they correspond to and are displayed in panel 4. Panel 4 as we can observe is refreshed as a function of time. Figure 14 shows two different images relayed to panel 4 in different time instances (t0 and t11). As the robot acts on the world, the world keeps changing and this is reflected in the state of the world seen in panel 4. As we can see, there existed two small red sticks at time t0 and at time t11 there is a long red stick in the world. This implies that the robot has acted on the world and made a long red stick. At t11, we also see that some objects are shaded in white, this implies that the objects are no longer available for use during subsequent reasoning tasks. Panel 4 in addition to representing how the world looks like, also gives a venn diagram like representation of the problem, such that the distance of objects from the robot is preserved (with appropriate scaling).

Panel 5 shows the high level goals existing in the goal space of the robot, these goals may be either user goals or sub goals created during the process of reasoning. The top most goal commands highest priority for execution at that point of time. At the level of execution, there is no difference between a user goal or a sub goal, and priorities of various goals are a function of time. Any changes in goal priorities is reflected in this panel. Panel 6 is depicts the contents in the mental work space of the robot. The objects in the world that are currently being considered by the forward inverse models as possible tools, imaginary objects created during reflections are displayed in the mental workspace. The only distinguishable difference between an imaginary object and a physical object at the level of reasoning is that the mentally created object has no arguments related to spatial location (a critical piece of information that is needed in order to make any physical actions on it). We will get into more details on these subtle issues in the chapter of reasoning. Panel 7 is the most precious piece of information for the developers of the reasoning–action generation system while performing any kind of testing, software upgradation, new functionality addition in the system. It gives a linguistic description of the all information flows, action executed, knowledge gained, goals created etc during a fixed time frame. Panel 1 is the output of the localization system and at all times keeps track of the global position and orientation of the robot in the playground using information received from the SICK LMS200 laser scanner. Panel 8 is a 3D simulation of the robot in the playground with the same software/control interfaces as the real robot and can be used to train/test the cognitive architecture remotely in simulation mode (with all the hardware, visualization, computational models working in the same fashion as with the real robot).

Panel 9 simulates the arm movements based on the outputs of the forward/inverse model for reaching that we will consider in the next chapter. Panels 2 and 3 display the subtle internal dynamics taking place in the growing neural gas based internals model representing spatial topology of the playground, pushing and the abstract reasoning system. The goal dependent value field that drives the dynamics is color coded and superimposed on the growing neural gas, red representing higher values.

Also seen in panel 2 is the value field for pushing in the trapping groove at that instance of time. In addition, trajectory of the ball as a result of pushing, the trajectory of the body in space during different simulations (for example, what is the sensory consequence of pushing the ball to the left with a particular amount of force, will doing so be useful in the context of the active goal etc) can be observed in real time form this panel. By analyzing the resulting field structure (wrt time) in different sensorimotor spaces for different environmental scenarios, it is not only possible to get a subjective view of the internal dynamics that is causing the behavior, but also predict what will be the resulting behavior when the environment is modified, what features are lacking in the current version of the computational architecture, how can the models be made more powerful, what new experiments can be created and tried on the robot using different permutation and combinations of objects in the environment etc. In addition, there is also an interactive learning scheme implemented in the reasoning system, in which the robots chooses its action and the user acts as the environment and textually enters the consequences of its action. In this way, the user can interact with the robot and transfer some of his knowledge into its global knowledge space, which will then begin to have an influence in its behavior. This way of text based communication was also created in the reasoning system keeping in mind the future experiments we plan to carry out, when there are multiple cognitive agents acting in the world, competing and cooperating for resources from the environment in order to realize their goals.

> Just like the history of the universe is co-determined both by the basic laws and an unimaginable long sequence of accidents or outcomes of chance events (as evident in the statement of physicist Murray Gell-Mann in the beginning of this chapter), just like the history of evolution itself, for example the transformation of a set of photosensitive cells into something that has the structure and beauty of a human eye is a result of random accidental tinkering (in words of French biochemist Francis Jacob), the behavior of any cognitive agent (natural/artificial) is also a function of a large set of accidental forces (physical and psychological) arising from the environment, that sculpts its global knowledge space, its reasons and its actions.

To conclude, in this chapter we initially introduced the GNOSYS robot, the body we aim to control and 'observe in action' in this thesis. An summary of the sensing and actuation capabilities, hardware interfaces, software layers, how information moves back and forth from the hardware layer to the higher level neural systems in the Gbrain was given in 2.1. Just like the history of the universe is co-determined both by the basic laws and an unimaginable long sequence of accidents or outcomes of chance events (as evident in the statement of physicist Murray Gell-Mann in the beginning of this chapter), just like the history of evolution itself for example, the transformation of a set of photosensitive cells into something that has the structure and beauty of a human eye is a result of random accidental tinkering (in words of French biochemist Francis Jacob), the behavior of any cognitive agent (natural/artificial) is also a function of the large set of accidental forces (physical and psychological) arising from the environment, that sculpts its global knowledge space, its reasons and its actions. Hence, the 'arena of action' that we created for the robot to gain sensorimotor experience, the experiments from animal cognition that inspired its design, the kinds of physical forces (tools, magnetized sticks, traps, obstacles) and psychological forces (user goals, rewards, survival) that are in play, the kind of accidents that can take place in the environment (learning to exploit small magnetized sticks like Betty learnt to use flexible pipe cleaners), the kinds of experimental scenarios that can be presented to the robot, the kinds of behavior that we can possibly observe, the computation framework that needs to be created to trigger perception-reasoning-action generation in the GNOSYS robot and the visualization interfaces we created to better understand and analyze the complexity in the system, all these issues were dealt with in different sections of this chapter. In the next two chapters, we move from the 'arena of action' to the 'action generation' systems that drive the all movements, actions and interventions initiated by the GNOSYS robot in the playground.

Actions I – Computing with the Body

To explore the external ,
Create an object of desire, move towards it
and you will **Reach**
To explore the internal ,
Do not create an object of desire, Do not move
and you will **Reach**

VEDIC SCRIPTURES (around 12000 BC)

The action of 'reaching' is fundamental for any kind of goal directed interaction between the body and the world. The significance of the reaching action in any kind of cognitive behavior can be easily appreciated by considering the amazingly infinite contexts in which the word 'reach' is used in our daily language, for example, physical (reaching a cup of coffee), temporal (reaching a state in time), metaphorical (governments reaching people), metaphysical and several others. This incredible diversity in the application of the reaching action in different contexts also indicates how deeply meaning of this action is embodied in our subconscious minds.

Considering the fact that motor commands needed to perform even seemingly simple actions like reaching a cup of coffee also requires spatiotemporal coordination of several variables both internal and external to the body like the current state of the arm, orientation of the body, range of motion for the joints, range of torques for the actuators, geometry of the task (shape of the cup) etc, robotitians have often wondered about the nature of the computational/neural substrate in the brain that makes 'Reaching' possible in humans and animals. This stems from the fact that unlike the range of direct problems common in conventional physics that involve computing effects of forces on objects, brains have to deal exactly with the inverse problems of determining the motor commands that would permit the intended, goal directed mechanical interaction with the world. Strikingly, many of the inverse problems faced by the brain to control movements are indeed similar to the ones roboticians must solve to make their robots move intelligently in the world. Hence, while the field of neuroscience benefits from the theories of construction and control of man made bodies, robotitians on the other hand have the profound parallel opportunity to learn about the structural and functional organization of the central nervous system.

Despite extensive research on the computational basis of reaching movements, the explosive amount of mathematical formulae which emerge from the kinematic and dynamic equations of even a deceptively simple device like robotic arm does not serve much in either

intuitively understanding the arm mobility or developing computational frameworks for dexterous and task specific control of the movements of a limb, network of limbs (e.g. left arm–waist–right arm) or networks of external objects kinematically and dynamically coupled to the body/internal body model. In this chapter, we try to seek a simple intuitive explanation of how goals, constraints and choices can meet in a dynamical system to give rise to spatio-temporal coordination of the various degrees of freedom in two different robotic platforms 1) the GNOSYS robot that we introduced in the previous chapter and 2) the 53 degrees of freedom baby humanoid 'iCub'.

The computational mechanism proposed in this chapter is based on non-linear attractor dynamics where the attractor landscape is obtained by combining multiple force fields in different reference systems. The computational process of relaxation in the attractor landscape is similar to coordinating the movements of a puppet by means of attached strings, the strings in our case being the virtual force fields generated by the intended/attended goal and the other task dependent combinations of constraints involved in the execution of the task. From this comes the nickname PMP (Passive Motion Paradigm) given to the model. Along with the formal analysis, several simulation experiments on a 3 link planar arm, implementation results of the model on the GNOSYS robot and extensions of the computational architecture to control the upper body of the baby humanoid iCub are presented. Several insights on the use of the proposed framework as general forward/inverse models for both real and mental action generation in robotic bodies of any arbitrary complexity and redundancy, general principles involved in creating goal specific forward/inverse models composed of different kinematic chains of the body and upstream integration of the proposed computational framework with higher level cognitive layers like reasoning are presented as we gradually navigate through this chapter.

3.1 A brief history of 'Reaching'

The subjective ease with which we carry ourselves gracefully in constraint filled uncertain environments and effect action plans to perfection often masks the enormously complex integrative apparatus needed to spell synergy among the thousands of sensors, musculo-skeletal units and neurons that contribute to any act's planning and execution. Modeling the way in which humans coordinate their movements in daily life is an important scientific topic from many points of view, such as medical, psychological, kinesiological, and cybernetic. Tasks and goals are specified at a rather high, often symbolic level ("Stack 2 cylinders", "Fetch the red ball" etc.) but the motor system faces the daunting and under-specified task of eventually working out the problem at a much more detailed level in order to specify the activations which lead to joint rotations, movement trajectory in space, and interaction forces. At the same time, the solution must be compatible with a multitude of constraints: internal, external, task specific and their possible combinations. To further complicate

things, there is no single solution to the problem: in fact, there may be countless ways of doing the same task. Even the task of aimlessly moving the hand from one point to another in space can be executed with a number of possible hand trajectories, each trajectory could in turn be achieved in infinite ways using different combinations of joint motions at the shoulder, elbow and the wrist, and finally each joint motion can in turn be generated in infinite ways using different combinations of muscles to generate the net force at any joint. Hence arises the problem of tackling with kinematic redundancies and if possible, exploiting them effectively. While motor redundancy is advantageous because it enables a robot to avoid obstacles, joint limits, limb interference and attain more desirable postures, (for example when it is not sufficient to simply tap a target because a precise force vector must be applied to the touched object), from a control and learning point of view, however, redundancy also makes it quite complicated to find good movement plans that do not crash when it turns out that the designated target is unreachable or barely reachable. How do humans decide what to do with their extra joints, and how should humanoid robots control all their joints in order to generate coordinated movement patterns? Moreover, is the selection/coordination of redundant DoFs independent of the spatio-temporal organization of the reaching movements?

Tasks and goals are specified at a rather high, often symbolic level ("Stack 2 cylinders", "Fetch the red ball" etc.) but the motor system faces the daunting and under-specified task of eventually working out the problem at a much more detailed level in order to specify the activations which lead to joint rotations, movement trajectory in space, and interaction forces. At the same time, the solution must be compatible with a multitude of constraints: internal, external, task specific and their possible combinations. To further complicate things, there is no single solution to the problem: in fact, there may be countless ways of doing the same task.

Early studies of human arm trajectory formation (Morasso, 1981; Abend et al, 1982) showed invariant spatio-temporal features, such as a symmetric bell-shaped speed profile, which can be explained in terms of minimization of some measure of *smoothness*, such as jerk (Flash and Hogan, 1985) or torque-change (Uno et al, 1989). Later studies emphasized the importance of physical or computational force fields in the neural control of movement or motor learning (Mussa Ivaldi et al, 1988; Bizzi et al. 1991; Shadmehr and Mussa-Ivaldi 1994).

Most approaches to motion planning in robotics were derived from the early study of Whitney (1969) named RMRC (Resolved Motion Rate Control), which is based on the real-time inversion of the Jacobian matrix of the kinematic transformation, i.e. the function that links the variation of the joint angle vector dq to the pose dx of the end-effector. Clearly, for redundant kinematic chains RMRC must be modified by using one form or another of pseudo-inversion, as the Moore-Penrose matrix that provides a minimum norm solution for dq or other more general pseudo-inversion methods (Liegeosis, 1977) that can associate an

arbitrary cost function to the inversion calculation. Another method (Extended Jacobian Method: Baillieul 1985, Šoch and Lórencz 2005) extends the usual Jacobian matrix with additional rows that take into account virtual movements in the null space of the kinematic transformation: the extended Jacobian matrix is square and can be inverted in the usual way. In any case, the classical approaches to robot planning/control work well only inside the workspace and far away from kinematic singularities. Even if this is acceptable for an industrial robot, which has no or a limited number of excess DoFs and operates in a well defined and predictable environment, it is not feasible for a cognitive robot supposed to carry out, as humans, activities of daily life in a generally unknown environment.

Another class of robot planning methods is based on non-linear, attractor dynamics.The Passive Motion Paradigm (or PMP: Mussa Ivaldi et al, 1988) is a computational model that addresses the problem of coordinating redundant degrees of freedom by means of a dynamical system approach, similar to the Vector Integration to To Endpoint (VITE model: Bullock and Grossberg, 1988). In both cases there is a "difference vector" associated with an attractor dynamics that has a point attractor in the designated target. The difference is that the VITE model focuses on the neural signals commanding a pair of agonist-antagonist muscles, whereas the PMP model focuses, at the same time, on the trajectories in the extrinsic and intrinsic spaces. The model was extended later in order to include terminal attractor properties (Tsuji et al, 1995). In comparison with a recent computational model for reaching by Hersch and Billard (2008) that builds upon the VITE model, the proposed extension of the PMP model in this chapter is equally well a "multi-referential dynamical systems" for implementing reaching movements in complex, cognitive robots but does not require any explicit inversion and/or optimisation procedure.

Another approach to motion planning, based on non-linear dynamics, has been proposed by Ijspeert et al (2002) in order to form control policies for discrete movements, such as reaching. The basic idea is to learn attractor landscapes in phase space for canonical dynamical systems with well defined point attractor properties. The approach is very effective for movement imitation, because it approximates the attractor landscape by means of a piecewise-linear regression technique but somehow misses the multi-referential feature mentioned above. Also in the PMP model there is a well defined attractor landscape but it is derived from the composition of different virtual force fields that have a clear cognitive meaning and thus allow the seamless integration of planning with reasoning (Mohan and Morasso, 2007). Hence, we not only address the problem of planning the motion of highly redundant body, but also address the problem of allowing reasoning/cognitive processes to access/modulate efficiently such computation.

The computational process of relaxation in the attractor landscape of the PMP model is similar to coordinating the movements of a puppet by means of attached strings (figure 1), the strings in our case being the virtual force fields generated by the intended/attended goal and the other task dependent combinations of constraints involved in the execution of the

task. The computational mechanism involves a process of passive simulation of movement as if it was imposed by an external agent, in such a way to distribute the desired motion to the global kinematic structure by recruiting joints, actuators, and tools while pulling the dynamical system to the equilibrium state. When motor commands obtained by this process of virtual relaxation is actively fed to the robot, the robot will reproduce the same motion. The units of this network operate in different spaces (end-effector space, joint space, actuator space, tool space) and locally compute their own reaction to the "source" of planned motion based on their local virtual impedance/admittance. Hence, the computational model presented in this chapter has the flavour of local to global computing principles typical of connectionist models of associative memory like the Hopfield networks (Hopfield 1982; Sejnowski et al 1988) that represent patterns by means of configurations of minimal potential energy. No matrix inversion is necessary and the computational mechanism does not crash near kinematic singularities or when the robot is asked to achieve

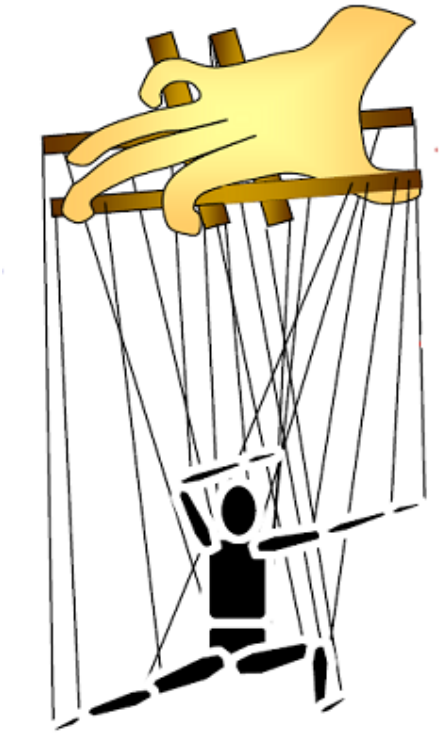


Figure 1. The "marionette" metaphor of the Passive Motion Paradigm (PMP). The "internal model" that coordinates and plans the motion of all the joints operates on a small set of force fields applied to "focal points" of the body model.

a final pose that is outside its intrinsic workspace: what happens, in this case, is the gentle degradation of performance that characterizes humans in the same situations. Moreover, the remaining error at equilibrium is a valuable information for triggering a higher level of reasoning, such as searching for an alternative plan or making/using an environmental object as a tool. The control of the timing of the relaxation process using the notion of terminal attractor dynamics further endows the generated trajectories with human-like smoothness and is crucial for complex motion patterns such as bimanual coordination, interference avoidance and precise control of the reaching time.

> The computational process of relaxation in the attractor landscape of the proposed framework is similar to coordinating the movements of a puppet by means of attached strings (figure 1), the strings in our case being the virtual force fields generated by the intended/attended goal and the other task dependent combinations of constraints involved in the execution of a task.

In contrast to optimal control based approaches (Shadmehr & Krakauer, 2008), the relaxation process associated with the proposed computational model does not look for optimal activation patterns based on a explicit/global criterion of optimality but rather focuses on a divide and rule strategy that focuses on formation of motor synergies so as to simultaneously incorporate multiple task-dependent constraints, on one hand, and destroy as many degrees of freedom as possible (in the Bernsteinian sense), on the other. The same relaxation process can dynamically coordinate the movements of a limb, network of limbs (e.g. left arm–waist–right arm) or networks of external objects kinematically and dynamically coupled to the internal body model (e.g. right arm-tool-left arm, as in driving a car or transporting objects using two arms).

3.2 Real/Mental Action Generation: From physical force fields to computational force fields

The central theme behind the formulation of the forward inverse models described in this chapter is the observation that motor commands for any kind of motor action, for any configuration of limbs and for any degree of redundancy can be obtained by an “internal simulation” of a “passive motion” induced by a “virtual force field” (Mussa Ivaldi et al, 1988) applied to a small number of task-relevant parts of the body. Here “internal simulation” identifies the relaxation to equilibrium of an internal model of limb (arm, leg etc, according to the specific task); “passive motion” means that the joint rotation patterns are not specifically computed in order to accomplish a goal but are the indirect consequence of the interaction between the internal model of the limb and the force field generated by the target, i.e. the intended/attended goal. It is important to note that, the force fields we are considering don’t really describe the biomechanical forces at play during the execution of

movements, but are just computational metaphors that describe the dynamics of the internal computational engine (a fact reflected in the title of this section).

In this sense, functioning the computational model is analogous to the mechanism of coordinating the motion of a wooden marionette by means of attached strings as discussed in the previous section. What makes it attractive from the computational point of view is its simplicity and robustness. It is simple because the planner/controller does not have to be concerned with all the degrees of freedom (DOF) at the same time but only has to deal with a smaller number of "end-effectors" that have to be eventually pulled by the virtual force fields (or simply strings in the puppet metaphor) generated by the goal. It is robust because no model inversion is necessary and the relaxation to equilibrium in a conservative force field can never crash. In simple terms, as the end-effector (hands, legs, beak etc) is pulled towards the goal, the rest of the body 'elastically' reconfigures itself to new posture that is necessary to position the end-effector at the goal. Moreover, when motor commands obtained by this process of virtual relaxation is actively fed to the robot, the robot will reproduce the same motion. Hence, a key feature of the architecture is that the same computational model is used to support mental simulations related to reaching movements employed by a higher level reasoning process and the actual delivery of motor commands during movement execution. The difference is that, in the latter case, the motor outflow interacts with the peripheral circuitry of the motor system (spinal circuitry and mechanical properties of the neuromuscular apparatus in the biological domain or its simplified equivalent in the robotic domain).

The relaxation mechanism does not require any cost function to be pre-specified in order to solve the indeterminacy related to the excess DOF's (the redundancy problem) and allows to integrate a range of internal and external constraints at runtime based on the requirements of the task that needs to be performed. For example, to grasp a stick that is placed horizontally on a table, the constraints may be to orient the wrist and gripper appropriately, keep all joint angles in the mid range with respect to the body and so on. In this sense, the internal model associated with the force-field represents the set of all geometrically possible solutions (in the form of a holographic memory) out of which one is implicitly selected as a function of the structural and task-specific constraints. The next three sections we describe the general computational framework, extensions to the forward inverse model so as to deal with multiple task specific constraints (hence dealing with and exploiting motor redundancy) and finally the problem of temporal coordination of different relaxations i.e. being in sync.

3.2.1 Relaxation of an internal model to equilibrium: The computational framework

As shown in figure 2, the basic structure of the forward inverse models is composed of a fully connected network of nodes either representing forces or representing flows (displacements) in different motor spaces (end-effector space, joint space, muscle space,

tool space etc). For simplicity we just consider the end-effector and joint spaces for discussion in this section.

We also do not consider issues related to temporal synchrony , force fields arising external and internal constraints for the time being. A reader may think of them being additional strings that are pulling the puppet (the internal body model) with different amounts of forces (representing the significance of the constraint with respect to the task being coordinated by the dynamics).

We also observe that a displacement and force node belonging each motor space is grouped as a work (force. displacement) unit (WU). There are only two kinds of connections 1) between a force and displacement node belonging to WU that describes the elastic causality of the coordinated system and 2) between two different motor spaces that describes the geometric causality of the coordinated system.

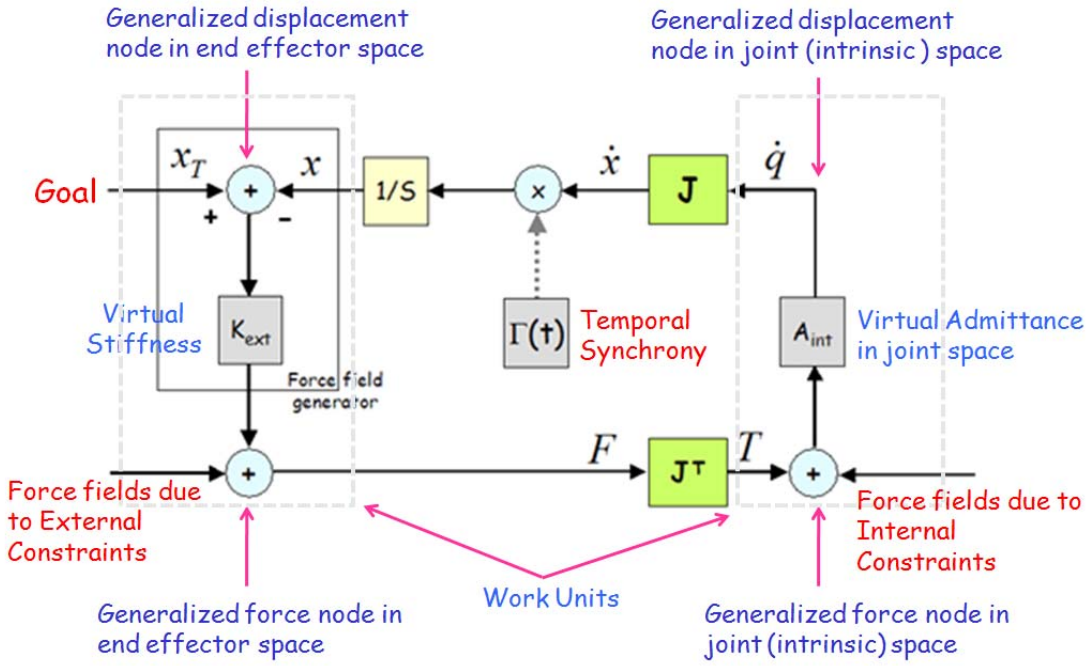


Figure 2. Basic computational scheme of the PMP for a simple kinematic chain. x is the position/orientation of the end-effector, expressed in the extrinsic space; x_T is the corresponding target; q is the vector of joint angles in the intrinsic space; J is the Jacobian matrix of the kinematic transformation $x = f(q)$; K_{ext} is a virtual stiffness that determines the shape of the attractive force field to the target; "external constraints" are expressed as force fields in the extrinsic space $F = F(x, \dot{x}, \ddot{x})$); "internal constraints" are expressed as force fields in the intrinsic space ($T = T(q, \dot{q}, \ddot{q})$); A_{int} is a virtual admittance that distributes the relaxation motion to equilibrium to the different joints; $\Gamma(t)$ is the time-varying gain that implements the terminal attractor dynamics.

Let x be the vector that identifies the pose of the end-effector of a robot in the extrinsic workspace and q the vector that identifies the configuration of the robot in the intrinsic

joint space: $x = f(q)$ is the kinematic transformation that can be expressed, for each time instant, as follows: $\dot{x} = J(q) \cdot \dot{q}$ where $J(q)$ is the Jacobian matrix of the transformation. The motor planner/controller, which expresses in computational terms the passive motion paradigm, is defined by the following steps (see fig. 2):

1) Associate to the designated target x_T a conservative, attractive force field in the extrinsic space

$$F = K_{ext}(x_T - x) \quad (1)$$

where K_{ext} is the virtual impedance matrix in the extrinsic space. The intensity of this force decreases monotonically as the end-effector approaches the target.

2) Map the force field into an equivalent torque field in the intrinsic space, according to the principle of virtual works:

$$T = J^T F \quad (2)$$

Also the intensity of this torque vector decreases as the end-effector approaches the target.

3) Relax the arm configuration in the applied field:

$$\dot{q} = A_{\text{int}} \cdot T \quad (3)$$

where A_{int} is the virtual admittance matrix in the intrinsic space: its modulation does not affect the trajectory of the end-effector but modifies the relative contributions of the different joints to the reaching movement.

4) Map the arm movement into the extrinsic workspace:

$$\dot{x} = J \cdot \dot{q} \quad (4)$$

5) Integrate over time until equilibrium

$$x(t) = \int_{t_O}^{t} J\dot{q}\, d\tau \quad (5)$$

The algorithm always converges to an equilibrium state which is reached asymptotically in the following conditions:

(a) When the end-effector reaches the target, thus reducing to 0 the force field in the extrinsic space (eq. 1);

(b) When the force field in the intrinsic space becomes zero (eq. 2) and this can happen in the neighborhood of kinematic singularities;

Case (a) is the condition of success termination. But also in case (b), in which the target cannot be reached for example because it is outside the workspace, the final configuration

has a functional meaning for the motion planner because it encodes geometric information valuable for re-planning .

Thus the basic PMP is a robust non-linear dynamic approach to the solution of the inverse kinematic problem that does not require any explicit inversion or optimization task. Redundancy is dealt with by the admittance matrix of the kinematic chain. For example, “freezing” or “unfreezing” a joint can be implemented in a simple way by manipulating accordingly the relevant elements of the matrix: moreover, this modulation can be carried out efficiently in real-time, in a task-dependent way.

The force field described by equation 1 can be isotropic or anisotropic according to the fact that the eigenvalues of matrix K_{ext} are equal or unequal. The flowlines in the former case are straight lines whereas are curved in the latter case. More complex curved trajectories can be obtained by adding a rotational component to the convergent force field given by equation 1. The relative values of the elements of the joint admittance matrix A_{int} measure the degree of involvement of each DoF in a given reaching movement. For example, a joint rotation can be “frozen” by setting to zero the corresponding admittance value; in any case, the distribution of admittance values can be task-dependent thus allowing to exploit redundancy in a goal-oriented way.

Prior to adapting the forward inverse models to suit the GNOSYS robotic environment, we performed initial simulation experiments with respect to a planar robot with three revolute joints and link lengths of one for each link. Figure 3 illustrates the reaching trajectories obtained by the relaxation mechanism shown in figure 1 and expressed by equations (2-5) , starting from an initial configuration of (π/4, π/4, 0) and without application of any joint fields (representing internal constraints) and the external force drive (for Null Space movements). Of the four different targets shown in figure 3, three are within the reachable workspace and one (Target 4) is not reachable. Note that in case the desired target is outside the workspace, the manipulator nevertheless tries to follow the desired coordinates as well as possible and fully extends the arm to a position that is at a minimum distance from the target. In this case, the manipulator also finally points in the direction of the desired endpoint coordinates. Hence even if there is no solution to the problem, the relaxation process always converges to the best possible solution under the influence of all constraints that exist at that point of time. Also note the relatively straight line motion of the end-effector (with a bell shaped velocity profile) which agrees with experimental evidence of kinematic invariance’s in hand reaching movements of humans().

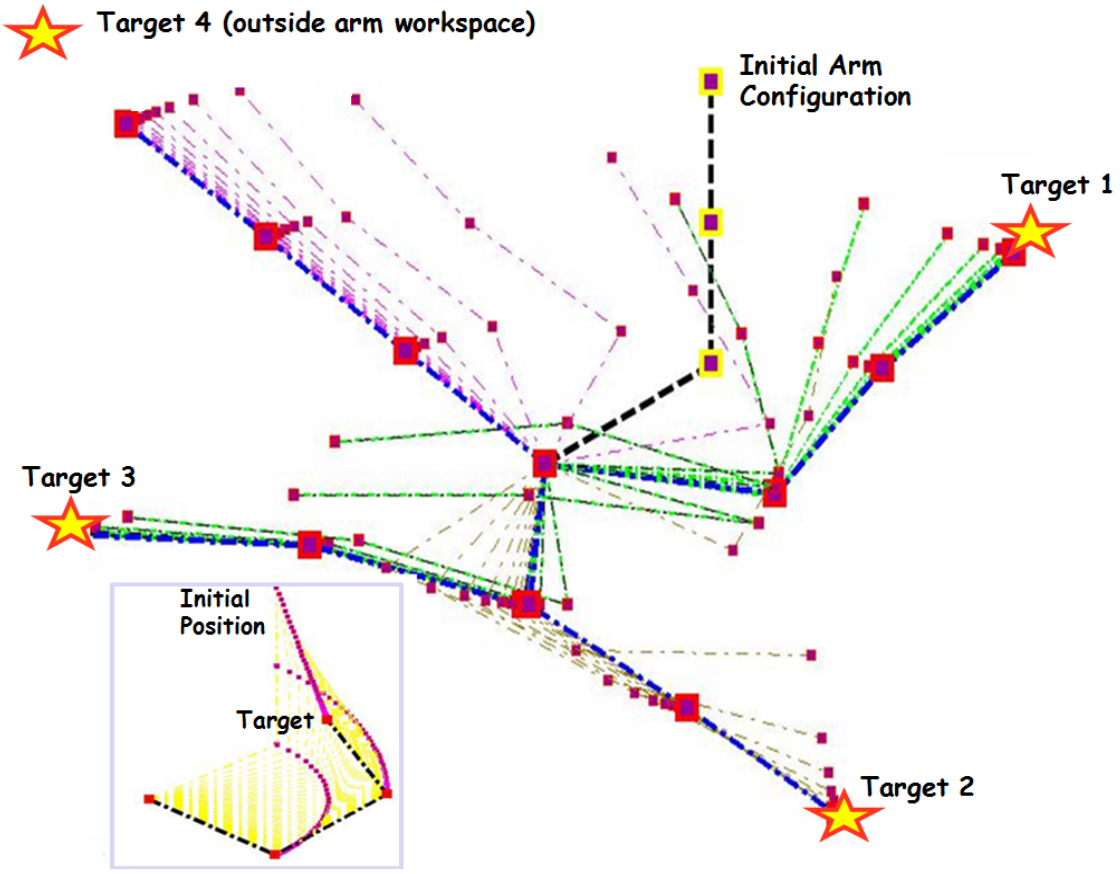


Figure 3. Reaching trajectories in a three link planar arm obtained using the relaxation mechanism shown in figure 2

Neural Network for distributed representation of the Jacobian's (horizontal weights J and J^T) in the Forward Inverse model

If a precise analytic expression of the kinematic transformation $x = f(q)$ is available, then the Jacobians can be computed in a direct way: $J = \partial f / \partial q$. In many cases, however, this is not possible or at least the expression of the kinematic transformation is not reliable due to large uncertainties in the geometric parameters of the robot (especially in the case of humanoids we deal with in section 3.6). Still we can obtain experimentally a "training set" of joint rotation readings with the corresponding coordinates of the end-effector. In this case, we can recover J from samples of the kinematic transformation. This purpose can be achieved by training a standard multi-layer network (ANN) with the back-propagation technique and the training set mentioned above. For example, we can use a three-layer network as shown in figure 4, where $\{q_i\}$ is the input array (joint angles), $\{x_k\}$ is the output array (position/orientation of the end-effector), and $\{z_j\}$ is the output of the hidden units.

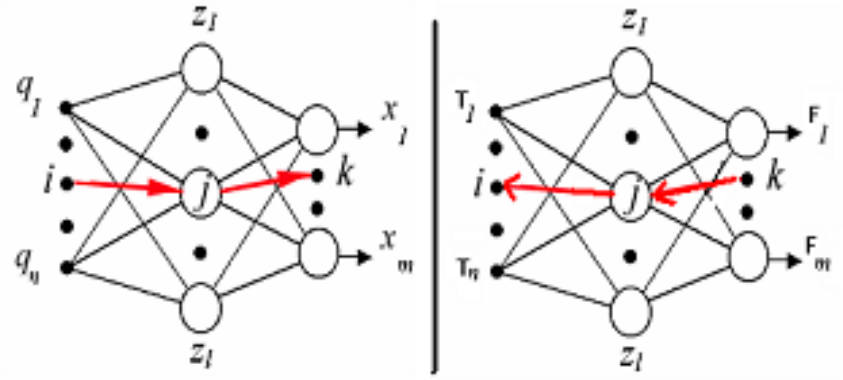


Figure 4. Feedforward neural network, trained (off-line) to approximate the kinematic transformation and used for evaluating (on-line) the corresponding Jacobian matrix.

The following are the equations of the above ANN:

$$x = f(q) \Rightarrow \begin{cases} h_j = \sum_i w_{ij} q_i \\ z_j = g(h_j) \\ x_k = \sum_j w_{jk} z_j \end{cases} \tag{6}$$

where $\{w_{ij}\}$ are the connection weight from the input to the hidden layer, $\{w_{jk}\}$ are the connection weights from the hidden to the output layer, $\{h_j\}$ are the net inputs to the neurons of the hidden layer. The neurons of the hidden layer are characterized by the logistic transfer function $g(h)$; the output layer is composed of linear neurons. After training, we can extract the Jacobian matrix from the neural network by applying the chain rule in the following way:

$$J_{ki} = \frac{\partial x_k}{\partial q_i} = \sum_j \frac{\partial x_k}{\partial z_j} \frac{\partial z_j}{\partial h_j} \frac{\partial h_j}{\partial q_i} = \sum_j w_{jk} g'(h_j) w_{ij} \tag{7}$$

Eq. 7 can be easily adapted to ANNs with more than 3 layers. At run-time, the ANN must be fed with the flow of $\{q_i(t)\}$ values in order to recover the corresponding $\{h_j(t)\}$ values. Thus it is possible to carry out the computations indicated in figure 2 by means of the two following equations:

$$\dot{x} = J \cdot \dot{q} \Rightarrow \dot{x}_k = \sum_i J_{ki} \dot{q}_i = \sum_i \left(\sum_j w_{jk} \cdot w_{ij} \cdot g'(h_j) \right) \dot{q}_i \tag{8}$$

$$T = J^T \cdot F \Rightarrow T_i = \sum_k J_{ik} F_k = \sum_k \left(\sum_j w_{ji} \cdot w_{kj} \cdot g'(h_j) \right) F_k \tag{9}$$

The block diagram of the resulting computational chain is shown in figure 5.

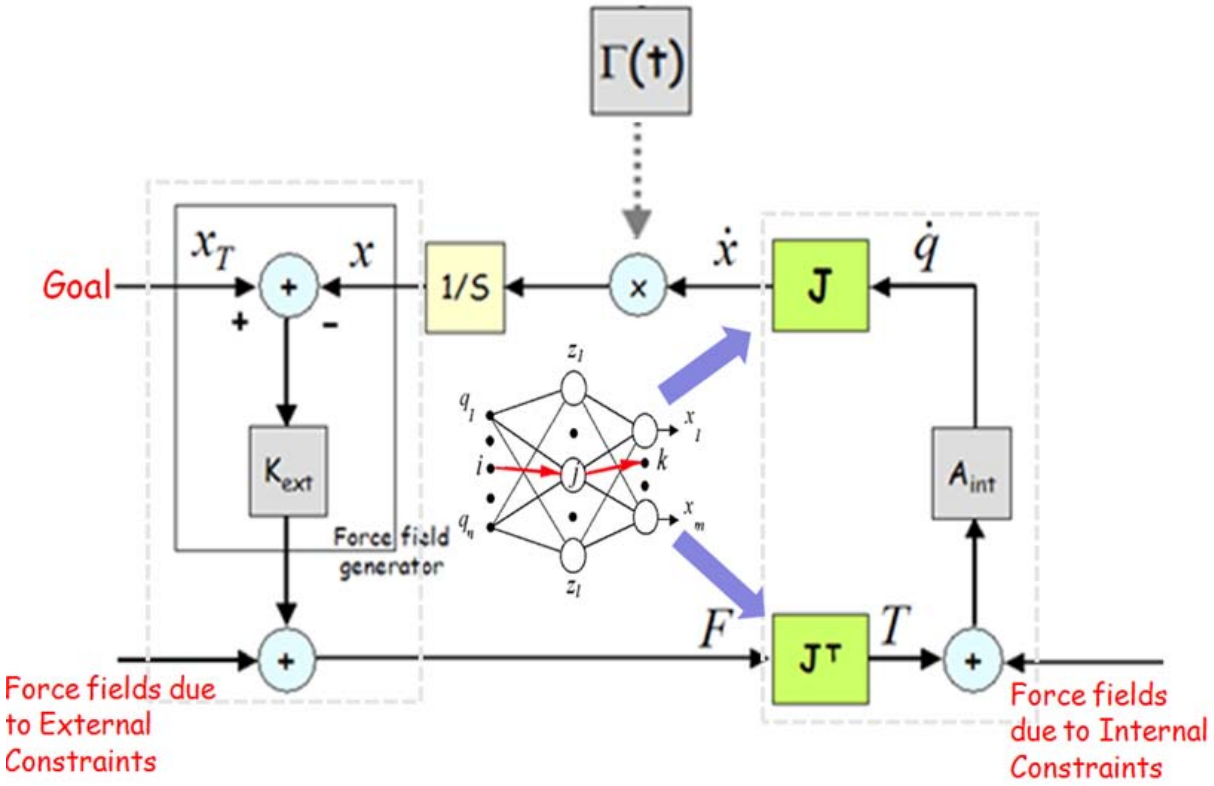


Figure 5: Forward/Inverse model pair with distributed representation of the Jacobian's using a neural network

3.3 Relaxation in finite time: The terminal attractor dynamics

The basic PMP model is an asympotically stable dynamical system with a point attractor that brings the end-effector to the target if the target is indeed reachable. However, asympotic stability implies that the equilibrium configuration is reached after an infinite time and does not provide any mechanism to control the speed of approach to equilibrium.

A way to explicitly control the time, without using a clock, is to insert in the non-linear dynamics of the PMP model a suitable time-varying gain $\Gamma(t)$ that grows monotonically as x approaches the equilibrium state and diverges to an infinite value in that state. The technique was originally proposed by Zak (1988) for speeding up the access to addressable memory in neural networks and then was applied to a number of problems in neural networks. Our purpose, however, is not merely to speed up the operation time of the planner but to allow a control of the reaching time as well as the speed profile in order to fit the human reaching patterns and to allow synchronization of complex tasks as in bimanual coordination. In particular, we propose to extend the basic PMP model (figure 5) by inserting the time-varying gain Γ(t), as a further development of what was proposed by Tsuji et al (1995) and Morasso et al (1997):

$$\begin{cases} \Gamma(t) = \dot{\xi} / (1-\xi) \\ \xi(t) = 6 \cdot (t/\tau)^5 - 15(t/\tau)^4 + 10(t/\tau)^3 \end{cases} \quad (10)$$

where $\xi(t)$ is a time-base generator (TBG): a scalar function that smoothly evolves from 0 to 1 with a prescribed duration τ and a symmetric bell-shaped speed profile (figure 6). A simple choice for the TBG is the minimum jerk polynomial function of equation 10, but other types of TBGs are also applicable without any loss of generality.

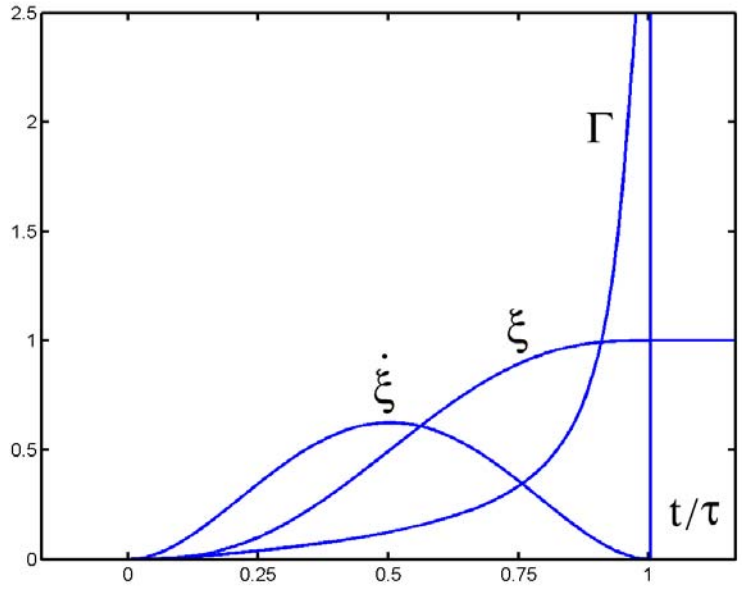


Figure 6. Time-base generator for terminal attractor dynamics - $\Gamma(t)$ - obtained from a minimum jerk time function - $\xi(t)$ - with assigned duration τ.

In summary, this extension of the basic PMP model in order to allow terminal attractor dynamics simply requires that equation 4 is substituted by the following one:

$$\dot{x} = \Gamma(t) \cdot J \cdot \dot{q} \quad (4a)$$

In order to demonstrate that in this way the target is reached after a time equal to τ and with an approximately bell-shaped speed profile, we can substitute the vector equation 4a with an equivalent scalar equation in the variable *z* defined as the running distance from the target along the trajectory generated by the PMP network ($z = 0$ for $x = x_T$): $\dot{z} = \Gamma(t)\, f(z)$, where $f(z)$ is, by construction, a monotonically increasing function of z which passes through the origin because $x = x_T$ is the point attractor of the dynamical PMP model. Therefore, for $f(z)$ we can formulate the following bound $\gamma_{\min} z < f(z) < \gamma_{\max} z$, where $\gamma_{\min}$, $\gamma_{\max}$ are two positive constants. By denoting with γ any of them, we can write the following equation:

$$\frac{dz}{dt} = -\frac{d\xi / dt}{1-\xi}\gamma z \tag{11}$$

from which we can eliminate time

$$\frac{dz}{d\xi} = -\frac{\gamma z}{1-\xi} \tag{12}$$

The solution of this equation is then given by:

$$z(t) = z_0(1-\xi)^{\gamma} \tag{13}$$

where z_0 is the initial distance from the target along the trajectory. This means that, as the TBG variable $\xi(t)$ approaches 1, the distance of the end-effector from the target goes down to 0, i.e. the end-effector reaches the target exactly at time $t = \tau$ after movement initiation. This applies to both limits of the bound

$$z_0(1-\xi(t))^{\gamma_{\min}} < z(t) < z_0(1-\xi(t))^{\gamma_{\max}} \tag{14}$$

In any case the terminal attractor $z = 0$ is reached at $t = \tau$. The speed profile may be somehow distorted in relation with a symmetric bell shape (fig. 7) but the terminal attractor property of the model is maintained for a wide range of values of γ.

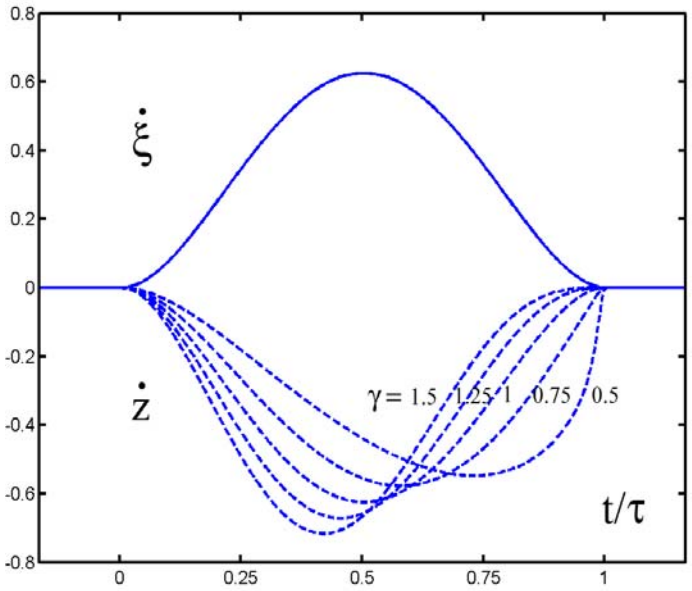


Figure 7. The property of reaching the target at time $t = \tau$ is maintained in any case, although the speed profile of the curvilinear variable $z(t)$ is somehow distorted, with respect to the bell-shaped profile of the TBG variable $\xi(t)$, when γ has values different from the ideal value $\gamma = 1$.

Figure 8 shows the trajectories generated in the end-effector space (x, y) and proximal space (q1, q2, q3) of the three link planar arm considered in figure 3, during relaxation to a target (x=0,y=2) staring from initial condition (x=2,y=1), under the influence of the terminal attractor relaxation. The time base generator signal 'gamma' is also shown in the figure . In this example, timing of relaxation process was set to 4, i.e. the end-effector should converge

to its fixed point at time=4 units. It is evident from figure 8 that the end-effector smoothly converges to the target in the specified time. Further, as the end-effector reaches its equilibrium configuration, all joint angles also converge to their final values as a result of the propagation of the force fields from end-effector space to joint space.

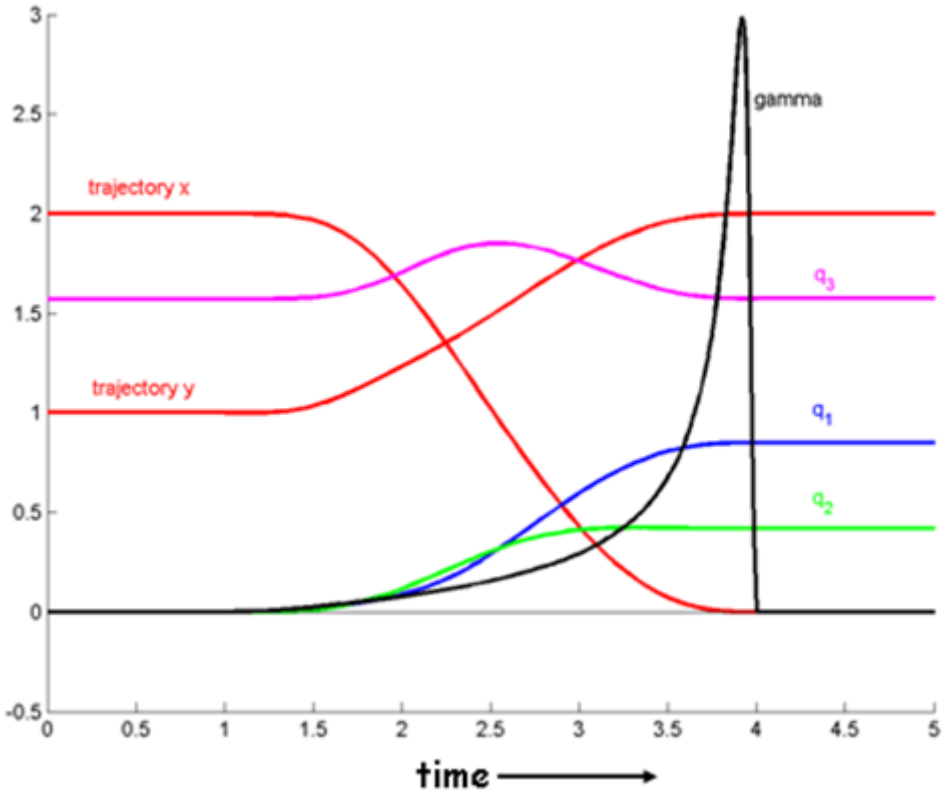


Figure 8. Trajectories in Proximal space (q1, q2, q3), distal space (x, y) and the time base generator function Gamma for convergence time t_f=4, while reaching a final target (x=0,y=2).

3.4 Relaxation under the influence of 'multiple' task specific constraints

Motor commands needed to perform even seemingly simple actions like lifting a cup of coffee are also dependent on several variables both internal and external to the body like the current state of the arm (joint angles) and orientation of the body, range of motion for the joints, range of torques for the actuators, geometry of the task (shape of the cup), obstacles in the environment to list a few. We effortlessly modify the orientation of our bodies / arms based on the the nature of the physical objects we interact with or based on the nature of the task being performed. No wonder, proper orientation always simplifies the required effort as well as increases the chances of successful execution of a task.

The most natural way to deal with constraints is to exploit redundancy. Most robotic manipulators existing today are inherently redundant in nature i.e. they possess extra degrees of freedom than that is needed to execute a task. While redundancy is advantageous because it improves flexibility in manipulation, allows the robot to avoid

obstacles, joint limits and attain more desirable postures, it also makes it extremely complicated from the computational point of view to find good movement plans. This is because of the fact that the inverse kinematic mapping for the case of redundant manipulators is ill posed (many to one) and requires regularization mechanisms to spell out acceptable solutions. This problem can be solved using conventional techniques by pre-specifying additional constraints (cost functions) that must be satisfied in addition to the solution (Ex: avoid singular configurations, minimizing torques, Minimizing Kinetic energy etc). One possibility while using ANN's is to imbed the constraint into the training set so that the neural network will learn to output those solutions which also satisfy some optimality criteria. But these approaches are rigid and do not allow any task dependent adaptation. Another solution in this context is to break the training into two phases: one which forms the global mapping and other which optimizes some criteria. In this way it is possible to have inverse models tuned to specific task needs (Rumelhart and Jordan, 1992). But this still does not allow any runtime task specific adaptation.

In the previous section, we described a forward / inverse model architecture that relaxes a kinematic chain representing task relevant parts of the body (body model) to a virtual force field generated by an active goal (and applied at the end-effector). We also saw how timing of the relaxation process can be precisely controlled using the concept of terminal attractors. In this section, we further extend the computational framework and explain how custom relaxation networks can be built at run time to simultaneously take into account a range of constraints dynamically specified by the nature of the task being performed. In simple terms, redundancy is taken into account not in terms of optimisation of a cost function but in terms of task-dependent mixtures of multiple constraints, expressed as additive force fields in both operational spaces. In the following example, we demonstrate this flexibility in the computational model by illustrating the simplicity with which the relaxation mechanism can be extended to simultaneously take into account three major classes of constraints a) Internal b) external and c) relaxation in the null space for producing the required interaction force (when in addition to reaching a target, a precise force vector must be applied to the touched object like in the case of pushing, tapping etc).

A. Internal constraints

Typical internal constraints are joint limits: $\left\{ q_i^{\min} < q_i < q_i^{\max},\ i = 1, n \right\}$. They can be implemented by means of an elastic force field in the joint space, with a nominal equilibrium for example, in the middle of the range of motion of each joint. Figure 9 shows the resulting composite relaxation network for reaching a goal target and additionally reaching an approximate posture in the joint space such that the joint rotations are well within the possible limits of motion. We must also note that range of motion for the joints need not be pre specified and are themselves a function of the task at hand. For example, in case of

minor injury to some joint in our arm (a sprain or a fracture) we are still able to execute a range of actions with the arm. The only difference is that the injured joint now becomes more stiff (which is represented by the K_{int} parameter) and hence contributes less to the overall solution. An alternative implementation is a repulsive field that steeply grows in the vicinity of the joint limits. In both cases, this force field in the joint space is added to the previous force field generated by the goal (as in figure 2) that attracts the end-effector to the target. While the latter force field allows the target to be reached, whereas the former field induces motions in the null space of the kinematic transformation that selects joint configuration patterns compatible with the target but as far as possible from the joint limits.

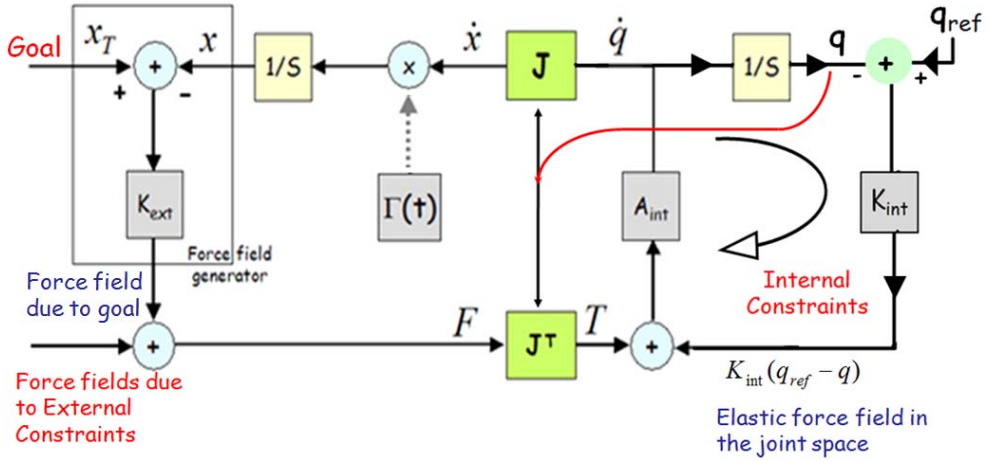


Figure 9. Forward/Inverse model pair after addition of an elastic force field in the joint space in order to relax an internal model of the arm to a target, taking into account joint related constraints

B. Relaxation in the null space

There are several situations where in addition to reaching a target with the end-effector, a precise force vector must be applied on the manipulated object (like in the case of pushing, tapping etc). After reaching the target pose, it is possible to further perform movements in the null space of the kinematic transformation[2] in such a way that for a given desired effort F_T the torque required of each actuator is within the allowed limits. This can be implemented by introducing F_T as an additional force drive in the inverse model (as shown in figure 10) and then saturating the computed torque vector according to the actuator constraints. The notion of optimality, in this case, can be related to the torque limits that characterize each actuator: an optimal arm configuration, from the point of view of the actuators, corresponds to a required torque output for each actuator that is as far as possible from the torque limit. This involves a kind of search in the null space of the kinematic transformation because, for a given force vector delivered at the end-effector, the actuator torques depend on the arm configuration via the transpose Jacobian. The solution

[2] The null space of the kinematic transformation $x = f(q)$ is characterized by the equation: $J(q) \cdot \dot{q} = 0.$

is given by a simulation of the model, after x has converged to x_T, with the target force F_T applied as an additional input in the extrinsic space.

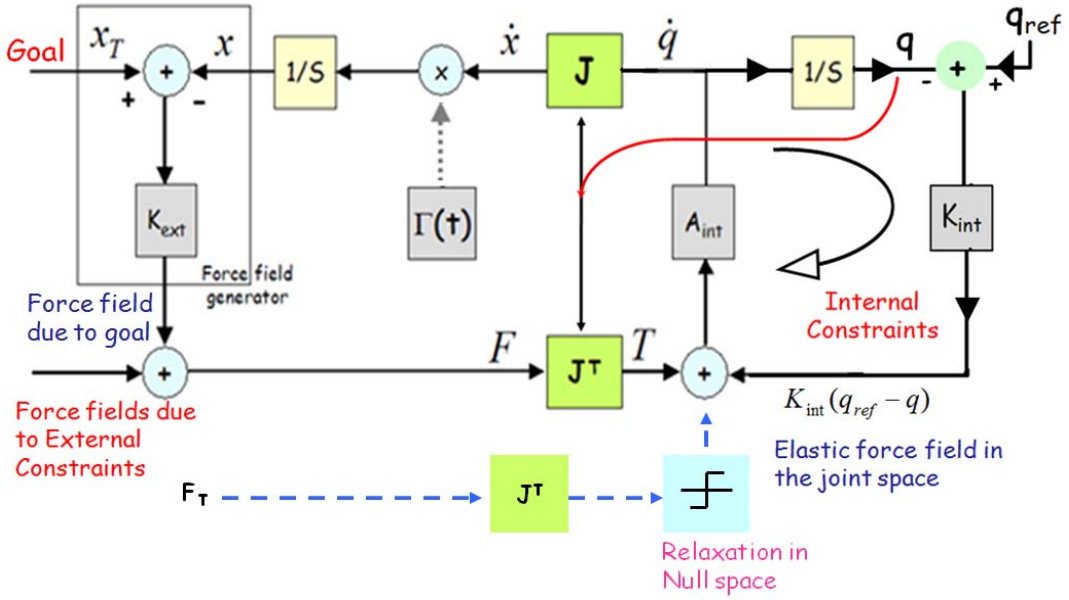


Figure 10. Composite forward/inverse model for executing/mentally simulating a reaching action to a target object taking into account multiple constraints like 1) the internal joints related constraints, 2) the timing of the movement and 3) the interaction force that needs to be applied on the manipulated object.

If the virtual stiffness is sufficiently high, such target force will determine a motion in the null space without displacing too much the position of the end-effector. The solution found in this way can be induced to comply with the torque limits by introducing saturation elements of the different actuator torques just after the transpose Jacobian. Note that null space movements are only possible for a redundant system and the effort constraint Ft is turned on only after target X_t has been reached.

C. Wrist orientation (for example, while grasping sticks placed on the table)

The distributed nature of the forward inverse model allows us to easily incorporate the process of relaxation at a target position with a desired orientation of the wrist. Orienting the wrist appropriately is critical when the robot tries to pick up sticks (and other objects) placed at arbitrary orientations on the table, for further use as tools to attain goals.Further, the effective extention of reaching space that the robot could get using a tool is a function of the orientation with which it is grasped. The resulting computational network is shown in figure 11. The new addition to the scheme is the inner loop that generates a new force field F_2 that defines an attractor applied to the wrist. In this case there are three weighted, superimposed force fields that shape the spatio temporal behavior of the system.

1. To the end-effector (to reach the target);
2. To the wrist (for orientation);
3. A force field in joint space as internal constraints of Joint limits.

The same TBG coordinates all the three relaxation processes.

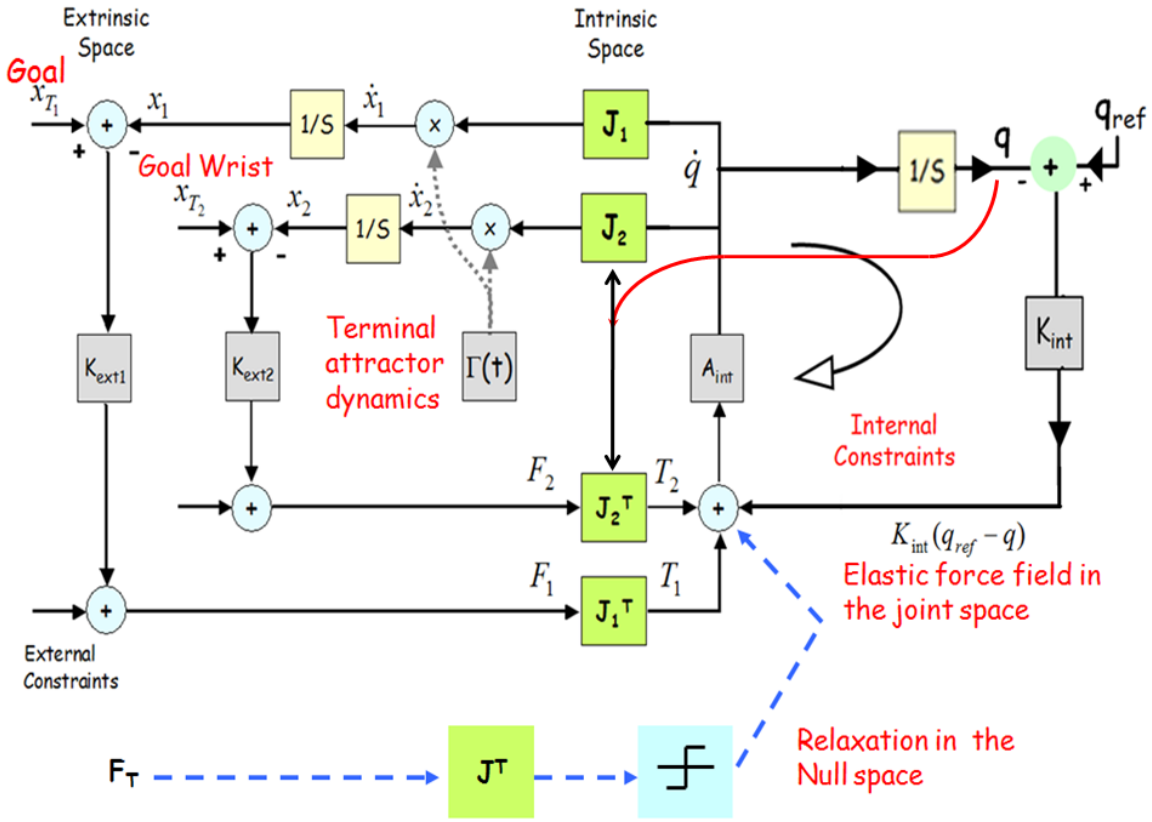


Figure 11. Composite forward/inverse model with two attractive force fields applied to the arm, a field F_1 that identifies the desired position of the hand/fingertip and a field F_2 that helps achieving a desired pose of the hand via an attractor applied to the wrist. This network assumes that the shoulder of the robot is "grounded": this means that the two force fields do not propagate beyond the shoulder joint. Force fields representing other constraints like joint limits and net effort to be applied (scaled appropriately based on their relevance to the task) are also superimposed on the earlier fields F1 and F2. The time base generator takes care of the temporal aspects of the relaxation of the system to equilibrium. In this way, superimposed force fields representing the goals and task relevant mixtures of constraints can pull a network of task relevant parts of an internal model of the body to equilibrium in the mental space.

Provided a stick (or any other object) is reached and grasped at a particular orientation say q_x using the scheme of figure 11, reaching an otherwise unreachable target using the grasped tool further involves updating the geometry of the arm taking into account the presence of stick in the gripper. This can be achieved supplementing the forward/inverse model pair by providing the adjustment terms emerging in the kinematics if reaching has to

be done with a tool. We must note that this process can be made independent of the length of the tool itself, that merely acts as a variable gain. In addition, orientating tools appropriately while reaching objects to be manipulated with them is a fundamental factor in achieving success with the tool (Even a stick can be used in several ways based on its orientation for example, pushing, pulling, digging, poking, fighting). Accounting for tool orientation is in fact, just a special case of the computational scheme in figure 11, with the only difference being that the relaxation process applied at the end-effector is now applied at the tool and the relaxation process applied at the wrist is replaced is now applied at the end-effector. This additional ability to orient sticks appropriately is extremely useful in a number of tasks critical for generation of flexible motor actions in complex experimental scenarios that require the robot to push objects, magnetically couple 2 sticks (especially if the second stick is placed horizontally on a table) etc. Before implementing these ideas on the robot, we initially validated all these extensions to the forward inverse model to perform a range of tasks using a 3 link planar arm simulation. Some simulation results of the solutions generated by the computational model for reaching task taking into account different combinations of task dependent constraints is shown in figure 12.

In summary, it is easily possible to extend the basic scheme shown in figure 2 and develop custom forward/inverse models in a well defined and task specific manner such that a range of runtime factors like available motor redundancy, internal constraints (as regards to task geometry, orientation, self-interference, and range of forces/torques) and external constraints (obstacles, efforts) are combined together based on their relevance to the active goal. The resulting field structure then shapes the overall dynamics of the system. Moreover, in case the forward simulation is successful then the movement is executed on the robot, otherwise the residual "error" or measure of inconsistency can be taken as a starting point for breaking the action plan into a sequence of sub actions i.e. ‘trigger higher levels of reasoning’.

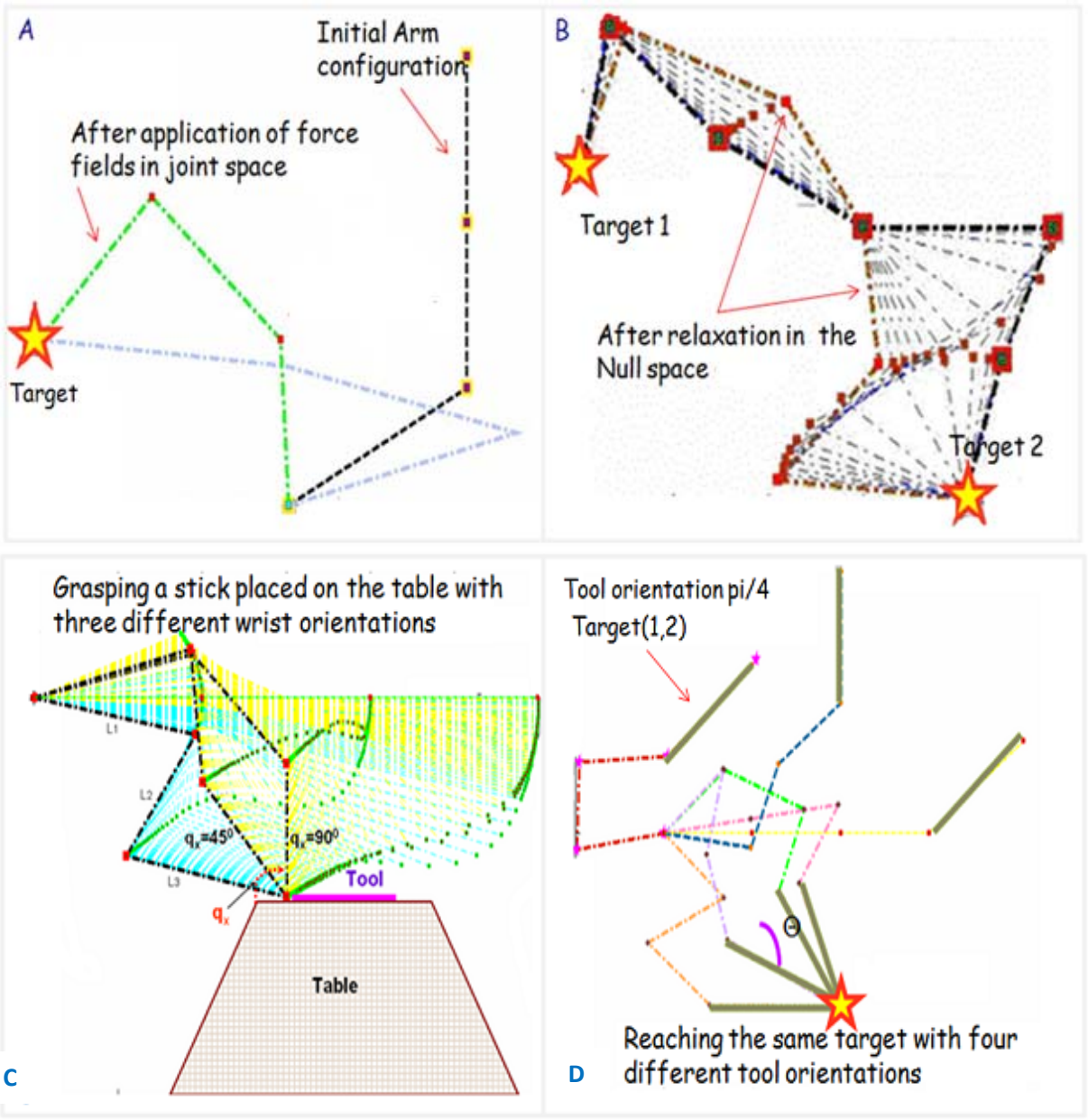


Figure 12. Panel A shows relaxation to target pose before and after application of attractive force field in joint space (computational models in figure and figure respectively). Panel B shows virtual movements in the null-space (using the scheme of figure) when a force drive Ft representing effort related constraints is turned on after target pose has been reached.Panel C shows relaxation to a target (a stick placed on a table) with different wrist orientations (using the composite forward/inverse model shown in figure 10). Panel D shows several examples of reaching otherwise unreachable targets using a gripped tool and further reaching them with specified tool orientations.

3.5 Action Generation System in "Action" on the GNOSYS Robot

In the previous section we introduced a forward/inverse motor control architecture that provides a general solution for mentally simulating reaching action to a target object taking

into consideration a range of geometric constraints (range of motion in the joint space, internal and external constraints in the workspace, orientation of wrists, tools etc) as well as effort-related constraints (range of torque of the actuators, etc.). We also presented a set of examples on how composite relaxation networks can be built in run time based on the nature of the task being realized. In this way, we not only get rid of preprogrammed cost functions, but also obtain a framework where goals, constraints and choices (i.e. arising out of redundancy in the body structure) can meet in a well defined manner to generate flexible actions. We also presented a set of simulation results using a simple three link planar arm to demonstrate the behavior of the dynamical system. It is at this point we make the transition from the simulation environment to the GNOSYS robot. Reaching being the most primitive of the actions that is needed for any embodied agent to autonomously intervene in the world, the forward/inverse models presented in the previous section was also the first system we incorporated into the action generation system of the robot. The first step in this process was to close the loop between vision and action. As mentioned in section 2.3.3, the visual systems return the salient points of the object of interest (for example the goal) in the camera plane. So we need a mechanism to extract depth information or convert the two dimensional image plane coordinates of an object of interest returned by the visual modules into three dimensional Euclidian space coordinates that are used in the forward/inverse model relaxation. The second step was to train the artificial neural network (figure 4) that represents the forward kinematic transformation between intrinsic and extrinsic spaces necessary to evaluate (on-line) the Jacobian matrices (horizontal connections) in the relaxation chain. The final step was to test the performance of the computational model on the robot in a set of demanding reaching tasks in environments that impose different combinations of constraints. We present a brief overview of the implementation process in the following sub sections.

3.5.1 3D reconstruction using “babbling” movements

We used a Direct Linear Transform (Shapiro, 1978) based approach for stereo camera calibration, 3D reconstruction and robot-camera integration. As shown in figure 13, the robot itself is used for generating the training set needed by means of “babbling” movements. Equation 15 maps the 3D coordinates of a generic point in space (*x, y, z*) into the corresponding coordinates on the plane of the camera (*u, v*).

$$\begin{cases} u - u_o = -d\,\dfrac{r_{11}(x-x_o)+r_{12}(y-y_o)+r_{13}(z-z_o)}{r_{31}(x-x_o)+r_{32}(y-y_o)+r_{33}(z-z_o)} \\ v - v_o = -d\,\dfrac{r_{21}(x-x_o)+r_{22}(y-y_o)+r_{23}(z-z_o)}{r_{31}(x-x_o)+r_{32}(y-y_o)+r_{33}(z-z_o)} \end{cases} \tag{15}$$

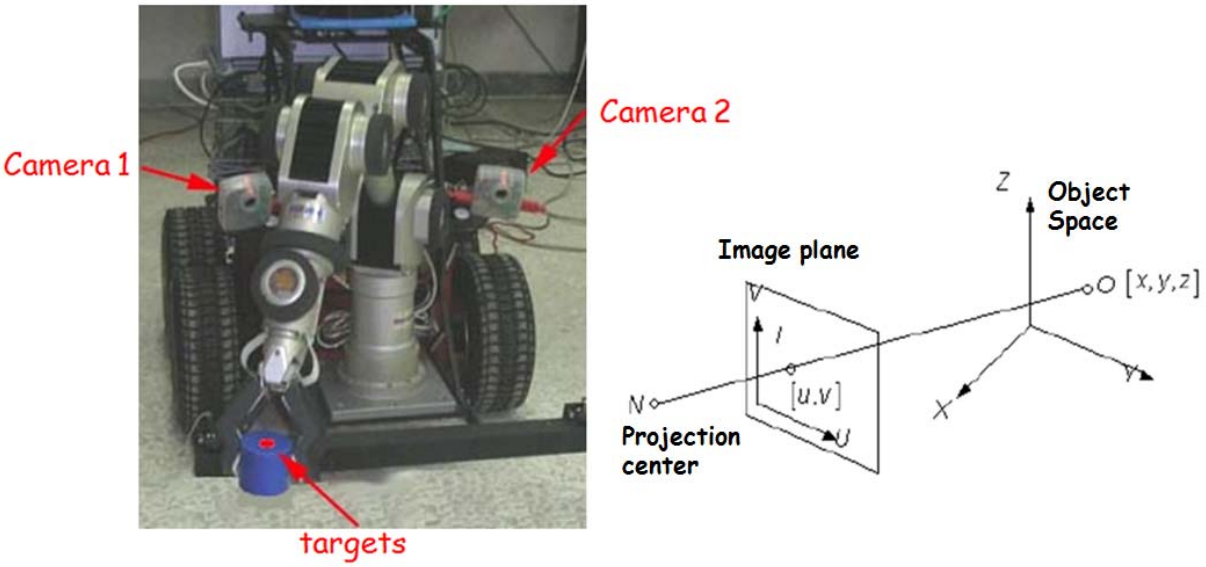


Figure 13. Performing Babbling movements to generate the training data required for 3D reconstruction of salient points (Regions of interest) as seen by the two cameras (and analyzed by the visual perception systems).

The equation is non linear with respect to both the transformation of coordinates and 7 unknown parameters: Camera position: [x_o, y_0, z_o], Camera orientation: [r_{ij}], and the principal distance: d. The trick of the DLT (Direct Linear Transform) is to express eq. 15 (non linear in 7 independent parameters) into an equation linear in the following 11 parameters (L_1-L_{11}) that, on the other hand, are not independent.

$$\begin{cases} u = \dfrac{L_1 x + L_2 y + L_3 z + L_4}{L_9 x + L_{10} y + L_{11} z + 1} \\ v = \dfrac{L_5 x + L_6 y + L_7 z + L_8}{L_9 x + L_{10} y + L_{11} z + 1} \end{cases} \tag{16}$$

From equation 16 it is be clear that we can bidirectionally move from visual space (u,v) to Euclidian space (x,y,z) if we can experimentally estimate the 11 unknown parameters (L_1-L_{11}) called as the calibration matrix of the system. In order to experimentally estimate the calibration matrix , we can use the robot itself to perform a set of babbling movements to generate the data. The procedure can be summarized as follows:

Let us consider that W is the workspace of the robot, i.e. the set of points reachable by the robot end-effector and identified by the vision system by a visual marker (for example a blue cylinder in figure 13) that is held by the gripper. The first step is to generate with the robot a set of control points i.e. targets of which we know, by experiments, both the Cartesian coordinates (x_i, y_i, z_i) and the corresponding camera coordinates (u_i, v_i). This involves 'i' iterations of the following four steps a) moving the arm randomly to some spatial location in the work space b) reading out the corresponding joint angles from the encoders c) using this information to perform the forward kinematics and estimate the distal position of the end-

effector 4) tracking the object gripped by the end-effector with the visual system and obtaining the camera plane coordinates (u_i, v_i) of the visual marker. We do not need a very dense sampling of W but we need at least to have a sufficient numbers of points on the boundary of W because DLT operates as a kind of interpolator and is known to have rapidly decreasing performance when we perform 3D estimates of target points outside the convex hull of the control points. Once the training set is obtained, the ***L*** matrix of the DLT is estimated for each camera by means of the Least Square method as expressed by equations 17 and 18.

$$\begin{bmatrix} u_1 \\ v_1 \\ .... \\ u_N \\ v_N \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\ ..... & & & & & & & & & & \\ x_N & y_N & z_N & 1 & 0 & 0 & 0 & 0 & -u_Nx_N & -u_Ny_N & -u_Nz_N \\ 0 & 0 & 0 & 0 & x_N & y_N & z_N & 1 & -v_Nx_N & -v_Ny_N & -v_Nz_N \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ ... \\ L_{11} \end{bmatrix} \quad (17)$$

$$U = A \cdot L$$

As seen from equation 17, for each iteration 'i' of a babbling movement, we get two equations (u1,v1 and so on) and N is the total number of iterations conducted to generate data. Even though the minimum number of babbling movements (N) required in this case is 6 (since there are 11 unknowns), keeping accuracy in mind we performed a set of 40 iterations of random movements with the robot to generate the data. The calibration matrix can then be estimated easily using equation 18.

$$L = \left[A^T A\right]^{-1} A^T \cdot U \quad (18)$$

After having estimated the calibration matrix L for both cameras, the reconstruction algorithm of the 3D coordinates of a target point (we ignore optical distortions for the moment) can be determined by 19 and 20.

$$\begin{bmatrix} u^{C1} - L_4^{C1} \\ v^{C1} - L_8^{C1} \\ u^{C2} - L_4^{C2} \\ v^{C2} - L_8^{C2} \end{bmatrix} = \begin{bmatrix} (L_1^{C1} - u^{C1}L_9^{C1}) & (L_2^{C1} - u^{C1}L_{10}^{C1}) & (L_3^{C1} - u^{C1}L_{11}^{C1}) \\ (L_5^{C1} - v^{C1}L_9^{C1}) & (L_6^{C1} - v^{C1}L_{10}^{C1}) & (L_7^{C1} - v^{C1}L_{11}^{C1}) \\ (L_1^{C2} - u^{C2}L_9^{C2}) & (L_2^{C2} - u^{C2}L_{10}^{C2}) & (L_3^{C2} - u^{C2}L_{11}^{C2}) \\ (L_5^{C2} - v^{C2}L_9^{C2}) & (L_6^{C2} - v^{C2}L_{10}^{C2}) & (L_7^{C2} - v^{C2}L_{11}^{C2}) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (19)$$

$$Y = A \cdot X$$

$$X = \left[x, y, z\right]^T$$

Hence,

$$X = \left[A^T A\right]^{-1} A^T \cdot Y \tag{20}$$

where L^{c1} and L^{c2} are the calibration matrixes obtained for camera1 and camera2 respectively. Once the 3D coordinates of image salient points in the robot reference system, are estimated by means of eq.20, they can be directly fed as goal to the forward inverse model pair for simulating/ executing actual movement of the arm.

The 3D reconstruction system was further used to estimate two other important pieces of information needed for the reasoning system 1) length of the tool and 2) orientation of the tool with respect to the robot body. In addition to the location of the center of mass of the object of interest, the visual modules also return the image plane coordinates of two most distant points along the major axes of the ellipse that encloses an object of interest (like a stick). The approximate length of the object can be hence estimated by finding the Euclidian distance between3D reconstructed coordinates of the respective image plane coordinates of the two most distant points in the object returned by the visual modules. The orientation of the object with respect to the body of the robot can also be estimated geometrically once we reconstruct the visual information of the object of interest in 3D space (and the vector representing the major axes of the ellipse enclosing the object) and receive the heading vector of the robot itself from the localizer (using information from the laser scanner).

Finally, the entire process involving iterations of random movements with the arm, estimating the spatial location of the end-effector through a kinematic transformation, requesting images from the vision system and analyzing them to get the image coordinates and after a set of 40 such iterations estimating the calibration matrix by solving equation 18 was compressed as an atomic software object that executes when ever commanded by the user/or the executive system of the robot. It is necessary to automate this process of learning the calibration matrix because the system tends to lose calibration from time to time as a result of small displacements in the cameras due to accidents, jerks experienced by the cameras due to the movement of the robot etc. In this way precious time and effort needed to monitor the robot while it is performing random movements and calibrating itself can be minimized.

3.5.2 Fine tuning the motor system of GNOSYS

In the previous section we dealt with the issues related to transformation of visual goals identified by the visual perception systems into corresponding spatial goals directed towards the appropriate end-effector and which has to be subsequently realized by the action generation system. In this section, we will briefly discuss notable issues related to implementation of the forward/inverse model on the GNOSYS robot and present some experimental results related to the performance of the architecture on the robot. The basic

input/output interfaces to the forward inverse model pair implemented on the robot is shown in figure 14.

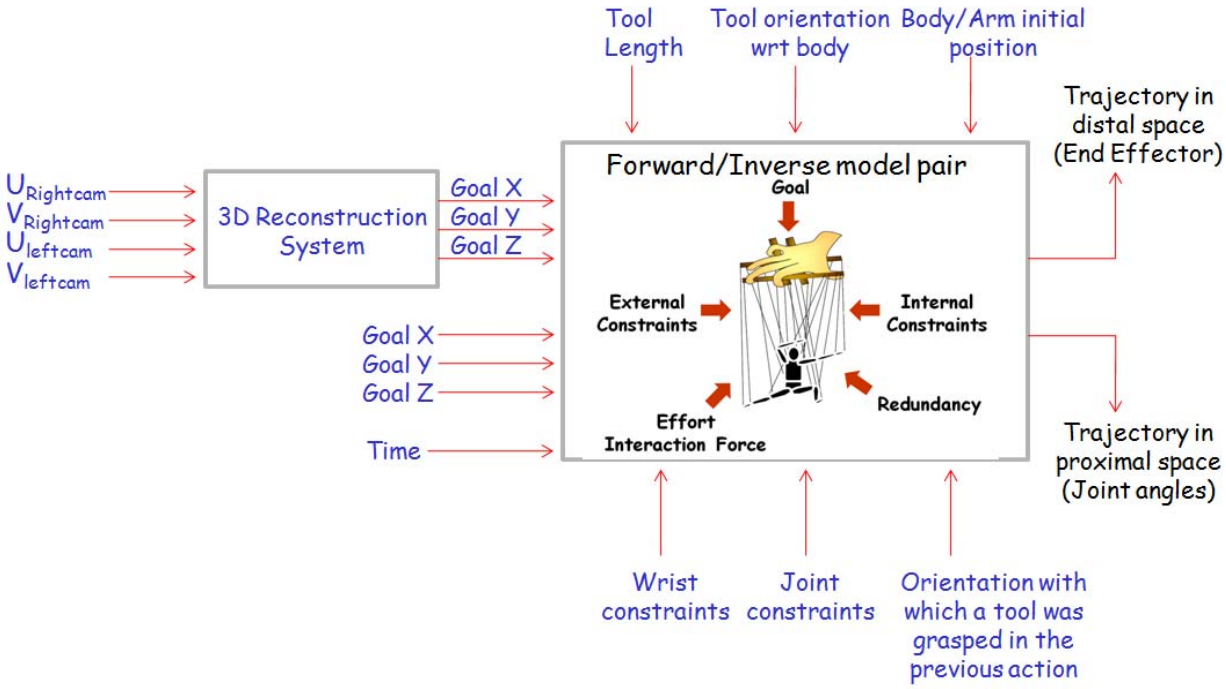


Figure 14.Input/output interfaces to the forward inverse model pair in the GNOSYS robot

The goal for reaching can come either from the user or from a higher level reasoning process and based on the nature of the high level being attempted (like grasping a ball in a animal reasoning scenario) this information can be both visual or spatial. Based on the goal, and the dynamically presented set of constraints related to the nature of the task (for example, just grasping a cylinder in front of the robot, grasping a stick with a particular wrist orientation, using a gripped stick to reach or push a ball) a custom relaxation network is built in real time as described in section 3.4. The output of the relaxation is a set of virtual trajectories in both distal and proximal spaces (as a function of time). This information however not sent to the actuators directly, but is sent back to the process that requested this reaching action to be virtually executed, which analyses the output and makes a decision of physically executing this action or doing something else (if the F/I model simulation predicts that the movement may not be successful directly, may be a change in body orientation is needed, may be a tool needs to be used etc). Assuming that in case the higher level process makes a decision to physically execute this action, all the information returned by the forward inverse model is packed in the form of a filler role binding, filler being the detailed information of the action and role being the temporal position of this action in the complete behavior that needs to be generated. This filler role binding becomes an atomic element in the overall plan descriptor that is sent to the execution system. We will deal with these issues in greater details in the

next chapters. For the time being a reader may visualize the complete behavior of the system as a thread in time on which there are many basket's attached, the contents inside the basket being the detailed information about atomic action (what is the target object, where is it located, what is the end-effector that needs to be used, what is the trajectory of the end-effector, what are the motor commands etc). For the simplest case of reaching (a ball that is in front of the robot, visible and reachable), the thread of time will have only one basket attached, that contains the motor commands needed to reach the goal. However we need to recall here that are not dealing with the problem of 'reaching targets', but are dealing with the problem of 'reaching goals' and hence this level of organization of information is critical to tame the unstructured world in which just reaching a ball may require a large sequence of actions, each action using different end-effectors (tool, arm, body) and using different high level computational models (vision, reaching, spatial navigation) to derive its own plan and that needs to be executed in a temporal order that is not dictated by the user but by the environment and the reasoning process that attempts to realize the goal in that environment (which is a function of the causal knowledge that the robot embodies).

We need to recall here that we are not dealing with the problem of 'reaching targets', but are dealing with the problem of 'reaching goals' and hence this level of organization of information is critical to tame the unstructured world in which just reaching a ball may require a large sequence of actions, each action using different end effectors (tool, arm, body) and using different high level computational models (vision, reaching, spatial navigation) to derive its own plan and that needs to be executed in a temporal order that is not dictated by the user but by the environment and the reasoning process that attempts to realize the goal in that environment (the reasoning process further is a function of the causal knowledge that the robot embodies at that point of time in its *life*).

Coming back details inside the forward/inverse model, we trained a two layer backpropgation network (5:48:36:3) to map the kinematic transformation from which the jacobians are extracted online during the relaxation process using equation 19 and 20. The updated expression for runtime computation of jacobians for a two layer MLP is given by equation 21,22 and 23. We also note that the complete relaxation process is also computationally inexpensive and can execute in real time on the robot, all computations taking place in one laptop (core2duo processor and running windows) externally connected to the robot (through Gnosys DLL).

$$J_{ki} = \frac{\partial x_k}{\partial q_i} = \sum_l w_{lk}.g'(p_l).(\sum_j w_{jl}.g'(h_j).w_{ij}) \qquad (21)$$

$$\dot{x} = J \cdot \dot{q} \Rightarrow \sum_i J_{ki}\dot{q}_i = \sum_i \left(\sum_l w_{lk}.g'(p_l).\sum_j w_{jl}.g'(h_j).w_{ij} \right)\dot{q}_i \qquad (22)$$

$$T = J^T \cdot F \Rightarrow T_i = \sum_k J_{ik}F_k = \sum_k \left(\sum_l w_{kl}.g'(p_l).\sum_j w_{lj}.g'(h_j).w_{ji} \right)F_k \qquad (23)$$

where i, j, l, k are the input, hidden and output layers respectively, W is the weight matrix and q_i and x_k are the inputs and outputs of the ANN.

Another interesting implementation detail we must note concerning the wrist orientations while reaching sticks is that, unlike the planar case (as explained in section 3.4), in a 3D space there is no unique solution to define the targets for the wrist fields. In this case, the possible solutions lie on the locus of a circle visualized on a conical surface that is defined by the specified orientation of the wrist and L4. This is shown in figure 15b.

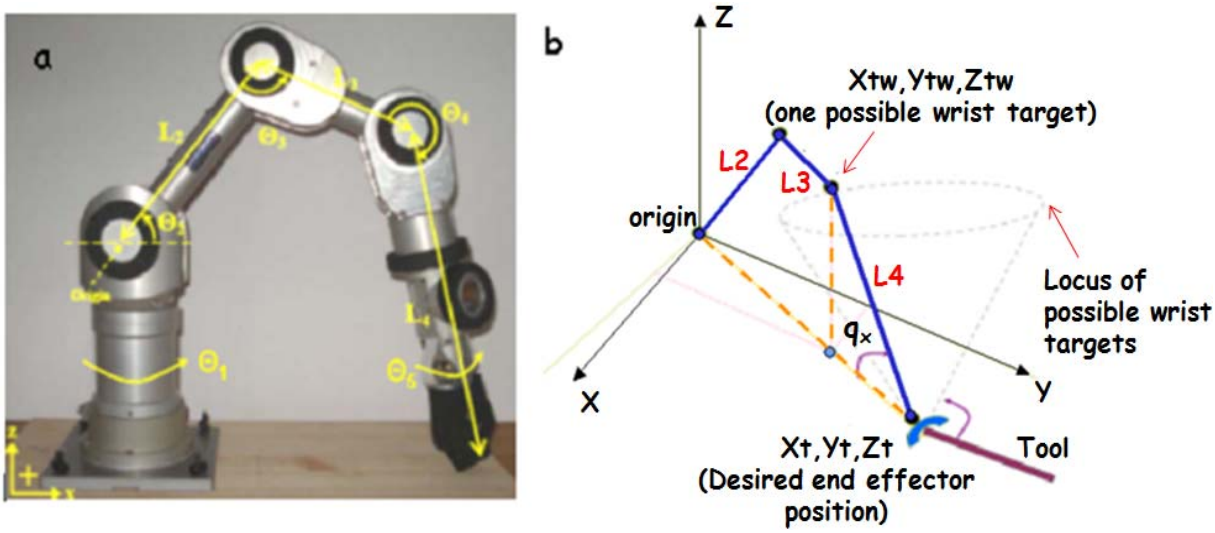


Figure 15: (a) Katana Arm; (b) Specification of targets for wrist orientation. The possible wrist targets lie on the locus of a circle visualized on a conical surface that is defined by the specified orientation of the wrist (qx) and length of link L4.

One possible way to derive target for the wrist X_{tw} is by solving quadratic equation 24. In that case the Y and Z wrist targets can be derived using eq. 25.

$$(1 + {y_t^2}/{x_t^2})x_{tw}^2 - 2(x_t + {y_t^2}/{x_t})x_{tw} + (x_t^2 + y_t^2 - d^2) = 0 \qquad (24)$$

$$y_{tw} = x_{tw}({y_t}/{x_t}) \quad \text{and} \quad z_{tw} = L_4 Sin(q_x) \qquad (25)$$

where x_t ,y_t and z_t are the targets for the end-effector and q_x is the desired orientation of the wrist.

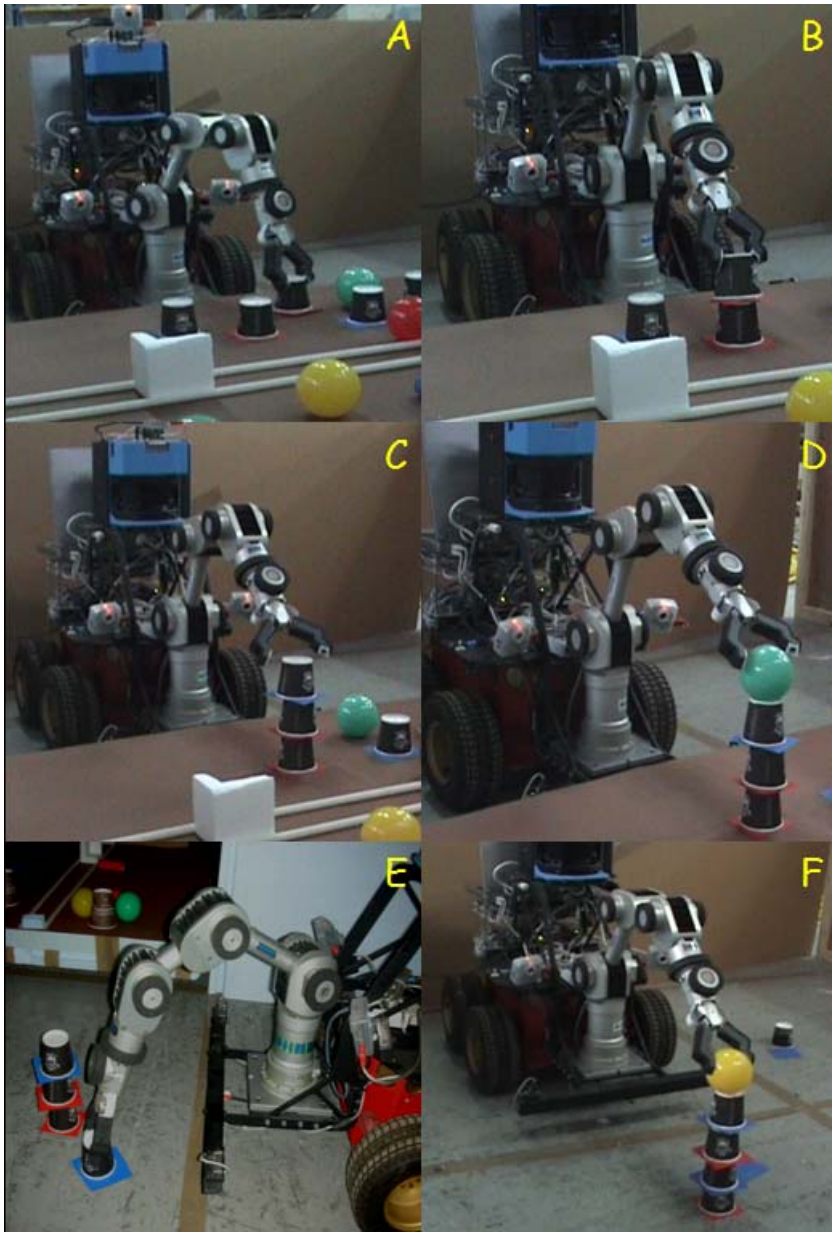


Figure 16. Panels (A-D) show the GNOSYS robot on the process of sequentially stacking 3 coffee cups and a ball on the table. Panels E-F show snapshots of instances of stacking objects on floor. Stacking/Unstacking tasks are extremely useful for testing the global functionality/performance/ interactions of different high level computational modules like visual recognition, 3D reconstruction, forward/ inverse mental simulation of reaching, real movement execution, and at the same time test and debug low level communication protocols, during this preliminary and most critical level of sensory-motor integration.

Finally, we rigorously tested the forward/inverse model architecture on the GNOSYS robot for all possible primitive actions needed for reliable object manipulation and information gathering. This was the first step towards fine tuning the system for more complex learning/acting cycles that we consider in the next chapters.

After preliminary experiments on reaching, grasping and stacking tasks (figure 16), we conducted an extended series of experiments on generation of more complex actions related to tool use, tool orientation discussed in section 3.4 using a planar arm simulation on the GNOSYS robot. Actions critical during reasoning scenarios (like exploiting magnetic coupling in small sticks to make a longer stick, pushing objects, sliding objects in a trapping groove) were tested and fine tuned so as to reach an excellent level of accuracy and repeatability. Figure 17 presents some of the snapshots of the agent performing a variety of motor actions using the computational framework presented in this chapter.

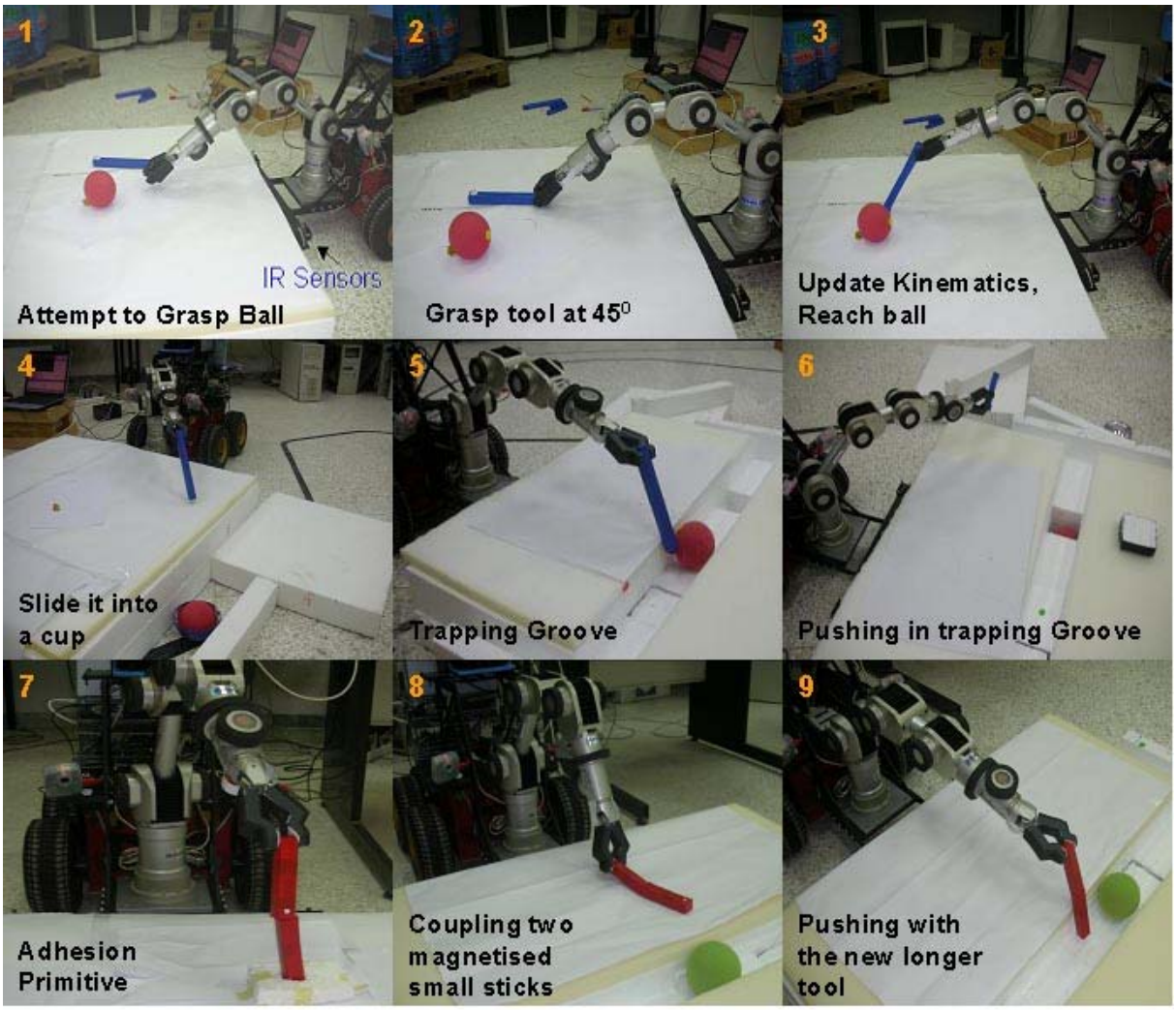


Figure 17: Action generation system 'in Action' on the GNOSYS Robot: Experiments; Panel 1: Even if the target is unreachable the F/I model simulation converges to the best possible solution; Panel 2: Appropriate Wrist orientation to grasp stick placed horizontally on the table; Panel 3: Reaching extended; Panel 4: Pushing objects placed on the table; Panel 5: Trapping Groove paradigm; Panel 6: Sliding objects inside the trapping groove using sticks of appropriate lengths; Panel 7: Adhesion Primitive: Magnetically Coupling two small sticks, in this case the second stick is oriented vertically; Panel 8: Coupling 2 small sticks, in this case the tool in the gripper is oriented at approximately 10 degrees, to reach the end of the second stick placed horizontally on the table; Panel 9: Sliding the green ball with the new longer tool.

3.6 'Towards Internal body models for Humanoids': Extracting general principles

A primitive example in which two hands must separately achieve different functions in different ways and yet be coordinated together in space and time can be seen in the behavior of nut cracking chimpanzees. While they use one of their hands to hold and position the nut, the other hand is effortlessly coordinated to strike the nut with a stone in order to crack it. Be it an example of a little bird trying to reach for a fish in the water using its beak, with both legs connected to a branch of a tree or a waiter in a busy restaurant simultaneously transporting a range of objects held in both arms to different spatial locations or a group of children holding hands and playing, all symbolize an elegant coordination of a well connected system of body and environment, in order to realize the goal being actively pursued. How does the brain plan this kind of goal dependent structural and functional coupling between degrees of freedom afforded by the body and degrees of freedom afforded by the immediately available world? How does it then control this dynamically generated chain of coupled degrees to freedom in order to achieve the desired end point interaction? In the previous section, we described a computational framework based on the idea of passive motion paradigm (Mussa Ivaldi et al, 1988), using which composite forward/inverse models can be built based on combinations of task dependent constraints, and then used to derive motor commands necessary to coordinate movements of the body (body + tool) in order to realize a goal. The discussion was however restricted to controlling and generating a range of complex actions with a single (redundant) kinematic chain, for example, the KATANA arm of the GNOSYS Robot (often coupled to tools). In this section, taking an example of a 53 degree of freedom baby humanoid 'iCub', we describe how the computational framework introduced in the previous sections easily scales up with the complexity of the body that has to be controlled. Humanoid robots particularly, have a large number of "extra" joints, organized in a humanlike fashion with several kinematic chains. Taking an example of complete upper body coordination in the baby humanoid iCub, we will demonstrate how the same force fields based relaxation process can dynamically coordinate the movements of a limb, network of limbs (e.g. left arm–waist–right arm) or networks of external objects kinematically and dynamically coupled to the body/internal body model (e.g. right arm-tool-left arm, as in driving a car or transporting objects using two arms). As we progress with these examples, we will also try to extract some general principles on how composite

How does the brain plan this kind of goal dependent structural and functional coupling between degrees of freedom afforded by the body and degrees of freedom afforded by the immediately available world? How does it then control this dynamically generated chain of coupled degrees to freedom in order to achieve the desired end point interaction?

forward/inverse models involving different kinematic chains in the body (connected in serial or parallel with each other or with external objects) can be generated on the fly and coordinated in a task specific manner, in space and time.

3.6.1 Forward/Inverse model for upper body coordination in the baby humanoid 'iCub'

A. 'iCub' general features

The iCub is a small humanoid robot of the dimensions of a three and half year old child and designed by the RobotCub consortium, a joint collaborative effort of 11 teams from Europe, three teams from Japan and two teams from USA. The 105 cm tall baby humanoid is characterized with 53 degrees of freedom: 7 DOF for each arm, 9 for each hand, 6 for the head, 3 DOF for the torso and spine and 6 DOF for each leg. The current design uses 23 brushless motors in the arms, legs, and the waist joints. The remaining 30 degrees of freedom are controlled by the Faulhaber DC motors. The iCub body is also endowed with a range of sensors for sensing forces, torques, joint angles, inertial sensors, tactile sensors, 3 axis gyroscopes, cameras and microphones for visual and auditory information acquisition. Most of the joints are tendon driven, some are direct, according to the placement of the actuators which is sort of constrained by the shape of the body.

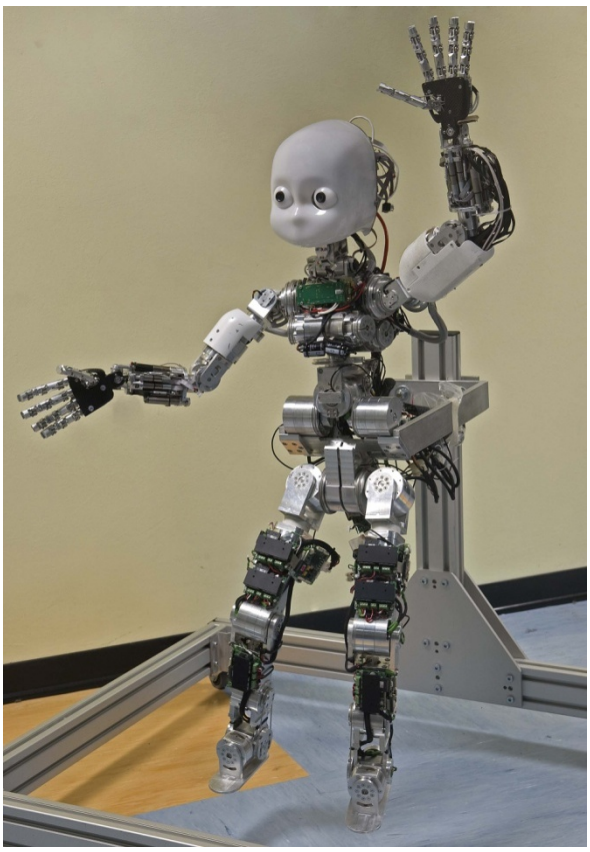


Figure 18. 'Where beauty meets complexity'.
The 53 DOFs iCub robot.

Apart from the interface API that speaks directly to the hardware, the middle ware of iCub software architecture is based on YARP (Metta et al, 2006), an open-source framework that

supports distributed computation with a specific impetus given to robot control and efficiency. The main goal of YARP is to minimize the effort devoted to infrastructure-level software development by facilitating modularity, support for simultaneous inter-process communication, image processing, as well as a class hierarchy to ease code reuse across different hardware platforms and hence maximize research-level development and collaboration. With special focus being given on manipulation and interaction of the robot with the real world, the iCub is characterized with highly sophisticated hands, flexible oculomotor system and reasonable bimanual workspace. We refer the interested reader to the RobotCub database (www.robotcub.org) for detailed technical description of the body geometry, kinematics, electronics, software architecture and CAD diagrams.

As a starting point, we initially implemented the forward/inverse model to control only the left arm of the humanoid with force fields applied at the end-effector and the wrist in order to determine the total pose of the hand. As shown in figure 19, there are two force fields: one that pulls the end-effector to the target and the other (F2) that attracts the wrist or hand to a proper pose with respect to the target. Figure 19 resembles the model shown in figure 10 with the major difference being in the number of degrees of freedom in the intrinsic space that are being controlled in the two cases. Another new addition for the case of humanoids is the ground reference node shown at the shoulder. Since there was only one kinematic chain involved in the case of the GNOSYS Robot, the base of the arm was the default reference. However, since there are many possible kinematic chains that can be coordinated simultaneously in the humanoid, it is necessary to identify the start and end points in the body model between which the force fields generated by the goal will propagate, and beyond which the force fields generated by the goal will not propagate.

We also note here that irrespective of the redundancy of the system, it is always possible to map the entire distal space to the proximal space by traversing through the horizontal (geometric) and vertical (elastic) links in the computational chain. The joint space hence becomes a natural site where multiple articulatory constraints are concurrently combined, to derive an incremental change in configuration of the body (and motor commands) and this process keeps circulating in the loop till the time the system achieves an equilibrium configuration i.e. such that the net disturbance force circulating in the chain is zero (this implies that the body has reconfigured to reach a final pose such that $X_{T1}=X_1$ and $X_{T2}=X_2$). Figure 20 shows four examples of applications of the computational model to the left arm of iCub: the starting position of the arm is symmetric with respect to the right arm and the four panels show the final poses with desired hand orientations around the target. Please note that the same time base generator $\Gamma(t)$ coordinates the motion of all the joints of the arm and the wrist.

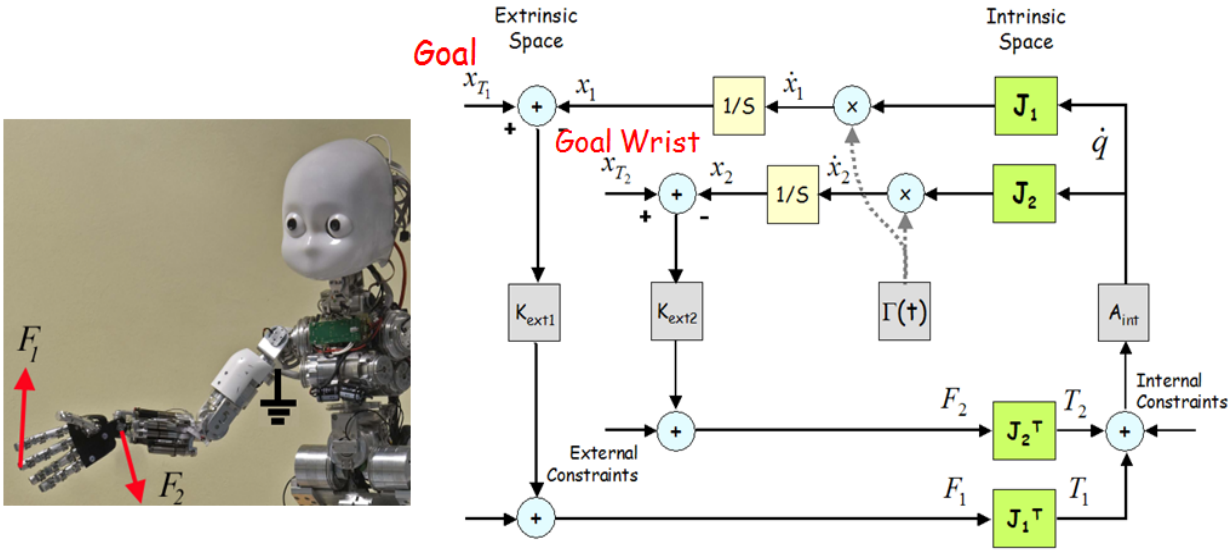


Figure 19. Composite PMP network with two attractive force fields applied to the arm of iCub: a field F_1 that identifies the desired position of the hand/fingertip and a field F_2 that helps achieving a desired pose of the hand via an attractor applied to the wrist. This network assumes that the shoulder of the arm is "grounded": this means that the two force fields do not propagate beyond the shoulder joint.

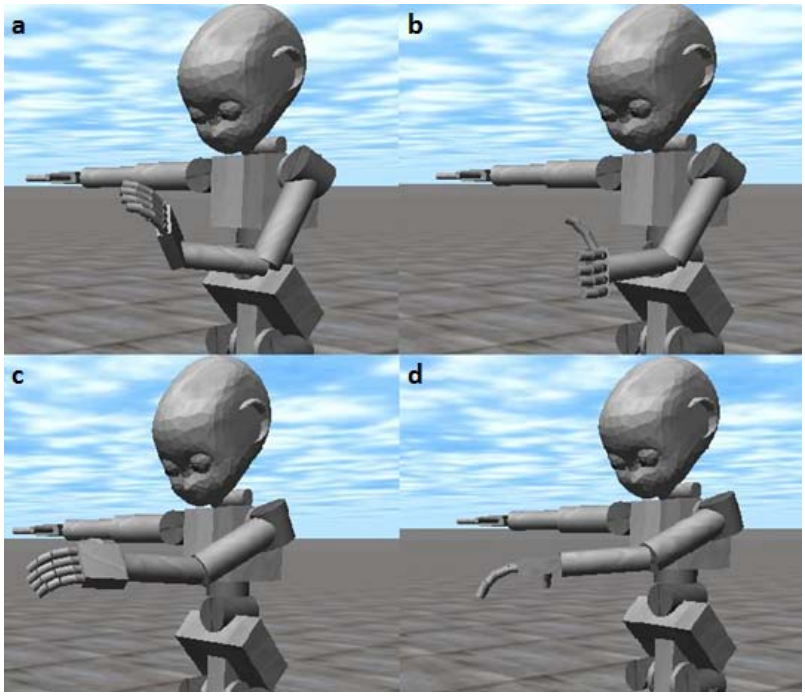


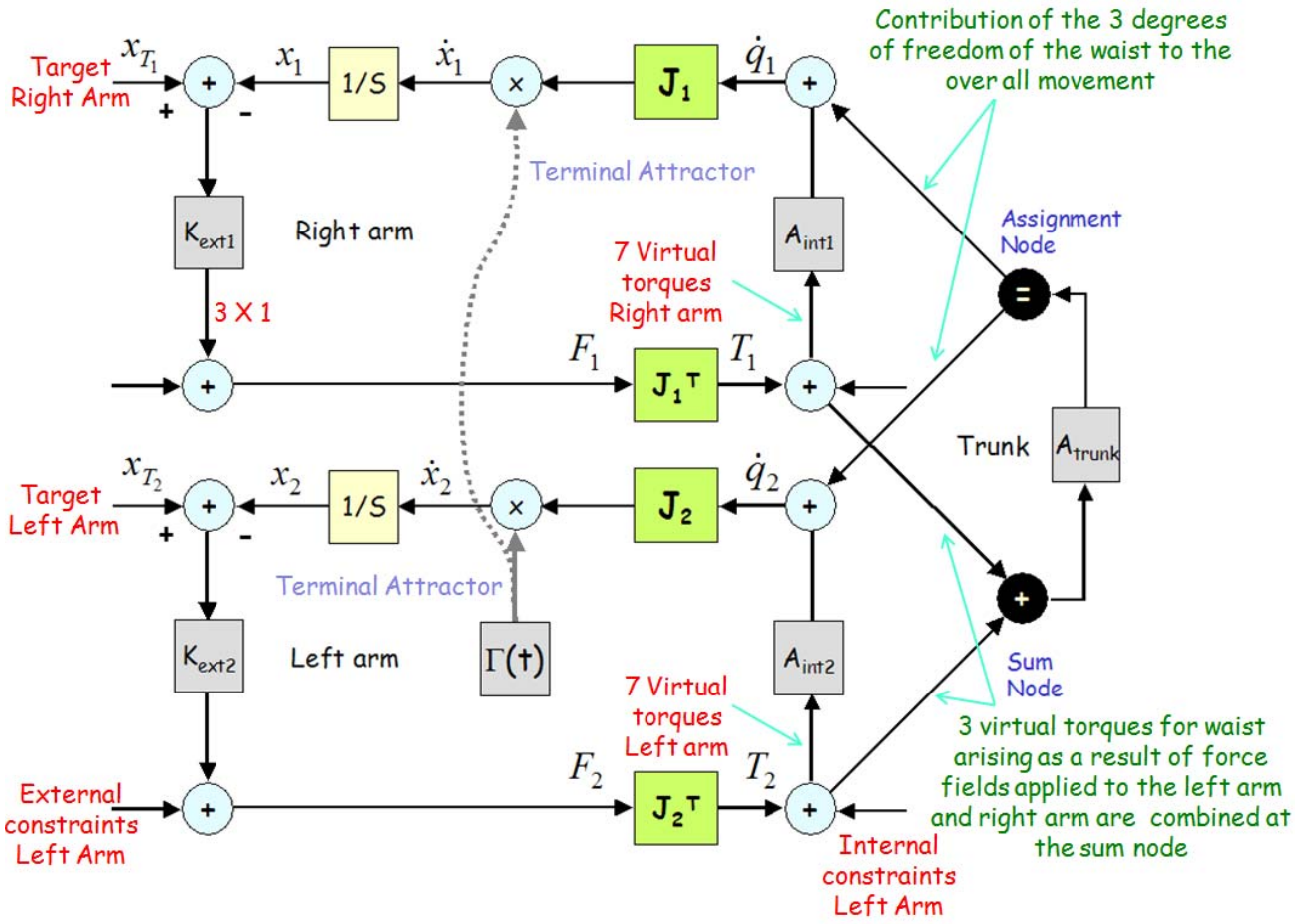
Figure 20. Examples of applications of the computational model of figure 18 to the left arm of iCub: the starting position of the arm is symmetric with respect to the right arm and the four panels show different final poses reached from the same initial position. Also note that the time base generator $\Gamma(t)$ coordinates the motion of all the joints of the arm and the wrist.

B. Combined relaxation of two arms and the waist in iCub

We now consider a bimanual coordination task, in order to reach at the same time two target points of an object in order to grasp it. As mentioned in the previous section, since there are multiple kinematic chains that can be coordinated simultaneously in humanoids, it is first necessary to decide the ground reference in the body beyond which there is no propagation of the goal induced force fields. Of course there is no need to pre specify this information about the ground reference and we will discuss later as to how the ground reference can dynamically change based on the nature of the task. Let us assume for the time being that the waist is grounded, and the task at hand is to coordinate all degrees of freedom involved in the 'right arm-waist-left arm chain' during the bimanual reaching task. Since we have already built a relaxation network for the left arm, and since the left and the right arm are acting in parallel, it is easy to hint that for the case of bimanual reaching there will be two parallel computational chains (one for each arm) whose end-effectors will be pulled by their respective goal target. The problem now is to the couple these two dynamical systems into a single relaxation process. We also observe that the waist of the robot is linked serially to the two arms and we need a way to couple the overall relaxation of the two arms with the relaxation at the waist. In other words, the three degrees of freedom allowed at the waist must also contribute towards reaching the respective targets of the two arms. In order to achieve this, we introduce two new nodes (in addition to generalised displacement, force and ground nodes that already exist) in the computational framework : a sum node and an assignment node. The resulting forward/inverse model for bimanual coordination is shown in figure 21.

In complex kinematic structures, characterized by several serial and parallel connections, the sum and assignment nodes can be used to add or assign displacements and forces to different connecting elements of the kinematic chain (in this case the left arm-waist-right arm network). In the resulting computational scheme shown in figure 21, A_{trunk} is the virtual admittance matrix of the waist. We may consider the scheme of figure 21 as a composite forward/inverse model, dynamically created for this specific task, reconfiguring the basic networks of the two arms (that were grounded at the shoulder as in figure 19). The transpose jacobians incrementally transform the force fields generated by the goal in each chain into ten virtual torques (7 for the respective arms and 3 for the waist). The virtual torques incrementally computed for the waist as a result of the force fields experienced by the two arms are summed at the sum node and transformed into 3 incremental joint rotations at the waist through the admittance matrix. The assignment node propagates the resultant incremental displacement computed at the waist back to the computational chain of the two arms. At the same time the incremental displacements at the joints of the each arm is also computed by using the 7 virtual joint torques and joint admittance matrices. The jacobians now compute the incremental update in the configuration of the body as a result

of the incremental displacements at different joints. In this one cycle through the computational chain, the whole upper body has incrementally reconfigured to a new pose towards reaching the respective goals of the two end-effectors. Part of the solution contributed by the waist, part of it contributed by the degrees of freedom of the two arms, based on their relative admittances. This cycle of propagation of disturbances through the computational chain continues till the time the whole upper body attains equilibrium (i.e there are no disturbance forces circulating in the network). This is the final solution of the complete relaxation process.



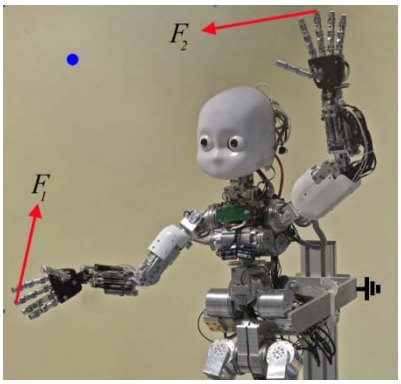


Figure 21. Composite PMP network with two attractive force fields applied to the right and left arms of iCub respectively. The waist joint of the robot is "grounded" in this case. The "sum node" allows the two force fields to be combined in determining the motion of the waist. The "assignment node" propagates to the two arms the motion of the waist. In this way the motion of each arm is influenced by both force fields.

By actively modulating A_{trunk} we can control the contribution of the degrees of freedom in the waist to the overall relaxation of the body to the two force fields applied to either end-effectors, without affecting the trajectory of the end-effector. If the trunk is very stiff, only the DoFs of the arms contribute to the final solution reached by the system: this is equivalent to "ground" both shoulders. For example if the target is within the arm's length (figure 20), the net can be grounded on the arms by "freezing" the trunk. For a farther away target (figure 22) this strategy would not allow to reach the target and then it is necessary to "unfreeze" the trunk choosing a sufficiently high value of the trunk admittance. In other words, a variable number of degrees of freedom of the body can become dynamically involved in the relaxation process based on the target goal to be reached.

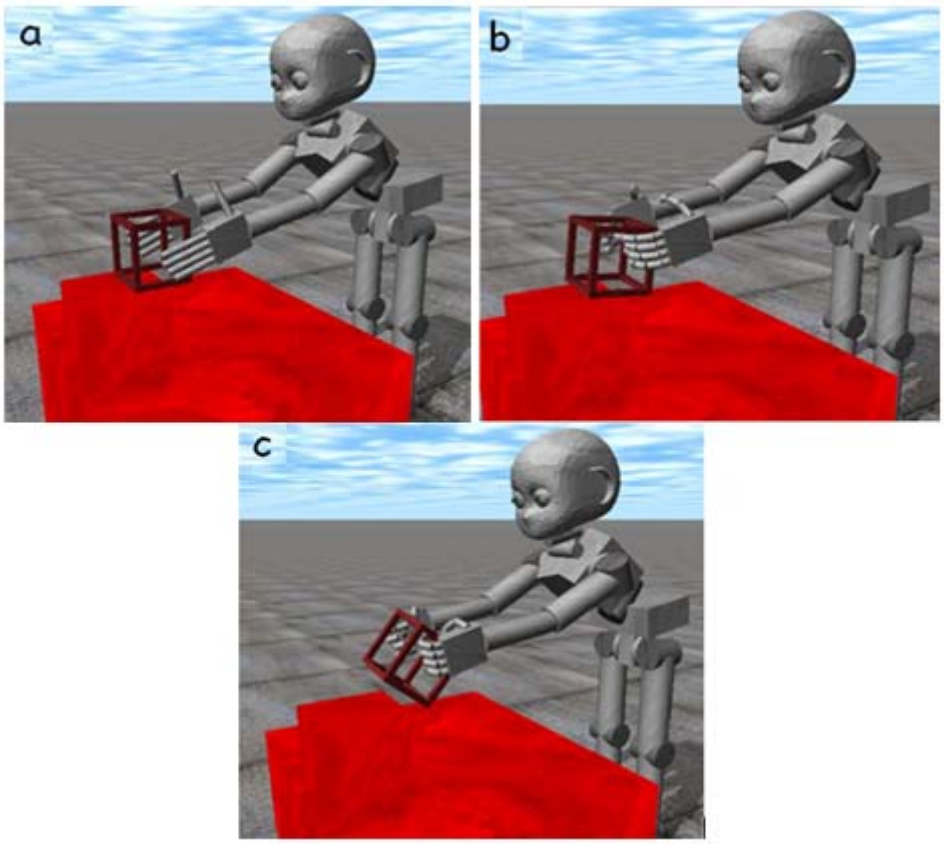


Figure 22. Example of applications of the computational model of figure 21. The initial configuration is with both arms fully stretched sideways. In this case the object is too far away and cannot be reached without a coordinated bending forward of the trunk (panel a). After reaching the object is grasped (panel b) and the is lifted (panel c). Note the influence of gravity on the object during lifting that makes the object rotate.

Figure 22 shows another example of bimanual reaching with "frozen" or "unfrozen" trunk. The figure also shows (lower panel) that the kinematics of the end-effector is the same in both cases although the overall motion of the body is different.

It is also possible for the robot to learn and exploit different settings of the stiffness values (end-point stiffness, joint stiffness) in a goal oriented fashion, in the simplest case as a function of the extrinsic space. This can be achieved by pseudo-randomly "titillating" their values from a fixed range and assigning a cost to each final solution (in the simplest case the cost can be a function of whether the target was reached successfully or not, and if the final posture attained is in permissible/favorable range of motion of all degrees of freedom). This can also be interpreted as shifts in the grounding of the kinematic network involved in the passive motion paradigm.

If the trunk is very stiff, only the DOF of the arms contribute to the final solution reached by the dynamical system: this is equivalent to "grounding" both the shoulders. By actively modulating the stiffness of the waist, we can control the relative contributions of the DOF in the waist to the overall relaxation of the body. In other words, variable number of degrees of freedom of the body can become dynamically involved in the relaxation process based on the nature of the goal that has to be reached.

It is worth noting that the motions of the two arms are not totally independent when the trunk is "unfrozen": there is a propagation of the force field applied to one hand to the other arm and vice-versa. The portion of the model that corresponds to the trunk of the body is the element through which the interaction takes place (figure 23). The sum and assignment nodes in general are dual in nature: if an assignment node appears in the kinematic transformation between extrinsic and intrinsic motor spaces, then a sum node appears in the force transformation between the same motor spaces. This is a consequence of conservation of energy that is structurally invariant across the different work units.

Figure 24 shows a simulation result of an example task in which only the left arm is given a target to be reached starting for an initial equilibrium configuration of the two arms. The trunk admittance is mid-range and we observe that, although the right target did not change, at the end of the task there is a slight readjustment in the posture of the right arm. This is a kind of interference between the two PMP networks: the goal oriented force field applied to the left arm propagates to the right arm and slightly displaces/disturbs it from its target goal. Since the right arm was not supposed to move from the previously reached goal, now there is a small force field (that was earlier zero) generated autonomously in the kinematic chain of the right arm in response to this disturbance. This new force field also circulates all through the chain and the whole body reconfigures to a new posture. In other words, the external motion induced in the left arm by the force field breaks the equilibrium in the right arm and the complete system relaxes to a new solution that satisfies both goals simultaneously.

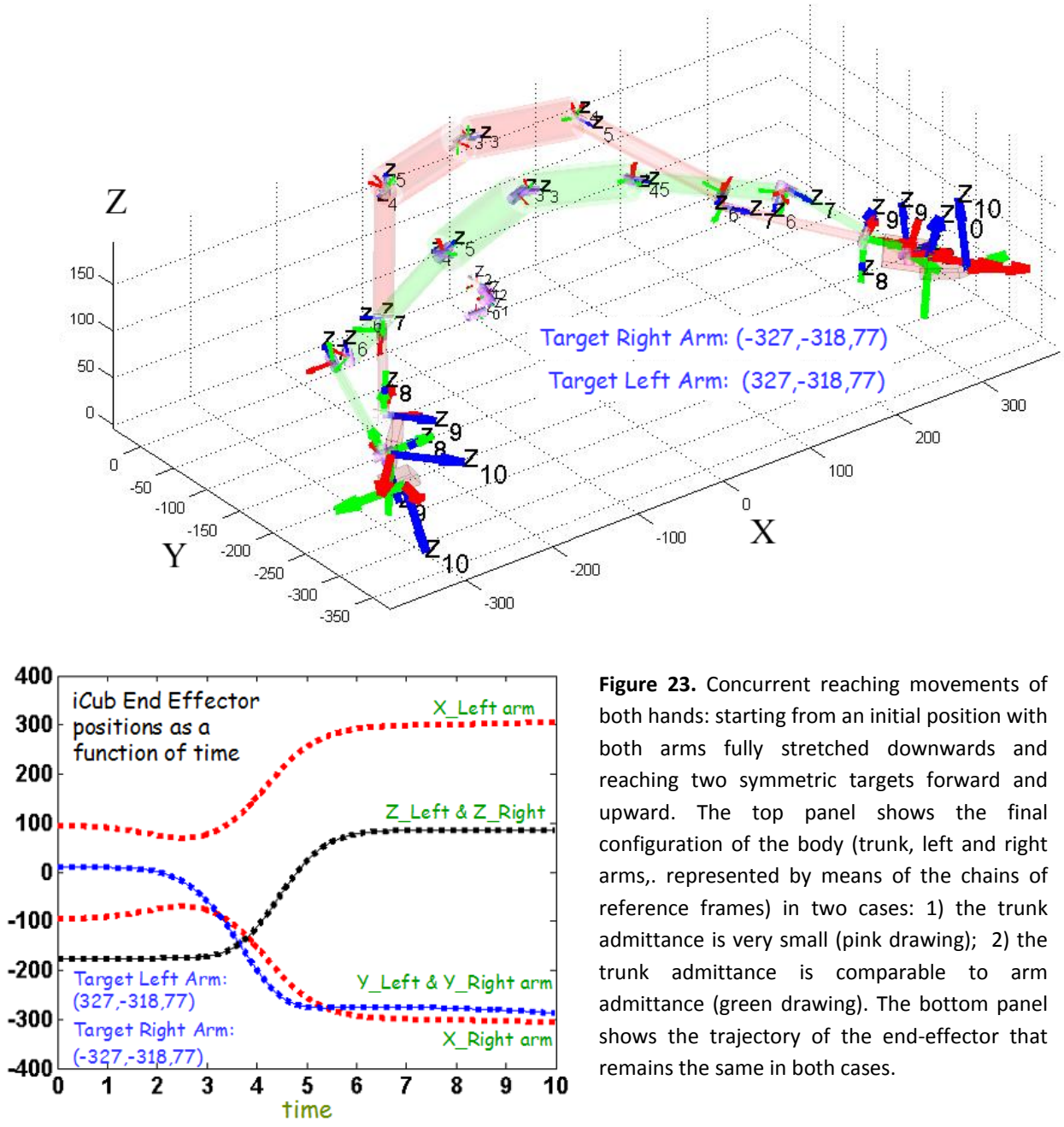


Figure 23. Concurrent reaching movements of both hands: starting from an initial position with both arms fully stretched downwards and reaching two symmetric targets forward and upward. The top panel shows the final configuration of the body (trunk, left and right arms,. represented by means of the chains of reference frames) in two cases: 1) the trunk admittance is very small (pink drawing); 2) the trunk admittance is comparable to arm admittance (green drawing). The bottom panel shows the trajectory of the end-effector that remains the same in both cases.

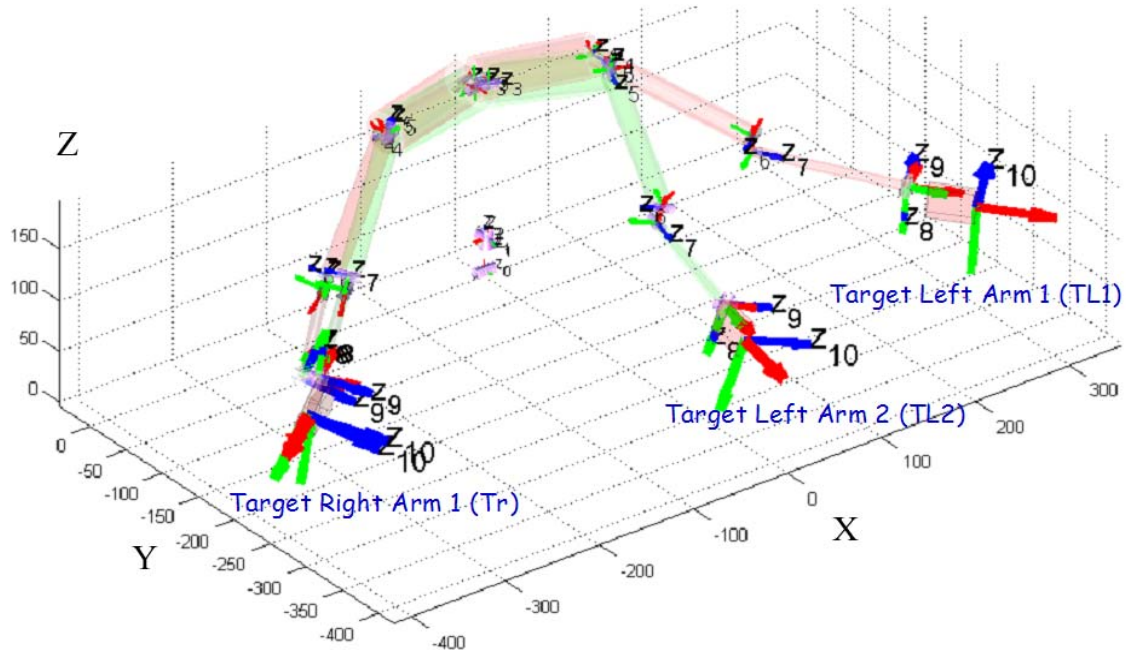


Figure 24. Reaching movement of the left arm from an initial position (green drawing) to a final position (pink drawing). The trunk admittance is comparable to the arm admittance. The figure shows an interference between the two arms: the force field applied to the left arm is propagated to the right arm resulting in a small perturbation in its the initial posture.

Instead of 'actively' controlling the body that in turn controls the external object and carries it to the spatial goal (which may be extremely complicated to achieve in computational terms considering the number of constraints that must be explicitly accommodated to into the controller to achieve this successfully), the computational framework of 'passive' motion introduced in this chapter moves exactly in the reverse direction i.e the goal pulls the external object that in turn pulls the extrinsic body (end-effector) that in turn pulls the intrinsic body (joints, muscles). An important consequence of this process is that any unpredictable disturbance occurring anywhere in this fully connected relaxation chain automatically leads to a dynamic reconfiguration of the body so as to compensate its effect.

3.6.2 When two arms are structurally connected to external objects

In section 3.4 we presented several examples of building composite computational chains based on PMP to simulate a reaching action towards a target object taking into account a

range of internal and external constraints. In the previous section, we saw how this computational model can be further extended to simultaneously coordinate multiple kinematic chains arising in a humanoid robot. In this section we will extract some general principles governing the runtime composition of a task specific forward/inverse model pair. For this purpose, we will consider an example of a combined task that consists of reaching, grasping, and then transporting an object (a cube) to a different spatial location using the two arms.

In spite of the fact that humans (almost unconsciously) execute such tasks with noticeable ease, it is worth observing the fact that in this example it is not even straight forward to specify the goal of the task (in computational sense) if we want an artificial agent to do the same. For example, the goal may be verbally described in one way as 'both arms must move in a coordinated way in order to allow the object coupled in between them to reach its target location in space'. The computational complexity in action generation for this bimanual transportation task stems from the fact that once an object is grasped successfully using the two arms, the task of transporting it poses stringent constraints both on the movement trajectory of the two arms as well as the timing of the motion of both arms. The basic requirement is that both arms must move to the target in such a way that they are always in contact with the object and any unpredictable effect on the object must be compensated by suitable reconfiguration of the body.

Instead of actively controlling the body that in turn controls the external object and carries it to the goal (which may be extremely complicated to achieve in computational terms considering the number of constraints that must be explicitly accommodated to into the controller achieve this successfully), the computational framework introduced in this chapter moves exactly in the reverse direction i.e the goal pulls the external object that in turn pulls the body (end-effector) which in turn pulls the intrinsic elements in the body (joints, muscles). Hence the general principle in composing a PMP based forward/inverse model pair is to always move from the most distal space to the most proximal space (from the goal to the external object, and then to the body). In this way, the external object in a sense is always kinematically and dynamically coupled with the body. A custom forward inverse model for performing the bimanual transportation task is shown in figure 25. The computational model by itself makes no difference between the representational schema of the motor spaces of the body and the external object. The tool space is represented exactly in the same way as the body, by means of a generalized force and position node, liked vertically by a virtual admittance matrix A_E (characterizing the incremental transformation from force information to position information in the tool space), and horizontally by the device Jacobian matrix J_D that form the interface between the body and the tool device (and based on the geometry of the task). During the transportation phase, the cube is pulled towards the goal target by one virtual force field; this pull in turn disturbs the end-effectors (both hands) that passively comply to the externally imposed motion; this disturbance then

circulates to the proximal space through the Jacobian matrices to derive an incremental change in the joint angles and so on. If the motor commands derived by this process of virtual relaxation are fed to the robot, the robot will reproduce the same motion. The external object can be a simple cube, as in fig. 26, or a more complicated tool with several new controllable degrees of freedom: for example, two arms linked in parallel to a steering wheel while driving a car. In this case, what changes in the computational model is just the device jacobians J_E that forms the interface between the body and the external object. In the case of the steering wheel task, J_E maps the transformation between the rotation of the steering wheel and the corresponding differential displacement seen at the end-effectors (hands). Everything else in the computational chain remains exactly the same and behaves the same way.

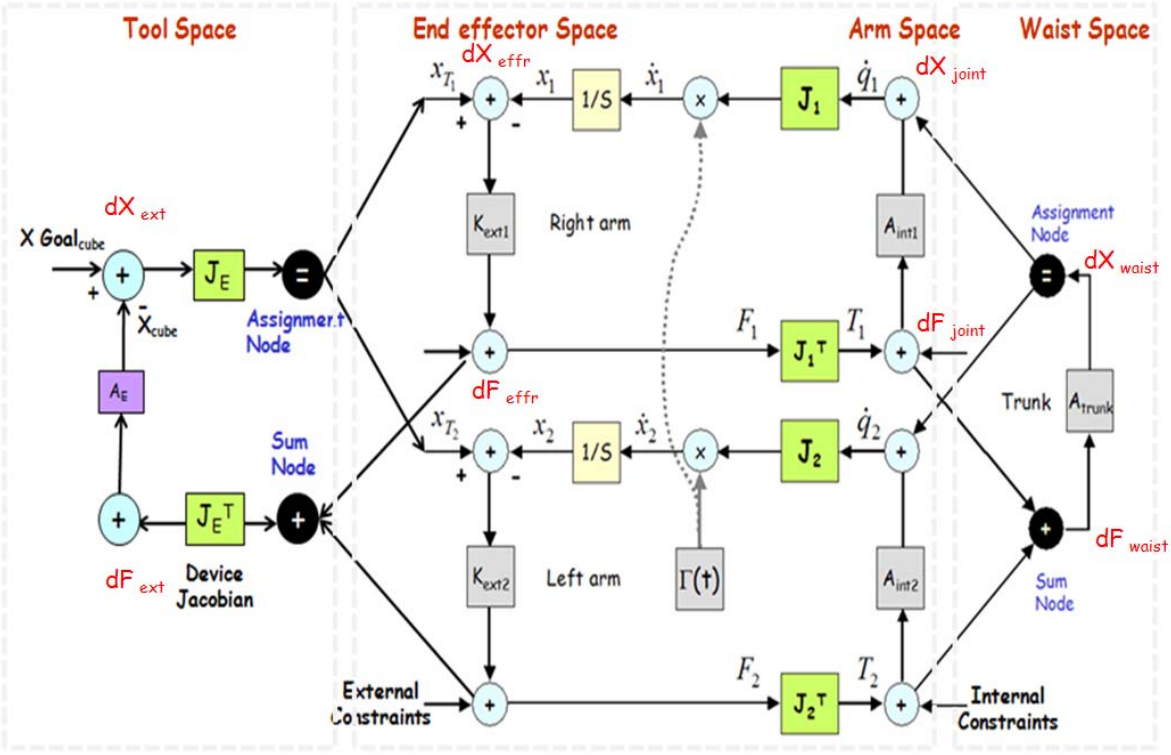


Figure 25. The PMP network that coordinates the transporting phase, with a force field applied to the object and propagated to the PMP sub-networks that correspond to the right arm, left arm, and the waist.

Before we conclude this section, we concisely summarize some general principles that need to be considered while creating any composite PMP chain for coordinating bodies of any arbitrary complexity and redundancy, structurally and functionally coupled with any external object.

(a) **Connectivity in the body**: Analogical to electrical circuits, connectivity in the case of PMP network for coordinating any arbitrary body are of two types:

Serial Connection: Position vectors are added, this is a common force connection

Parallel Connection: Force vectors are added, this is a common position connection. For example, when we push an external object with both arms, the force applied by each arm acting in parallel is added.

In figure 26, the two hands are connected in parallel to the cube, links in each arm are connected in series, muscles are connected in parallel to each link. Waist is connected in series with the two arms.

(b) **Work Units (WU)**: All motor spaces have a pair of generalised force and displacement vectors grouped together as a work unit (as shown in figure 2). As seen in figure 24, nodes dX_{ext} and dF_{ext} form a WU in tool space, nodes dX_{eef} and dF_{eef} form a WU in the end-effector space, nodes dX_{joint} and dF_{joint} form a WU in the joint space, dX_{waist} and dF_{waist} form a WU in the waist space. This fundamental structure of communicating with flows and forces is preserved in all the computational chains described in this chapter. This also implies that the same motor task can be potentially described with representations used by any WU. For example, the task of turning a steering wheel with two hands can be described using a mono-dimensional steering wheel pattern in tool space or a 6-dimentional end-effector space pattern or a 7-dimensional joint rotation pattern or multi-dimensional muscle contraction patterns. Further, the relaxation process achieved by the passive motion paradigm incrementally derives trajectories in all these spaces not through an explicit ill-posed inversion process but through a well-posed direct computation: the passive relaxation to a virtual pull.

(c) **Geometric Causality of the controlled system:** This is expressed by the Jacobian matrixes that form the horizontal links in the computational chain. They link two WU or motor spaces together. The Jacobian determines the mapping of small displacements in one direction and transpose Jacobian determines the dual relation among forces in the opposite direction. In figure 24, the space Jacobian J1 (6 X 10) maps joint rotation patterns dX_{joint} and dX_{waist} (of the right arm and waist) into displacement dX_{eef} of the right arm. The transpose Jacobian $J1^T$ projects disturbance force F1 generated on the right arm chain to corresponding joint torques. The device jacobians J_E forms the interface between the body and the external object and represents the geometrical relationship between the tool and the concerned end-effector. In the case of the steering wheel task, J_E maps the transformation between the rotation of the steering wheel and the corresponding differential displacement seen at the end-effectors (hands). In case of a different tool being coordinated by the two arms, it is only J_E that changes in the computational model.

(d) **Elastic Causality of the controlled system:** These are the vertical links in the computational model and are expressed by stiffness and admittance matrices. They link incremental forces to displacements in each WU. Hooke's law of linear elasticity can be generalised to non linear cases by considering differential variations: dF =K.dX

and dX= C. dF. We must note that even though an elastic element is reversible in nature, in articulated elastic systems, a coherence of representation dictates the direction in which causality is directed.

(e) **Circularity:** All loops in all the computational models described in this chapter are closed and any node is accessible from any other node. So the choice of elastic transformation is constrained by the requirement of circularity. In figure 24, if we enter the network at dX_{eef} and exit at dX_{joint}, we get the following rule for computing incremental joint angles : $dX_{joint} = J_1^T(Kext1.dX_{eef}).Aint1$.

(f) **Sum and Assignment Nodes(+/=):** In complex kinematic structures, where there are several serial and parallel connections, the Sum and Assignment nodes are used to add or assign displacements and forces. They are dual in nature. As seen in figure 24, if an assignment node appears in the displacement transformation between two WU, then a sum node appears in the force transformation between the same WU's. This can be understood as a consequence of conservation of energy between two WU's. Further, sum and assignment nodes can also appear at the interface between the body and the tool, in order to assign/sum forces and displacements from the external object to the end-effectors and vice versa. However, during the process of relaxation, the external object is as much a part of the body itself, and any change in tool space reverberates all though the computational chain (tool-end-effectors-joints-trunk), to bring the whole system (tool + body) to a new equilibrium. In the example of steering wheel rotation task, a small wheel rotation would assign (through an assignment node) motion to the two hands connected in parallel weighted by J_D. The force disturbance is computed for the imposed displacement dx using the stiffness matrix. The resultant force vector determines a torque vector which yields a joint rotation dq via a transpose Jacobian and compliance matrix respectively. Finally, the steering wheel torque is the summed (sum node) contribution coming from the two arms weighted by transpose of the device Jacobian. In sum, the relaxation process allows us to effectively characterize this highly redundant task of bimanual co-operation and incrementally derive the multi dimensional actuator patterns from a mono-dimensional steering wheel rotation plan.

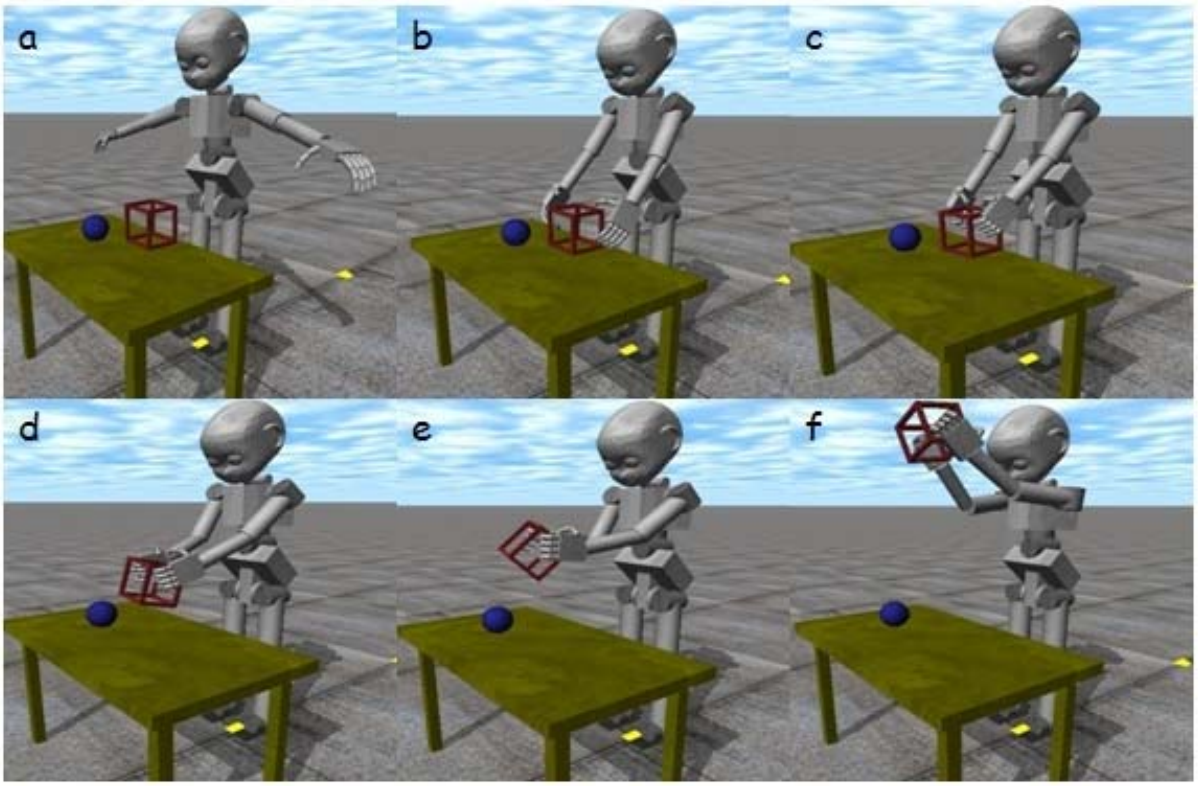


Figure 26. Example of a bimanual transportation task coordinated by the composite forward/inverse model pair shown in figure 24. The different phases involved in the task are reaching (frames a and b), grasping (frame c), and then moving an object to a different spatial target (frames d-f) using both arms.

3.7 Putting all together

Exploiting the solutions offered by the physics of passive motion and the concept of terminal attractors, we presented a simple, distributed computational framework for representing and solving a range of ill-posed coordination problems arising in redundant robotic platforms like the GNOSYS robot and the baby humanoid 'iCub'. Analogous to the coordination of the movements of a wooden marionette by means of attached strings, the computational model presented in 3.2 operates by means of a relaxation to equilibrium of a network of task relevant parts of the body to the virtual pull induced by superimposed force fields, representing the goals and task relevant mixtures of constraints. Further as shown in 3.5, the virtual force fields representing targets and constraints in different spaces are superimposed at runtime to yield a net force field that relaxes the internal model to a final configuration. This solution is computed implicitly by the dynamics of the computational model and represents the best trade-off among the multiple set of constraints relevant at that point of time.

We wish to emphasize that the force fields we are considering in this chapter don't really describe the biomechanical forces at play during the execution of movements, but are computational metaphors that describe the complex dynamics of the internal computational

engine. The internal model associated with the force-fields and represented by means of the neural network described in section 3.2.1 learns a whole family of geometrically possible solutions as a sort of a holographic memory, from which one solution is implicitly selected based on the nature of the task being executed and the pull of the field in the direction of the intended movement. Further, the proposed architecture is also endowed with nice computational properties like robustness, run time optimization, fast task adaptation, interference avoidance and local to global computation that makes it both biologically plausible and extremely useful in the control of complex robotic bodies as we saw in the various implementation examples (Reaching, Stacking, Tool use and orientation, bimanual coordination etc).

Robustness

The robustness of the computational machinery stems from several reasons:

a) No model inversion is needed as the system always operates by means of incremental well-posed, direct computations (for example, a force based transformation as we move from the distal to proximal space, and a position based transformation as we move in the reverse direction);

b) The dynamical system automatically stays away from singular configurations and relaxation to equilibrium in a conservative force field can never crash;

c) The system does not “search” for a solution, as it has access to a whole family of solutions stored in the form of a holographic memory, and “discovers” the one that is closest at hand;

d) Even if the target is outside the reachable workspace, the robot nevertheless tries to approach the target as much as possible by fully extending the arm to a position that is at a minimum distance from the target. Hence, what we see in such cases is a *gentle degradation* of performance that characterizes humans in the same situations. Although there is no exact solution to the problem, the network “does its best”;

e) It is well known that moving one part of the body can generate interaction forces that tend to destabilize other parts of the body (Tcheang & Wolpert 2007). As shown in figure 23, the proposed model is powered with autonomous stability maintenance mechanisms that predict and actively oppose these interaction forces.

Flexibility

Flexibility of the computational machinery is made possible by the following items:

a) There are no pre-defined cost functions/optimization constraints (minimum torque, minimum jerk, signal dependent noise etc.) in the model per se; hence there is a scope

of operating online, facilitating runtime co-evolution of a plans and the corresponding control processes needed to achieve them;

b) Multiple constraints can be concurrently imposed in a task-dependent fashion by simply switching on/off different task relevant force field generators (as shown in section 3.4, 3.7);

c) The same flexibility is also observed in the recruitment of different degrees of freedom afforded by a complex body in the performance of a specific task (as shown in figure 22). Modulating the compliances of different joints will result in different amounts of free motion at that joint and hence determine how much that degree of freedom is willing to contribute to the overall final configuration attained by the system. For example, if the wrist is assigned very high stiffness, the other more compliant degrees of freedom in the kinematic chain will experience larger displacements and bear the burden of reaching the solution and so on. This tuning can also be learnt and, for the case of a far away target that cannot be reached just using the DOF at the arms, the torso becomes more compliant and its DOF contribute to the overall solution.

d) Custom relaxation networks containing many kinematic chains and sometimes linked with external objects like in figure 25 can be composed in a systematic manner based on the task at hand (for example, when using a tool that is coupled to the body) and can be easily used to incrementally derive the motor commands needed to produce the desired coordination. This applies for any combination of closed and open kinematic chains, complex kinematic structures with any arbitrary redundancy and any arbitrary topology.

Local to global computing

From the perspective of local to global computing, at each instance of time, every element in the computational chain of figure 2 makes a (incremental) local decision regarding its contribution to the overall externally induced pull, based on its own compliance. All such local decisions contribute towards driving the system to a configuration that minimizes its global elastic potential energy. Similar to many connectionist models in the field of artificial neural networks, the mechanism to regularize and exploit redundancy by means of attaining configurations that minimize global potential energy essentially uses only local asynchronous interactions. Analogous to auto associative memories like the Hopfield neural network (Hopfield, 1982) that reconstruct a stored memory pattern from a distorted cue by filling up all the missing information during the progression to attain an equilibrium state that minimizes a global computational energy function, a plan in the proposed computational model does not need to specify the behavior of all joints and muscles but only needs to deal with a small number of “end-effectors” or a tool, all missing information automatically filled

in by the incremental updation of the outputs of each node in the computational chain as per the principle of minimization of global elastic potential energy of the system.

Forward/Inverse internal models

An interesting area of research closely related to the model proposed in this paper is the use of forward/inverse internal models, now wide-spread in the field of cognitive science. The existence of neural mechanisms that mimic input output characteristics and the inverse models of the motor apparatus are supported by several behavioral, neuropsychological and imaging data (Miall and Wolpert, 1994; Wolpert and Kawato 1998). A forward model or an emulator is a computational mechanism that captures the forward or causal relationship between the inputs and outputs of a system. If we consider the arm as the target system, the forward model predicts the next state (position and velocity), given an initial state and motor command. The inverse model does the opposite: it takes a goal-state as input and produces a sequence of motor commands necessary to achieve it. It is quite easy to observe that a PMP network, in anyone of the different versions considered in this paper, is an integrated forward/inverse model: (1) a forward motor controller that maps tentative trajectories in the intrinsic (joint) space into the corresponding trajectories of the end-effector in the extrinsic workspace and (2) an inverse motor controller that maps desired trajectories of the end-effector into feasible trajectories in the joint space, concurrently taking into account the motion of the end-effector predicted by the forward model. We also note that unlike forward/ inverse models using supervised neural networks (e.g. Jordan networks: Jordan, 1986), the proposed model using the notion of passive motion paradigm operates by seeking stationary configurations of a non linear dynamical system and is somatotopic in nature.

Mental Simulation

The advantages of having forward/inverse models are numerous, ranging from overcoming transductive and transport delays, canceling sensory re-afference, aiding distal supervised learning, and mental simulation of actions, among others. A key element of the proposed architecture is that the same computational model can be used to support mental simulations possibly employed by sub-symbolic reasoning processes and the actual delivery of motor commands during movement execution, after such reasoning has verified the consistence of the motor plan. This point of view is in agreement with the CODAM concept (Corollary Discharge of Attention Movement, Taylor, 2003). In other words, one can reason about reaching without actually reaching and yet use the same neural/computational substrate to do so. The relaxation of the coupled forward/inverse model pair provides a general solution for mentally simulating an action of reaching a target position taking into consideration a range of geometric constraints (range of motion in the joint space, internal

and external constraints in the workspace) as well as effort-related constraints (range of torque of the actuators, etc.). If the forward simulation is successful, the movement is executed; otherwise the residual "error" or measure of inconsistency can be used to trigger a higher level of reasoning regarding possible availability of a tool that could be used to get closer to the goal. In the forthcoming chapters, we will enter into the details of how abstract reasoning couples with the various forward/inverse models and thereby initiates mental simulations of goal directed sequences of actions.

Reasoning: From animals to cognitive robots

Several studies from animal reasoning suggest that a large circle of animals appear to be involved in some form of prospection and reasoning that involves using/making tools to achieve otherwise unreachable goals (Visalberghi and Tomasello,1997; Weir et al, 2002). Such experiments from animal reasoning form ideal scenarios for developing-validating computational architectures of cognitive behavior in humanoid robots. In a previous work using a simple 5 dof arm and 2 cameras (Mohan and Morasso, 2007), we presented preliminary results of the possible use of the proposed computational architecture coupled with a recurrent neural net to solve the two sticks paradigm form animal reasoning. A humanoid platform like the iCub, further affords the possibility to attempt more complex and challenging scenarios with multiple competing goals, sometimes necessitating tool use/simple tool manufacture (like the case of New Caledonian crows) to realize the goal. Validating such scenarios in robotic embodiments require computational architectures to reason about intelligent exploitation of available degrees of freedom in body and in the environment. Reaching being a fundamental operation in any kind of interaction with the physical world, the forward/inverse model presented in this chapter is a crucial component in the overall three layer computational architecture for robotic reasoning we will present in this thesis. In simple terms, the computational model presented in this chapter (and used for planning/simulating actions at a the detailed level of forces, positions, effort, bodily constraints, task geometry etc) is weakly coupled to a more abstract computational layer that initiates planning at a more general level of goals, rewards, object actions, situation plan (sensorimotor space-action space-work space), which is further coupled to a third layer that involves active intervention of the robot in the world (for exploring, learning and populating its global knowledge space).

From the perspective of linguistics, several proponents of the embodied interactionist theory of meaning have argued that the same neural substrate employed in mental simulation of action is also used in understanding actions of others and for grounding the meanings of verbs (Gallese & Lakoff, 2005; Glenberg, 1997; Feldman & Narayanan, 2004). In other words, understanding a piece of language necessarily entails an internal mental simulation of the described scenario, by activating a subset of the neural structures that

would be involved in perceiving the percepts or performing the motor actions described. There have been many recent neurophysiological and behaviorial studies supporting the simulation theory of language comprehension (Pulvermuller, 2003). A key to grounding the meaning of an action verb to the activity of a vast distributed network of neurons in the forward/inverse model is the idea of parameterization (Feldman et al, 2007). Motor actions such as reach, touch, tap, push, press (and their tens of counterpart verbs) involve many coordinated neural firings, muscle contractions, etc. but we have no awareness of these details and neither can we talk about them. What we are aware of (and capable of describing linguistically) are fixed sets of parameters accompanying the verb, mainly – goal, force, direction, constraints, end-effector, posture, repetition, speed. This view point is very interesting from the perspective of the computational model proposed in this paper because these are exactly the variables that the proposed computational model in this paper controls in a task dependent fashion during both mental simulation and real execution of an action. Hence the crucial hypothesis is that since languages only label the action properties of which we can be aware, and there are a fixed set of embodied features that determine the semantic space for any set of concepts, such as motor actions and verbs, a parameterized version of the proposed computational model could be used for grounding/understanding verbs in a language. In other words, the verbs reach, press, push, touch, tap are grounded on the same mental model of action with only minute changes in the simulation parameters that gives an internal sense of the minute difference in their meanings (and also enables us and possibly our robots to execute them when linguistically commanded). Future research in this direction with regards to computational modeling is expected to provide further insights into how language could be integrated with the motor knowledge in the cognitive system.

Actions II–Computing in the world

The ten thousand things carry the Yin,
and move to embrace the Yang
Through **blending of the material force** Ch'i,
they achieve harmony

TAO TE CHING
(Written around 6 century BC by Taoist sage Lao Tzu)

The ability to navigate in space is an essential component in the behavioral repertoire of all animals, critical for their survival, finding food, mates, and avoiding predators. For centuries humans have been fascinated by the navigation and way finding capabilities of ants, bees, birds, fishes and other primates. Ancient civilizations devised many myths to explain the periodic appearances of different birds in their environments. The fact that many birds can fly all the way from Europe to Africa and back in one year is even harder to believe than the myths devised by the tribes from ancient civilizations. How animals acquire, store, update, and use spatial representations of their environments for localization and navigation is one of the central questions in cognitive science. A large body of neuroanatomical, neurophysiological, and behavioral data acquired from experiments conducted on mammals (primarily rodents), posit involvement of a range of neural systems being involved in spatial memory and planning, like the head direction cells (Blair et al, 1998), spatial view cells (Georges-Francois et al, 1999), hippocampal place cells (O'Keefe and Dostrovsky, 1971) that exhibit a high rate of firing whenever an animal is in a specific location in the environment corresponding to the cell's "place field" and the recently found grid cells located in the entorhinal cortex in rats, known to constitute a mental map of the spatial environment (Hafting et al, 2005). Just like the animals, the GNOSYS Robot also faces exactly similar problems related to learning a mental map of the spatial topology of the playground and use it in coordination with the forward/inverse models for arm in order to mentally simulate sequential goal directed movements of the body and the arm in tricky environmental scenarios. In addition, it also needs to learn the causality of pushing objects in the trapping groove using sticks.

Once again using powerful field computing principles applied on a growing neural gas, we describe how internal models for spatial topology and pushing can be learnt by GNOSYS through self organization of randomly generated sequences of sensorimotor data. During the course of discussion we also present experimental results from the robot's behavior that

touch several important issues like representations, goals, sub goals, optimality, contradictions, reinforcements, causality and rationality.

4.1 The 'How do I get there' problem

The GNOSYS playground is a 3×3m enclosure with goal objects placed at different locations on the floor and on the centrally placed table as seen in figure 10 (of chapter 2). In the last chapter, we presented a forward/inverse motor control architecture for executing/mentally simulating a reaching action directed at a goal object taking into account a range of task dependent mixtures of constraints represented in terms of superimposed force fields. In addition to arm/tool movements that we considered in the previous chapter, all experimental scenarios and user goals discussed in chapter 2 further require autonomous navigation and way finding capabilities in the environment. This requires the robot to be autonomously able to deal with issues like 'where am I', 'where are the other objects relative to me in the playground' and 'how do I get there in ways such that different task related constraints are satisfied'. Reasoning further necessitates creation of a mental map of the spatial topology and an internal model for pushing action, that operate in close coupling with the internal model of the arm in order to perform a range of goal related 'what if' experiments in the mental space. In simple terms, we must eventually make different internal models representing space, body and physical causality speak with each other in order to be able to have any realistic chances of generating flexible behaviour in the environment.

One key feature regarding various internal models (arm, spatial map, pushing, abstract reasoning) presented in this thesis is the fact that all of them are structurally and functionally identical, use the same protocols for acquisition of information, same computational mechanisms for planning, adaptation and prediction. The only difference is that they operate on different sensorimotor variables, move in the presence of different value fields, towards different goals (local to their computational scope), using different resources of the body/environment. This level of self similarity in the basic functionality of all the computational models, is also highly advantageous during the development of the software architecture of the system. It allows us to group all the learning, planning , prediction related software processes as a 'super structure' that is just called by different internal models during their executions. The obvious advantage is reduction in complexity of the software, efficient code reuse, flexibility in learning different things using different body resources while still using the same software and finally a high level of abstraction in the architecture. The proposed computational machinery brings together several interesting old and novel computational concepts, with particular reference to the following ones:

a) The theory of self organizing maps (von der Malsburg, 1973; Willshaw & von der Malsburg, 1976; Kohonen, 1995), their extensions with regards to growing networks

(the growing neural gas model of Fritzke, 1995) to grow and represent the different sensorimotor state spaces;

b) Neural field dynamics (Amari, 1977; Erlhagen and Schoner, 2002) to organize goal directed planning , prediction, tracking in the various state spaces;

c) The interesting concept of sensorimotor maps of Marc Toussaint to dynamically couple sensor representations with motor representations during prediction and planning (Toussaint et al, 2006) . We suitably extended the dynamics in the sensorimotor space in order to dynamically switch between exploration and exploitation based on contradictions between the top down (mentally simulated) and bottom up information (real sensory);

d) Some intuitive heuristics regarding distribution of (delayed) rewards to neuronal populations in the appropriate internal models, that contributed towards the overall plan that led to the robots success/failure in the task it attempted;

e) Temporal Hebbian learning (Dayan, P., & Abbott, L., 2001) to change the internal model (and lateral connectivity between neurons) in response to dynamic changes in the world detected by the robot;

f) Learning different value fields of experience through exploration which modulate the behavior based on their relevance to the active goal being realized by the robot.

We also note that the forward/inverse models presented in the previous section were used to generate actions related to pushing during the explorative phase of learning the internal model related to pushing i.e. the sequence of reaching and grasping a stick, reaching a goal object with the stick held in the gripper and then randomly pushing the object in different directions, at the same time tracking the object through vision. The spatial exploration is triggered through random speed set commands, at the same time tracking the body position/orientations through the localizer, and using the infrareds to detect obstacles like the tables (which are not sensed by the lasers) and use of the vetoing scheme in the Gnosys Agent (section 2.1) to arbitrate the operation of the different functionalities in case of conflicting requests. In the next few subsections, we will describe different aspects of the computational model in detail taking the examples of space related computations to navigate in the playground and pushing of objects in the trapping groove, while at the same time presenting several experimental results on the resulting behavior of the robot in different environmental settings.

4.2 Representation: on self organizing sequences of sensorimotor data

Figure 1 shows the general computational structure of the all the internal models described in this and the forthcoming chapter. The central element of the architecture is a growing

intermediate neural layer common to both perception and action, called the sensorimotor space (SMS).

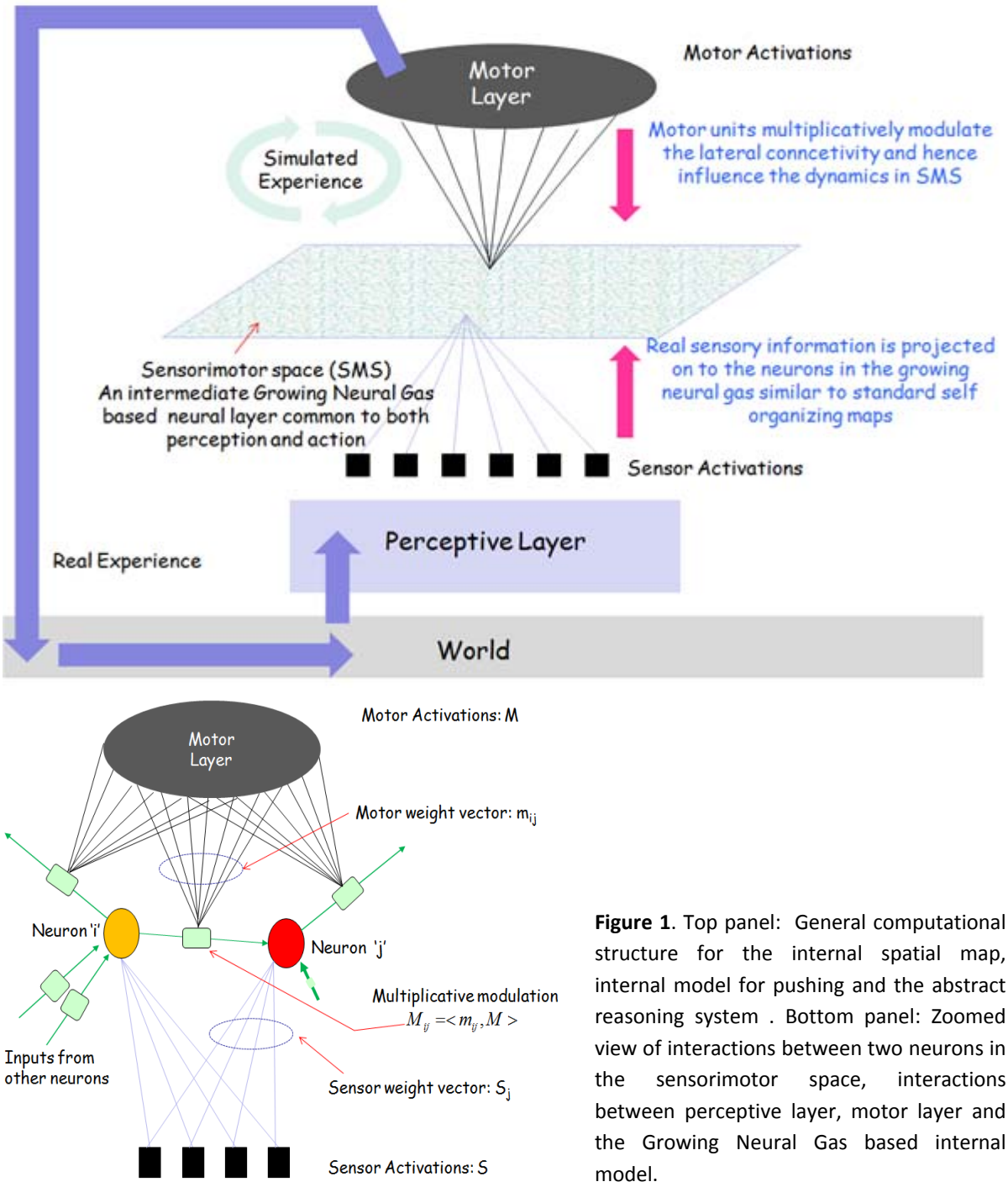


Figure 1. Top panel: General computational structure for the internal spatial map, internal model for pushing and the abstract reasoning system . Bottom panel: Zoomed view of interactions between two neurons in the sensorimotor space, interactions between perceptive layer, motor layer and the Growing Neural Gas based internal model.

This neural layer not only self organizes sequences of sensorimotor data generated by the robot through random motor explorations (through the loop of real experience) but also sub symbolically represents the forward inverse functions of various sensorimotor dependencies

(that is encoded in the connectivity structure where ever and when ever learning takes place). Further, it also serves as a proper computational substrate to realize goal directed planning (using quasistationary value fields) and perform 'what if' experiments in the mental space (through the loop of simulated experience shown in figure 1). Both the SMS and the complete lateral connectivity structure is learnt from Zero using sequences of sensor and motor data generated by the robot. Neither the number of neurons, nor what they represent is pre-decided. Even though theoretically there is also no need to predecide the sensorimotor variables that will be in play in each internal model (ideally all sensorial and motor DOF can be involved in every SMS) while working on the robot we predefined the set of sensorimotor variables that will be in involved in the exploration phase to grow the SMS. This is important in order to avoid unexpected accidents with the robot, for example even though an object could be potentially pushed with the robots body and if allowed the robot may eventually learn this association, pushing in our case was predecided to be carried out with the DOF involved in the arm (and using sticks if needed).

Provided that the world is consistent, both mental simulation (top down through motor modulated lateral connections) and real performance (bottom up through self organizing competition) should activate the same neural population in the SMS, the coherence between them forming the basis for the stability of the sensorimotor world of GNOSYS robot.

This level of circularity and complementarily between perception, action and imagination inherent in the computational machinery of figure I, also resonates very well with compelling evidence from several neuro-psychological, electrophysiological and functional imaging studies, which suggest that much of the same neural substrates underlying modality perception are also used in imagery, we are able to activate motor structures of the brain in a ways that resembles activity during a normal action but does not cause any overt movement and a simulated action can elicit perceptual activity that resembles the activity that would have occurred if the action had actually been performed.

In addition to incrementally self organizing the state space based on incoming sensorial information (like a standard GNG), the motor information is also fully integrated with the SMS at all times of operation. As seen in figure 1b, motor units project to lateral connections in between the neurons in the GNG (Toussaint et al, 2006) and influence the dynamics of the SMS. This allows motor activity to multiplicatively modulate these lateral connections hence cause anticipatory shifts in neural activity in the SMS similar to that which would have occurred if the action was actually performed. Moreover, provided that the world is consistent, both mental simulation (top down through motor modulated lateral connections) and real performance (bottom up through self organizing competition) should activate the same neural population in the SMS,

the coherence between them forming the basis for the stability of the sensorimotor world of GNOSYS robot.

This level of circularity and complementarily between perception, action and imagination inherent in the computational machinery of figure 1, also resonates very well with compelling evidence from several neuropsychological, electrophysiological and functional imaging studies, which suggest that much of the same neural substrates underlying modality perception are also used in imagery; and imagery, in many ways, can 'stand in' for (re-present, if you will) a perceptual stimulus or situation (Zattore et al 1993, Berhmann 2000, Fuster 2003). Studies show that imagining a visual stimulus or performing a task that requires visualization is accompanied by increased activity in the primary visual cortex (Le Bihan et al 1993, Kosslyn et al 1993). The same seems to be true for specialized secondary visual areas like fusiform gyrus, an area in the occipito-temporal cortex, is activated both when we see faces (Kanwisher, et al 1997) and also when we imagine them (O'Craven, K.M. 1996). Lesions that include this area impair both face recognition (Damasio, A.R. *et al 1990*) and the ability to imagine faces (Young et al). Brain imaging studies also illustrate heavy engagement of the motor system in mental imagery i.e. we are able to activate motor structures of the brain in a ways that resembles activity during a normal action but does not cause any overt movement (Rizzolati et al 2001, Parsons et al 2004, Grush 2004) and further, a simulated action can elicit perceptual activity that resembles the activity that would have occurred if the action had actually been performed (Metzinger and Gallese 2003, Hesslow, G, 2002). EEG recordings on subjects performing mental rotation tasks have revealed activation of premotor and parietal cortical areas, indicating that they may be performing covert mental simulation of actions by engaging the same motor cortical areas that are used for real action execution (Williams). fMRI studies have similarly found activation of the supplementary motor area as well as of the parietal cortex during mental rotation (Cohen et al 1996). Similar results have also been obtained from experiments that involve auditory imagery of melodies that activates both the superior temporal gyrus (an area crucial for auditory perception) and the supplementary motor areas. To summarize, there is enormous evidence suggesting that imagining perceiving something is similar to actually perceiving it, only difference being that, the perceptual activity is generated by the brain itself rather than by external environmental stimuli. Similarly the areas of cortex used in movement control also have a role in motor imagery. Further, mentalization also known to affect the autonomic nervous system, the emotional centers and the body in same way as actual perceptual experiences.

We now outline the process of learning the SMS to represent the spatial topology of the GNOSYS playground. Since all wheels of the robot steer at the same angle (the direction of travel or, equivalently, the orientation of the robot), and the robot moves on a x–y plane in the playground, the robot has three sensorial states of freedom 'S' (its global location in the playground in x–y coordinates and orientation).The localization system of the GNOSYS

robot (Baltzakis et al, 2007) computes this sensory information based on the range measurements from the laser scanner (additionally using information about the robot's action at time t measured by the odometry encoders) and communicates this information to the perceptive layer shown in figure 1a.

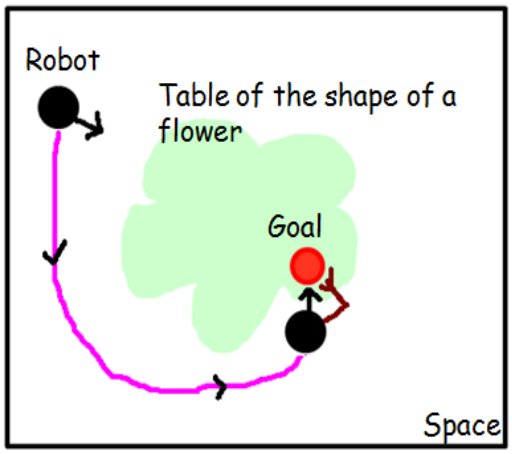


Figure 2. Obstacles are implicitly represented in the internal spatial map.

The motor signal 'M' is two dimensional representing the translation (meters) and rotation (in radians) initiated by the robot (and appropriately converted into speed set commands communicated to low level hardware) during very time step of random exploration. The infra red sensors that detect obstacles like the tables (that are not tall enough to be detected by the lasers) don't directly become involved with the perceptive layer, but influence the learning of the internal model indirectly by vetoing to inhibit the action (translation/rotation) taken by the robot, in case a threat due to obstacle is detected. This has three major advantages a) obstacles in the environment need not be pre programmed (for example using repulsive fields etc that is very common in literature) b) dynamic changes in the world, like a new object placed somewhere in the play ground can be detected, and a learning phase to adapt the internal spatial map so as to represent the modified world can be initiated c) obstacles can be of any geometry, tables can be of any shape. This flexibility arises from the fact that obstacles are not explicitly represented in the spatial map, What is learnt is the free space where motion is possible and whatever remains independent of its geometry is an obstacle in the playground. This way of implicitly representing the tables (and other obstacles) also further solves the important problem of approaching an object placed in the table in order to reach or grasp it. The difficulty in computational terms can be imagined by considering a table in space of the shape of a flower (as in figure 2) and the robot has to position its body in the most appropriate manner around this table so as to grasp a ball placed on of its edges. In our case there is no need to add any affordance related algorithm into the computational model to achieve this, the equilibrium achieved by the dynamics of the SMS of the spatial map automatically finds the best possible way to move and position the body (taking into account other constraints like energy, shortest path etc) so as to eventually grasp the ball.

During the process of learning the GNG, the simulated experience loop is turned off (we will enter into this issue in the next subsection related to dynamics in the SMS). In other words, the only loop active in the system is the loop of real experience. The agent is now allowed to move randomly in the play ground with a maximum translation of 14 cms and maximum rotation of 20 degrees in one time step (in order to achieve the representational density necessary to perform motor tasks in future that require high precision). These movements generate the data i.e. sequences of sensory and motor signals $S^{(t)}$ and $M^{(t)}$ using which both the sensory weights, lateral connections between neurons and the motor weights of the motor modulated lateral connections are learnt.

Considering the fact that initiating random body movements with a robot of the size and complexity of GNOSYS and in an environment of $3m^2$ area is expensive in terms of time and energy, this phase of data generation was carried out in the virtual simulation environment of the robot, that has identical hardware and software interfaces like the robot. After this phase of initial training, the complete architecture can be used to drive the real robotic platform. Any subsequent adaptations in the internal model due to dynamic changes in the world from now could be possibly done online using the real robotic platform (since these phases of learning are short and inexpensive mainly in terms of battery power drawn during navigation). We also fundamentally reject any concerns regarding use of the virtual simulation environment for simplifying the cumbersome training of the spatial map, considering the fact that we never encountered any major computational problems, degradation in performance once the trained system was incorporated on the real robotic platform. Further all subsequent adaptations to the internal model can take place on the robot itself and are directly dependent on changes made by the user in the real world. The only reason for explicitly mentioning this fact is because this problem ultimately leads to the question as to whether physical embodiment is necessary in order develop a computational theory of cognition. From our experience in the last few years on working with both the virtual simulation environment as well as the real robotic platform, we feel that at the level of computation and information processing, it does not make a difference if you are working with a real robotic platform acting in a real world or an equivalently

> This however does not mean that we out rightly negate the need of a physical instantiation. We merely take the position that suggests to exploit the best of the two worlds, 'real' and the 'virtual', use the virtual simulation platform to run tasks that are energetically (and temporally) cumbersome to perform with the robot in the real world and use the real robotic platform to then learn dynamic changes in a world that is not constant, physically interact with various objects using the arm and perform challenging user goals in complex environments (since using a physical instantiation in turn saves us from the burden of simulating the dynamics of the complex world).

complex virtual simulation of the robot situated in a identically complex world. This however does not mean that we out rightly negate the need of physical instantiation. We merely take a position that suggests to exploit the best of the two worlds, 'real' and the 'virtual', use the virtual simulation platform to run tasks that are energetically (and temporally) cumbersome to perform with the robot in the real world and use the real robotic platform to then learn dynamic changes in a world that is not constant, physically interact with various objects using the arm, perform challenging user goals in complex environments (since using a physical instantiation saves us from the burden of simulating the dynamics of the complex world).

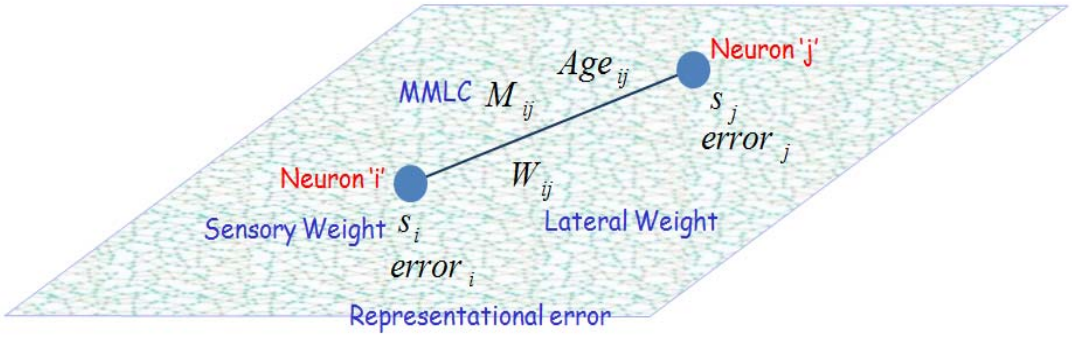


Figure 3. Free variables that need to be learnt in this phase of self organization.

Below we summarize the protocol we employed to grow the SMS in all the internal models using the respective randomly generated sequences of sensor 'S' and motor data 'M'. We applied the standard techniques for self organization of representational maps based on the works of Kohonen (), their extensions with respect to growing networks Fritzke(), Carpenter () and taking into account issues related to the motor coupling (Toussaint et al 2006, 2004). The free variables that need to be learnt in this phase are as follows:

1. *N*: No. of neurons in the sensorimotor space (N);
2. s_i : sensory weights for each neuron (N X D_{Sensor}), these are randomly initialized;
3. $error_i$: local estimate of representational error (useful information for growing) (N)
4. Age_{ij} : Age of lateral connection for pruning off excess and less valuable lateral connections in the GNG (N X N). Age of all connections are initialized to zero in the beginning.
5. W_{ij}: Lateral weights (these are edges that encode neighborhood, possible state transition, permit spreading of activity in the direction of the gradient of value field, locally adapted in response to dynamic changes in the world) (N X N).

6. M_{ij} : how current motor actions influence shifts in activity in the sensorimotor space(N X N X M).

D_{Sensor}: DOF in the sensory space and M: DOF in the motor space respectively.

All adaptation mechanisms are standard. Learning of these free variables progresses in the manner that is briefly outlined below:

- We start with one single neuron with randomly initialized sensory weights.
- Then a random motor activation M^t is generated and the incoming sensory information S^t is observed. In the case of formation of the spatial map the motor activation is two dimensional representing the a random translation (maximum of 14 cms in one time step) and random rotation (maximum of 20 degrees in one step). The sensory information is the global position of the robot (x(t), y(t)) coming from the localizer. In the case of pushing the motor activations correspond to the random incremental change in degrees of freedom of the katana arm (that determines the direction of push) and the location of the tool with respect to the object being pushed (that determines the displacement of the object as a result of the pushing action). The sensory information is the location of the object after the pushing action which is a result of a visual scene analysis of the new images grabbed by the cameras and appropriately reconstructed to 3D space using the computational model presented in 3.4.1. For the case of abstract reasoning system that we consider in the next chapter, the sensor and motor states themselves are more abstract concepts representing situations, plans etc;
- Of all the neurons that exist in the SMS at that point of time, we then have to find the neuron 'i' that shows maximum activity for the observed sensory stimulus S^t at time t. This also implies that neuron 'i' sensory weights S_i such that $||(s_i\text{-}S^t)^2||$ has the smallest value, among all neurons existing in the SMS at that instance of time;
- New neurons are incorporated into the SMS when the difference between the actual perception and the best matching unit say 'i' becomes too large. To make this detection more robust, we assume that every neuron in the SMS has a measure of its own local representational error, that accumulates with respect to time. For this purpose we use a low pass filter as in equation 1

$$\tau_e \dot{error}_i = -error_i + (1 - \frac{1}{\sqrt{2\pi}\sigma_e} e^{\frac{-(S_i - S)^2}{2\sigma_e^2}}) \quad (1)$$

Whenever this error measure exceeds a threshold called vigilance, $error_i > v$ (in our case v=0.25), we generate a new neuron j with the codebook vector equal to the

current perception, and a lateral connection from neuron 'k' that was the winner at time t-1, with the motor weight equal to the current motor signal $m_{jk}=M^t$. Finally, local errors of both neurons 'i' (that was originally the best match) and 'j' (the newly added neuron) are reset to zero. The local error is a statistical measure and nodes that cover a larger portion of the input distribution will have a faster growing error than other nodes, statistically. Large coverage is equivalent to larger updates of the local error, since inputs at greater distances will be mapped to the node. Since we want to minimize the errors, knowing where the error is large is useful when growing new neurons.

- We now adapt the sensory weights of the Winner and its topological neighbors (all neurons laterally connected to the winning neuron) by small fraction as follows.

$$s_i \longleftarrow\!\!- s_i + e_w(\overline{S} - \overline{s_i})$$
$$s_n \longleftarrow\!\!- s_n + e_n(\overline{S} - \overline{s_n}), \forall n \in Neighbours(i) \quad (2)$$

While setting e_w and e_n too high usually results in an unstable network, with nodes moving all around all the time (as shown in zoomed view in figure 4), setting them too low often makes training slow and ineffective. In all our experiments we choose the following values: e_w=0.04 and e_n=0.0006 .

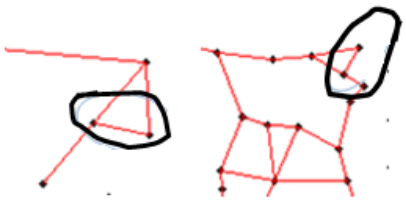


Figure 4. Node Movement during learning of the SMS.

- Next, we adapt the motor weights for all the neurons such that a motor weight m_{ij} between neurons i and j learns on average, the motor signals that contribute to a transition from stimulus s_j to s_i. This can be achieved in a Hebbian way using a simple rule (Toussaint et al, 2006) given as follows

$$m_{ij}^T = \frac{1}{\sum_{t'=1}^{T} \alpha_{ij}^{t'}} \sum_{t=1}^{T} \alpha_{ij}^t M^t \quad (3)$$

where T is the current time step of exploration, α_{ij}^t determines how well a transition from stimulus s_j to s_i at best represents the executed motor action M^t at time t. The simplest realization of this rule is to make $\alpha_{ik}=1$ if neuron k is the winner in SMS at time t-1 (it's the presynaptic neuron) and i is the winner in SMS at time t. In simple terms, equation 3 is a weighted averaging rule that approximately makes the motor

weight m_{ij} represent the motor activity M that is needed to generate shift in activity from neuron j to neuron i in the SMS. In case the inverse mapping is straight forward, this training can also be done once the GNG is stabilized, by tuning the motor weights of neighboring neurons (laterally connected) in the GNG to the average motor signals that needs to be generated in order to cause a transition between them.

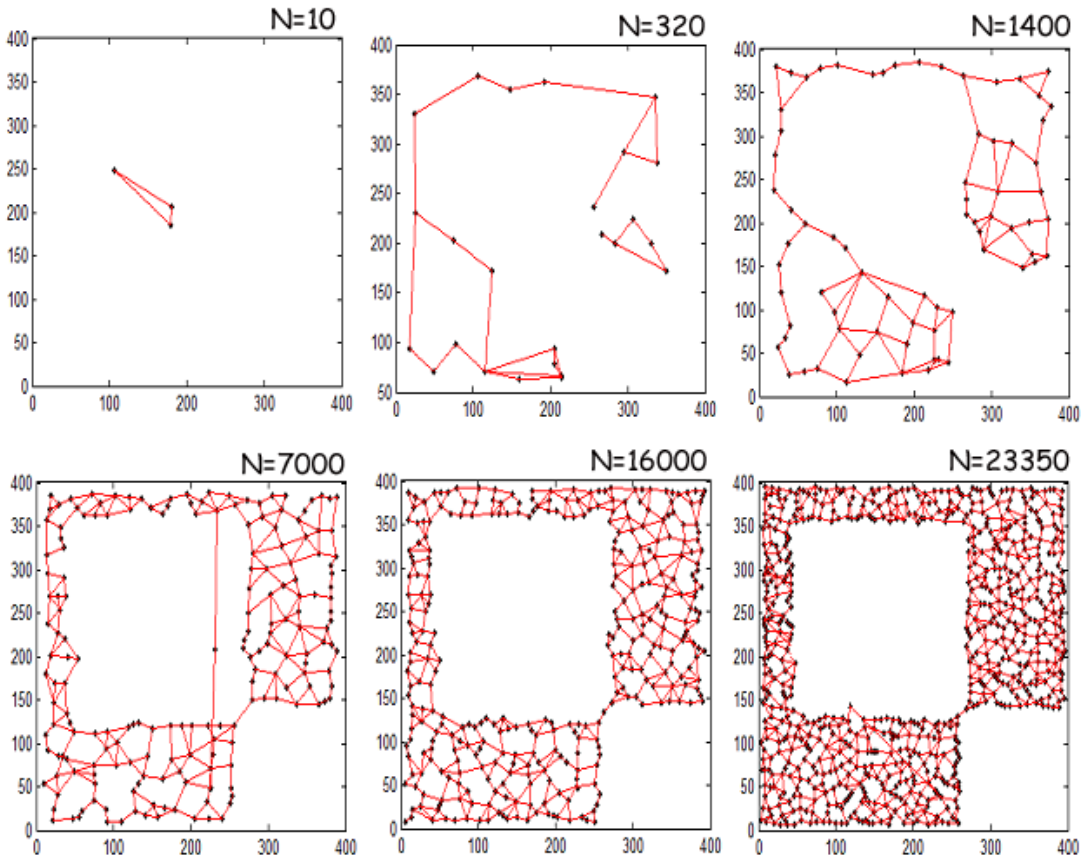


Figure 5: Progression of Growing Neural Gas in the GNOSYS Playground. Panel 1f shows the lateral topology of the SMS after 23350 iterations of self organization after which the map becomes almost stationary. World Model learnt from Zero by random exploration. Starting with two neurons, the number of neurons N grows with time, what they represent changes with time. (In the figure N=710, half million lateral weights and a million MMLC).

- We then proceed to the lateral weights W_{ij} in the SMS. Lateral weights are simply edges between neurons that encode neighborhood, possible state transitions, permit spreading of activity in the direction of the gradient of value field and are locally adapted in response to dynamic changes in the world. The simplest mechanism to organize the lateral weights in the SMS is to use the technique of Fritzke (Fritzke et al, 1995) that involves growing a lateral connection between successive best winning neurons 'k' and 'i' with a lateral weight initialized as $w_{ik}=1$, incrementing the age of all other neighboring lateral connections, and finally pruning off the connections

whose age cross an age threshold Age_{max} (in our case equal to 25). In the subsequent sections, we will describe how lateral weight's can be adapted further based on temporal Hebbian rule, usually when the world dynamically changes.

We finally eliminate the dead neurons (with no lateral connections) existing in the system and proceed with the next step of random motor activation, sensory input observation and another incremental phase of learning the free variables in the system using the procedure mentioned above. As newer regions in the sensorial space are explored through random

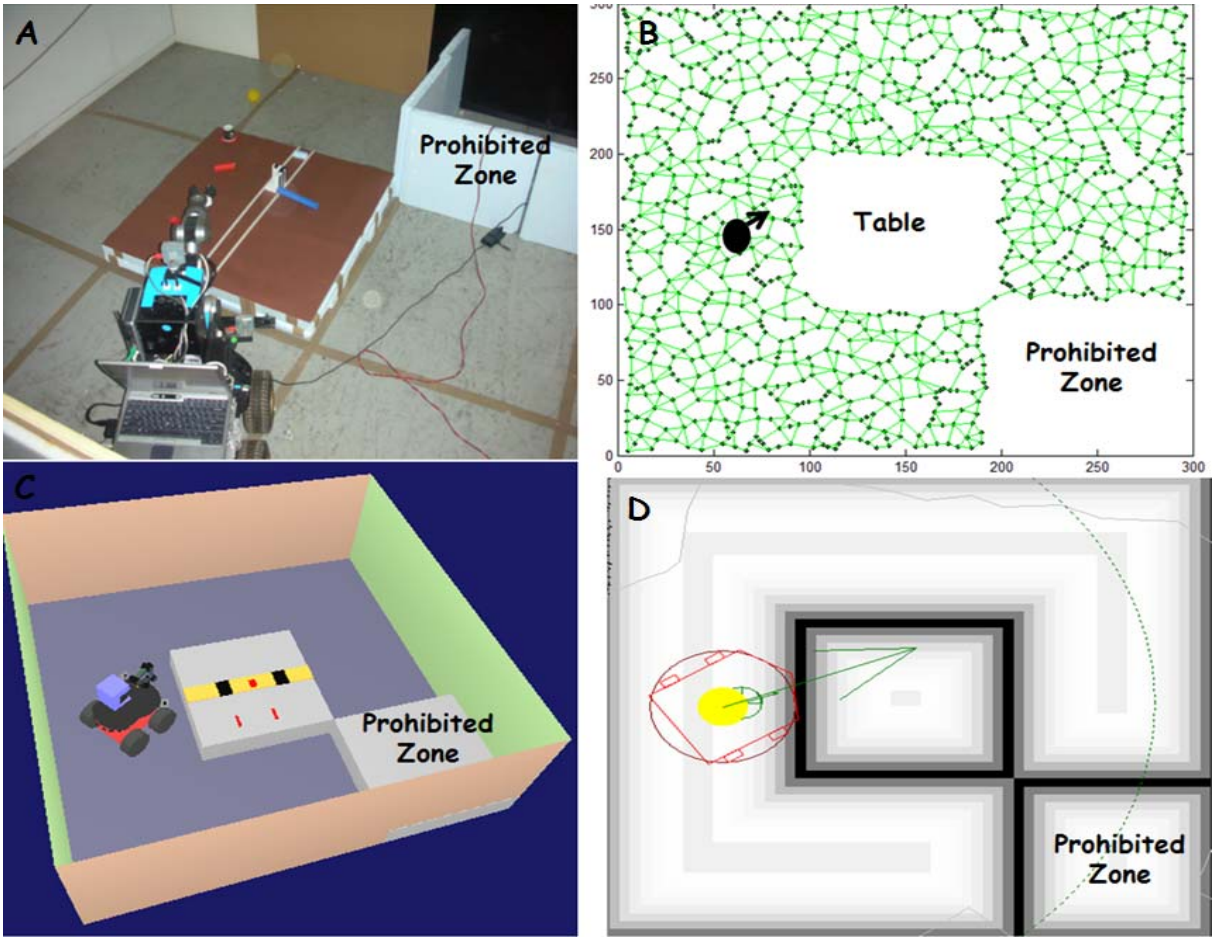


Figure 6. Panel A Shows the revised environmental set up, every square marked is of area $1m^2$; Panel B shows the lateral topology of the sensorimotor space of the spatial map after the initial phase of self organization on sequences of sensory motor data. Panels C and D show the virtual reality environment of the set up.

motor activations, the internal map initially grows more and more and becomes more densely connected. In the later stages, excess lateral connections are pruned of to give a Voronoi graph like appearance. This process continues till the time the internal map becomes almost quasi stationary.

Figure 5 shows the lateral topology of the sensorimotor space in the initial version of the GNOSYS playground, during the progression of this initial phase of random action generations, data acquisition and learning. After an initial set of experiments in this

environment, we switched to the newer environmental layout that was described in chapter 2 (the only major difference between the two environments is in their dimensions, the newer one is a 3×3m enclosure while the older one was a 4×4m enclosure with larger tables). While switching to the newer environment, we retrained the robot to develop the new spatial map. Figure 6 shows the lateral topology of the SMS after training in the new environment.

4.3 Dynamics: On moving in the sensorimotor space

After learning the sensorimotor space, lateral connectivity and the motor code books, we now deal critical issues related to exploitation of the representational scheme outlined in the previous sub section to move neural activity back and forth in the sensorimotor-action spaces, realize goal directed planning, accommodate multiple task specific constraints in the planning process and adapting the internal model to dynamic changes in the world. We begin the discussion with an overview of the dynamical behavior of each neuron in the sensorimotor space. Activation dynamics on sensorimotor space mainly organizes the processes of tracking, planning, prediction, novelty and plasticity in the different internal models. To every neuron 'i' in the SMS we associate an activation x_i governed by the following dynamics:

$$\tau_x \dot{x}_i = -x_i + S_i + \beta_{if} \sum_{i,j} (M_{ij} W_{ij}) x_j \tag{4}$$

We observe that the instantaneous activation of a neuron is a function of three different components. The first term induces an exponential relaxation to the dynamics (and is analogous to spatially homogenous neural fields of Amari et al, 1977). The second term is the net feed forward (or alternatively bottom up) input coming from the sensors at any time instant. The Gaussian kernel compares the sensory weight s_i of neuron i with current sensor activations S^t.

$$S_i = \frac{1}{\sqrt{2\pi}\sigma_s} e^{\frac{-(S_i - S)^2}{2\sigma_s^2}} \tag{5}$$

Finally, the third term represents the lateral interactions between different neurons in the SMS, selectively modulated by the ongoing activations in the motor space. Hence, through this input the motor signals can couple with the dynamics of the SMS. If M is the current motor activity, and m_{ij} the motor weight encoded in the lateral connection between neuron i and j, the instantaneous motor modulated lateral connection M_{ij} between neurons i and j is defined as

$$M_{ij} =< m_{ij}, M > \quad (6)$$

The instantaneous value M_{ij} i.e. the scalar product of motor weight vector m_{ij} with the ongoing motor activations M keeps changing with the activity in the action space and hence influences the dynamics of SMS. Due to this multiplicative coupling, a lateral connection contributes to lateral interaction between two neurons only when the current motor activity correlates with the motor weight vector of this connection. Inversely, by multiplicatively modulating lateral interactions between neurons in the SMS as a function of the motor activity in the action space, it is possible to predict the sensorial consequences of executing a motor action. Interaction between action space and sensorimotor space by virtue of motor modulated lateral connectivity thus embeds 'Situation-Action-Consequence' loops or Forward Models into the architecture and offers a way of eliciting perceptual activity in the sensorimotor space, similar to that which would have occurred if the action was performed in reality.

The element β_{if} in equation 4 is called the bifurcation parameter and is defined as follows

$$\beta_{if} = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(S_{Anticip}-S)^2}{2\sigma^2}} \quad (7)$$

This parameter basically estimates how closely the top down (predicted) sensory consequence $S_{Anticip}$ of any incremental motor action M correlates with the bottom up (real sensory information) S. $S_{Anticip}$ can be easily computed by only considering the effect of top down modulation in equation 4 and finding the neuron 'k' in the SMS that shows maximum activation x_k among all neurons.

$$x_k = \sum_{k,j} (M_{kj} W_{kj}) x_j, \ for\ all\ k, j \in (1, N) \quad (8)$$

Since sensory weights of every neuron is approximately tuned to the average sensory stimulus for which it was the best match, the anticipated sensory consequence $S_{Anticip}$ is nothing but the sensory weights of the neuron k that shows maximum activation the under effect of top down modulation. The bifurcation parameter hence is a measure of the accuracy of the internal model at that point of time. $\beta_{if} \longrightarrow 0$ implies that the internal model is locally inaccurate or there is a dynamic change in the real world i.e. 'the world is working differently in comparison to the way the robot thinks the world should be working'.

What should the robot do when it detects the fact that the world is functioning in ways that are contrary to its anticipations? The best possible solution is to work on real sensory information and engage in a incremental cycle of exploration to adapt the sensorimotor space, learn some new lateral connections, grow new neurons and eliminate few neurons

(using the scheme outlined in 4.2). This flexibility is incorporated in the dynamics in the following fashion:

As we can observe from equation 4 that as $\beta_{if} \longrightarrow 0$, automatically the top down contribution to the dynamics also gradually decreases, in other words the system responds real sensory information only. Hence in this case only the real experience loop (of figure 1) is functional in the system. Now comes the next problem of how to trigger motor exploration dynamically, and this is the third important function of the bifurcation parameter. The bifurcation parameter controls the gradual switch between random exploration and planned behavior, by controlling the amount of randomness in motor signals in the dynamics of the action space as evident in equation 9.

$$\bar{a} = \beta_{if} (\sum_{i=1}^{N} x_i \overline{m_{k_i i}}) + \zeta(\bar{r}) \quad (9)$$

We also note that X_i in equation 4 are time dependent activations and the dot notation $\tau_x \dot{x}_i = F(x)$ is algorithmically implemented using an Euler integration step:

$$x(t) = x(t-1) + \frac{1}{\tau_x}(F(x(t-1))) \quad (4a)$$

In sum, a consequence of the dynamics presented in this section is that at all times, information flows circularly between the sensorimotor space and the action space. While the current goal, connectivity structure and the activity in the sensorimotor space project upwards to the action space and determine incremental motor excitations that are needed to realize the goal, motor signals from the action space influence top down multiplicative modulations in the lateral connections of the sensorimotor space hence causing incremental shifts in the perceptual activity. In the next section we will describe the representational scheme described in the previous section and the dynamics described in this section serve as a general computational substrate to realize goal directed planning (in simple terms, the problem of how goal couples with the knowledge and influences the dynamics of the SMS).

4.4 Goals: On what causes movements in the sensorimotor space

In addition to the activation dynamics presented in the previous section, there exists a second dynamic process that can be thought as an attractor in the sensorimotor space that performs the function of organizing goal oriented behavior. The quasi-stationary value field *V* generated by the active goal together with the current (nonstationary) activations x_i allow the system to generate motor excitations that lead toward the goal. Unlike classical reinforcement techniques in which Q (Quality Matrix) is iteratively derived by exploration in an environment with defined reward structure, we do not assume that the reward structure is known or even constant. Further, rewards are not just functions of state space (like

standard techniques), but also depend on the motor actions initiated from the state. Value field dynamics acting on the sensorimotor space is defined as follows:

$$\tau_v \dot{v}_i = -v_i + R_i + \gamma (W_{ij} v_j)_{\max} \tag{10}$$

$$R_i = DP + Q \tag{11}$$

Let us assume that the dynamical system is given a goal G that corresponds to reaching a state s_G in the sensorimotor space. Just like the sensory signals couple with the neurons in the SMS through feed forward connections, the motor signals couple with the neurons in the SMS through motor modulated lateral connections, the goal G couples with the SMS by inducing reward/value excitations in all the neurons. As seen in equation 10, the instantaneous value v_i of the i^{th} neuron in the SMS at any time instance, is a function of three factors 1) the instantaneous reward 2) the contribution of the expected future reward, where γ (approx 0.9) is the discount factor and 3) the lateral connectivity structure of the SMS. The value field dynamics on the SMS performs the similar computations like those done by the value based iteration methods in classical reinforcement learning approaches(Sutton and Barto, 1998), but also takes into account the lateral connectivity between neurons in order to establish the value. This implies that any adaptation that takes place on the lateral connectivity will also influence the value function and hence influence the behavior. Equation 11 shows the general structure of the instantaneous reward function we used in our computational model. The first term in the reward equation DP expresses the default plan if available (for example, take the shortest, least energy path in the case of the spatial map). We will see in the later sections that it is in fact not really necessary to have a default plan in the reward structure and further there can be situations where new reward functions must be learnt by the system in order to initiate flexible behavior in the world. The second element in the reward function models these additional Goal dependent qualitative measures in the reward structure that are learnt through user/self penalization/rewards.

To keep things simpler in the initial phase of the discussion just to get an overall view of how the connectivity, goals and dynamics co-operate, we will not consider the involvement of the Q term in the reward structure for the time being. Further, coming back to the problem of reaching spatial goals using the spatial sensorimotor map we learnt in section 4.2, let us consider that the spatial goal induces a reward excitation

$$R_i = \frac{1}{Z} e^{\frac{-(s_i - G)^2}{2\sigma_R{}^2}} \tag{12}$$

on every neuron in the SMS (similar to Toussaint et al 2006), where s_i is the sensory weight of the i^{th} neuron, and G is the spatial goal in the playground that has to be reached by the robot and Z is chosen such that $\sum_i R_i = 1$.

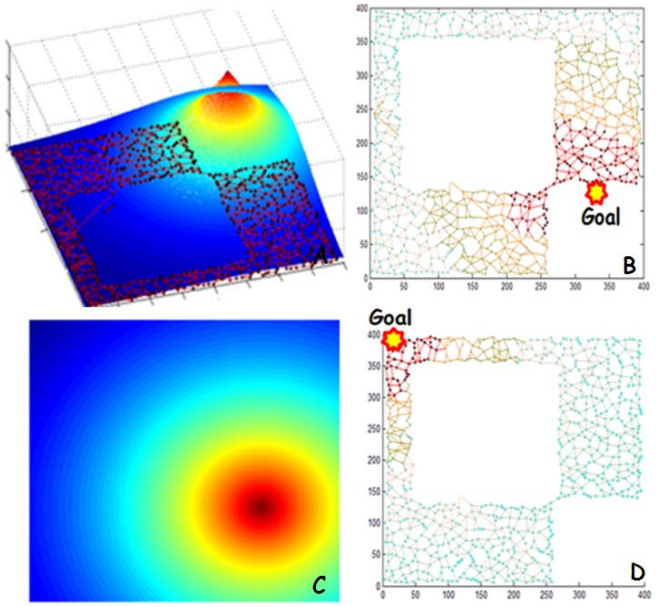


Figure 7. Quasi-stationary value fields superimposed on a growing neural gas in the GNOSYS playground. Panel A-C show different views for the same spatial goal. Panel D shows the resulting value field as the spatial goal changes (Red indicates greater value).

It is quite clear from the expression 12 that neurons in the SMS that have internal sensory weights closer to the active goal (in an Euclidian sense) will elicit higher rewards. Under the influence of this reward excitation, the value field (eqn.10) with relax quickly to its fixed point

$$v_i^* = R_i + \gamma(W_{ij} v_j)_{\max} \qquad (13)$$

Figure 7 shows the quasi-stationary value fields superimposed on the SMS for the case of different spatial goals (after relaxation to their fixed point). Panel A,B and C show different views for the same spatial goal, panel D shows the resulting value field for a different goal.

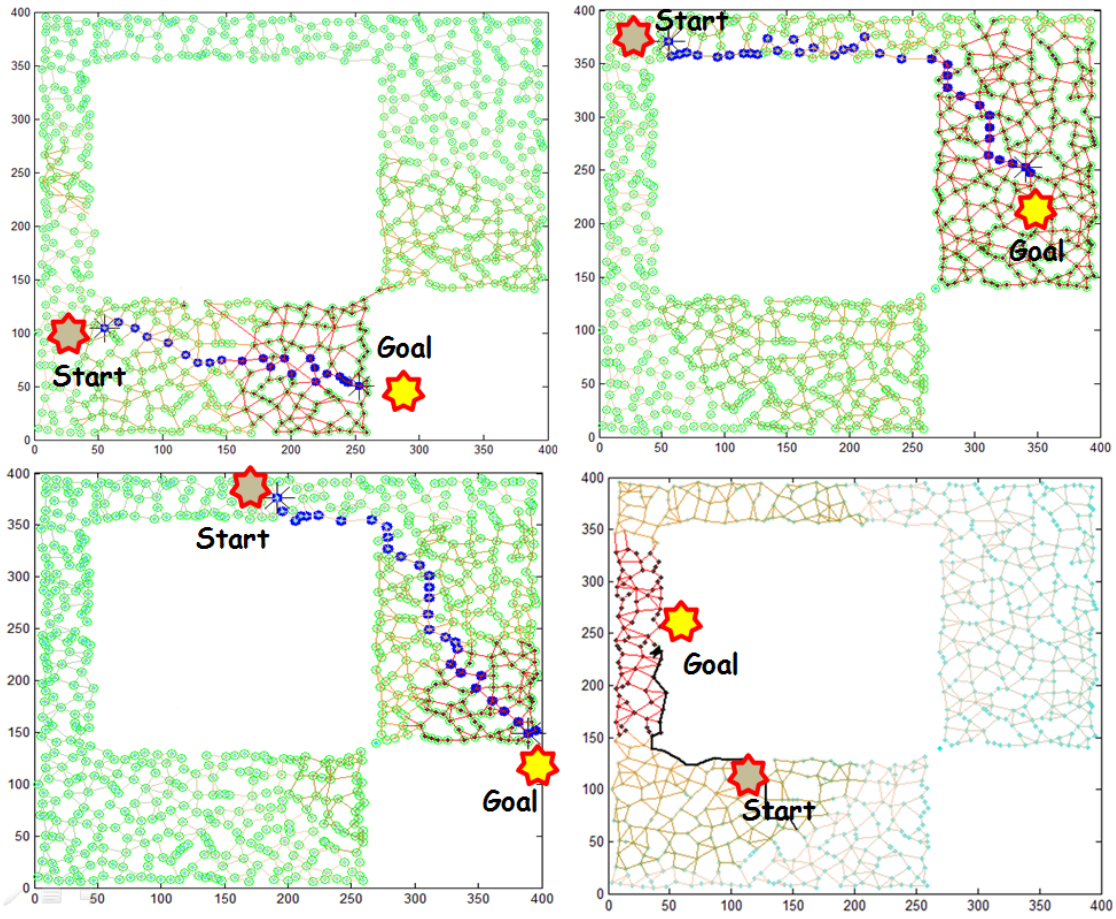


Figure 8. Movements to different spatial goals in the GNOSYS play ground.

The coupling between the value field and the dynamics of the SMS can now be understood by revisiting the expression for action selection (equation 9). The element $m_{k_i i}$ represents the motor weights of a lateral connection between neuron i and its immediate neighbor k_i such that $k_i=\text{argmax}_j(w_{ij}V_j)$. The value field influences the motor activity by determining the neighboring neuron that holds maximum value with respect to the currently active goal. In other words, it determines how valuable any motor excitation m_{hi} is with respect to the goal currently being realized. The motor action that is generated is hence the activation average of all the motor reference vectors $m_{k_i i}$ coded in the motor weights, for all N neurons and at that time instance. The goal induces a value field, that influences the computation of the incremental motor action to move towards the goal for the next time step, this motor activation in turn influences the dynamics of the SMS and causes a shift in activity , now the next step of valuable motor activation is computed and this process progresses till the time the system achieves equilibrium. Hence, the information flows between the SMS and the motor system is in both ways: In the "tracking" process as given by equation 4 the information flows from the motor layer to the SMS: Motor signals activate the corresponding connections and cause lateral, predictive excitations.

In the action selection process as given by equation 9 information moves from the SMS back to the motor layer to induce the motor activations that will enable the system to move closer to the goal. Moreover, using the mental map of the spatial topology the robot can also plan virtual body movements around the playground as a part of a higher level plan, when triggered by a reasoning process. Figure 8 shows the trajectories generated by the robot while moving to different spatial goals in the playground. In sum, output of this circular dynamics involving SMS, action space and the goal induced value field is a trajectory: a trajectory of perceptions in the sensorimotor space and a trajectory of motor activations in the action space.

4.5 Sub Goals: When intermediate states suddenly become more valuable

Before we consider the problem of learning new value fields in a task specific fashion, we will briefly consider the problem of adapting the lateral connections in the SMS in a more flexible fashion using the temporal Hebbian rule. Lateral connectivity between neurons formed during the explorative development of the GNG contributes to the system dynamics in several ways:

1. Top down prediction of future sensorimotor states (both M_{ij} and W_{ij})
2. Value field structure is a function of the lateral weight W_{ij} (equation 10). In this way, even during the time period of execution of a single goal, the value field need not necessarily be constant and can change (locally) in response to run time changes in lateral connectivity.
3. Incremental computations of motor signals (M_{ij})

Adaptations in lateral connectivity in response to new experiences (mainly arising because of a) new changes in the world, b) explorative motor actions triggered in response to these changes that may activate a neighboring neuron with which the lateral connection does not exist or is weak can be initiated using the temporal hebbbian rule (THL).In simple terms, THL strengthens the synapse between pre and post synaptic neurons if they fire in sequence depending on the inter spike interval. In our case the action dependence (modulations in lateral connectivity induced by action space) must also be incorporated into the adaptation rule. The lateral connections can be adapted as follows:

$$\tau_w \dot{W}_{ji} = k_{ji}(C_{ji} - W_{ji}k_{ji}) \tag{14}$$

$$\tau_k \dot{C}_{ji} = -C_{ji} + (\dot{x}_j . A . x_i) \tag{15}$$

$$\tau_k \dot{k}_{ji} = -k_{ji} + (A . x_i) \tag{16}$$

$$A = \frac{1}{Z_A} e^{\frac{-(\overline{M_{ij}} - \overline{M})^2}{2\sigma_A^2}} \quad (17)$$

In order to account for the action dependency, we consider the term $A.x_i$ instead of x_i to represent the presynaptic activity. 'A' is a measure of how closely the initiated action M correlates with the codebook vector M_{ij} of the MMLC between neurons i and j. x_j is the post synaptic unit, and $\dot{x}_j.A.x_i$ is considered as a measure of temporal correlation. C_{ji} and k_{ji} are simply low pass filters over $\dot{x}_j.A.x_i$ and $A.x_i$ respectively. Effects of the above mentioned adaptation mechanism on system behavior is illustrated in a simple example of figure 9c.

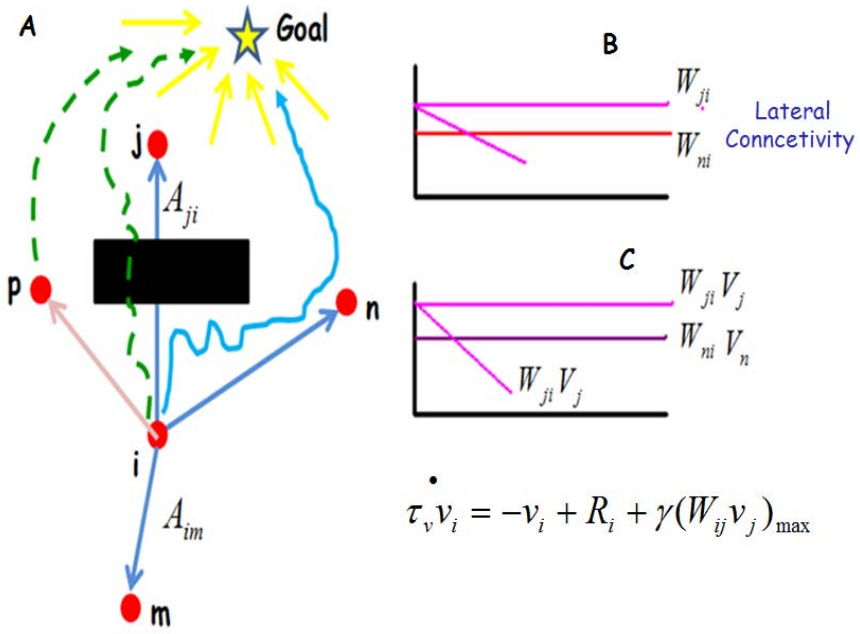


Figure 9.Pictorial representation of how changes in the lateral connectivity (due to a dynamic change in the world) can cause local changes in the value field and result in a change in behavior

Firstly, if the motor signals generated were to move from i to somewhere around j, there are no/infinitesimal changes in lateral weights connecting neuron i to m, as A_{im} (computed using equation 17) will be very small. If an obstacle was placed suddenly, that prevents the movement of the system from i to j, even after initiation of motor signals (this can be as a result of the infra red sensors detecting the obstacle and hence vetoing the translation command, hence the real sensory information from the localizer says that the robot is still in

the same spatial location irrespective of issuing an active translation command), then lateral weights between pre and post synaptic neurons i and j start decreasing slowly. As the lateral weight W_{ji} is decreasing, there are local changes in the value field structure due to the change/decrease in the $W_{ji}V_j$ term involved in the value field dynamics as shown in panel C. This may now cause another neuron (like n) to now hold more value (after some period of exploration during which W_{ji} decays sufficiently and potentially pull the system towards a new path to the goal. Some time, through random exploration, the system may jump to a new state 'p', in which case new lateral weights are formed between i-p. The local changes in lateral connectivity slowly are imprinted in the system, and the behavior become more smoother as the system becomes habituated to the new changes in the world model. As shown in future sections, value field itself may change with respect to time based on reward based interactions with user or self evaluations of success. This may in turn effect behavior and cause changes in the lateral connectivity. One interesting outcome of the discussion in this section is the fact that it is possible to clearly define what we mean by a sub goal, at least for the internal models described in this and the next chapter. A sub-goal is nothing but an intermediate state in the perceptual space (SMS) that usually does not interfere in the normal solution (because its value is relatively lesser compared to normal states that contribute to the solution) but suddenly becomes more valuable because a dynamic change in the world caused a decrease in value of the normal states (hence increasing the relative importance of the sub goal with respect to the normal states). We may even generalize this definition to many real world incidents like, in case we miss a bus when we are in a hurry to reach our place of work, a taxi may suddenly seem more valuable (even though it is not the general path to the goal). As the value field changes, the next incremental updates for the motor activation changes, which subsequently causes a change in the perceptual activity in the SMS through top down modulation and so on. In this way the system moves towards the goal using an alternative path. No wonder, we initiate actions in order to call a taxi and reach the place of work in time!

One interesting outcome of the discussion in this section is the fact that it is possible to clearly define what we mean by a sub goal, at least for the internal models described in this and the next chapter. A sub-goal is nothing but an intermediate state in the perceptual space (SMS) that usually does not interfere in the normal solution (because its value is relatively lesser compared to normal states that contribute to the solution) but suddenly becomes more valuable because a dynamic change in the world caused a decrease in value of the normal states (hence increasing the relative importance of the sub goal with respect to the normal states).

4.6 Constraints: Learning 'when' to optimize 'what'

After learning the SMS for the spatial map through sequences of sensorimotor data in 4.2, dealing with the issue of dynamics i.e. moving neural activations in the sensorimotor space in 4.3 and accounting for the interaction of the representation and dynamics with the currently active goal in section 4.4, we now move towards more subtle issues like taking into account heterogeneous optimality, switching between exploration and exploitation, holonomic constraints imposed by the wheeled platform in the following sections. In the last chapter, we discussed how using superimposed force fields the forward inverse model for the arm movement accounts for multiple task dependent constraints, at the same time reaching the goal. In this section, we use essentially the same field computing principles on the SMS, to plan body movements taking into account multiple task specific and bodily constraints.

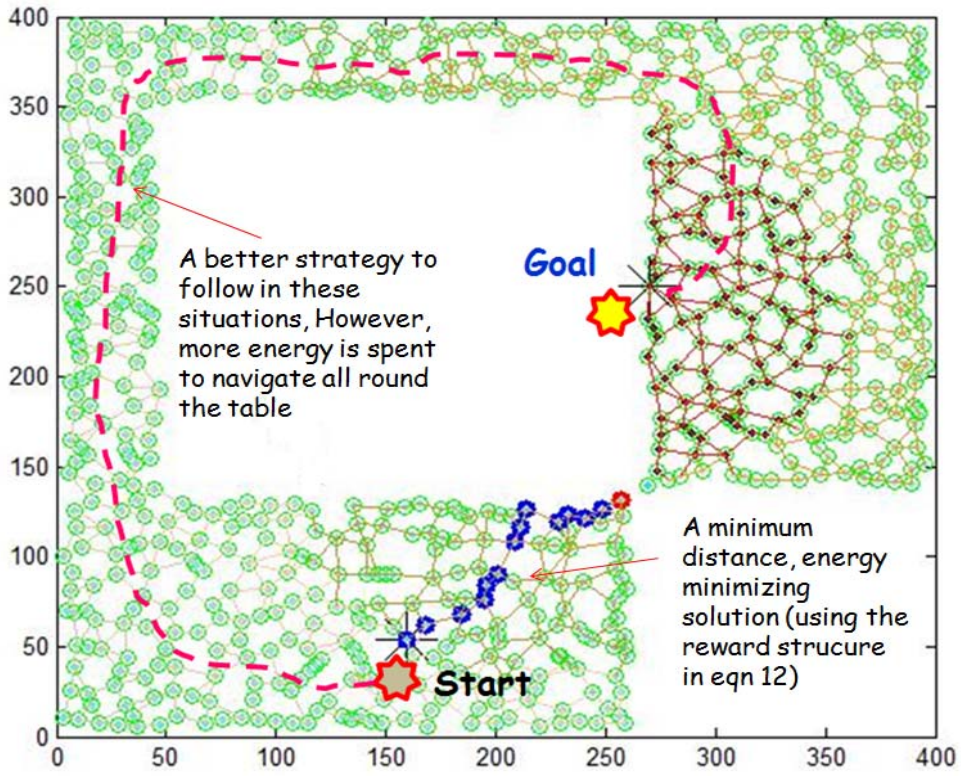


Figure 10. Top view of the playground. The prohibited zone creates situations where goal dependent switches in factors that need to be optimized must be learnt by the robot.

During the discussion in this section, we will revisit two issues that we had temporarily kept on hold in the previous sections in this thesis. The first is related to the use of the prohibited zone in the playground as seen in figure 6 for conducting experiments with the robot. The second issue is related to use of the Q terms in the reward structure of equation 12 that we had not considered in the previous discussions. As shown in figure 10, there can be scenarios

where imposition of minimal distance or energy based criteria in the value fields result in sub optimal strategies (like using the reward structure in equation 12 that causes neurons nearer to the goal induce greater rewards). In other words, instead of having fixed optimization criteria that is always minimized, it is important to learn/have multiple constraints that are combined in a goal dependent fashion and represented in the value fields that organize goal directed behavior in the SMS. In the example of figure 10, the neurons on the right of Start (marked in blue) induce greater reward excitations (when only distance / energy based criteria is present in the value field). This results in the system getting stuck in the location close to the edge of the prohibited zone (approximately, 250,135) and not reaching the spatial goal in ways that could possibly allow any arm related manipulation of object placed in that spatial location. However, that's the best the system can do, unless newer goal dependent adaptations to the value field learnt. However if the longer route from the left (marked in pink) all around the table was chosen, the robot would have arrived to a much better solution with respect to reaching closer to the goal and eventually grasping it (however, at the expense of more energy consumed to take the very long route). Such goal dependent switches in factors that need to be optimized must be learnt by the robot through either runtime rewards given by the user or a self evaluation of its own success.

> How do we make a robot learn subtle goal dependent changes in strategies (of what to optimize) in the continuous, growing representation of the sensorimotor space learnt through pseudorandom explorations and spanning a thousand states with half a million lateral connections?

It is not very difficult find an alternative path through exploration if we choose to discreteize the problem (by dividing the playground into limited set of predefined regions/rooms) and use reinforcement learning methods (like Q learning etc) to learn the Q values of moving from one region to other. But in this case, we are both choosing the representation (predefining regions, limiting state space) and constraining the set of possible solutions, which makes learning through exploration just a trivial exercise through few iterative trials.

These goal dependent qualitative measures in the reward structure that can be learnt through user/self penalization and rewards is represented by the second component in the reward structure. Further as we will see in the example of pushing, there in fact no need to have a default plan, and the reward structure can potentially be learnt directly through exploration.

$$R_i = DP + Q \tag{18}$$

We define the component Q as follows,

$$Q = q_1 + q_2 + ... + q_n \tag{19}$$

$$q_i = \varphi_n . U_i . \frac{1}{\sqrt{2\pi}\sigma_G} e^{\frac{-(G-Gi)^2}{2{\sigma_G}^2}} \tag{20}$$

In simple terms, Q is a net resultant of superposition of a set of n fields q_n on the SMS. Every individual component q_i is learnt goal dependent additional field and has a scalar value on every neuron in the SMS. U_i is the i^{th} interactive or self penalization/reward given to the system. $\varphi_n . U_i$ is the gradually scaled value $(1 > \varphi_n > 0)$ of reward/ penalty given by the user at the end of the performance of the behavior passed on to all the neurons in the the SMS (mainly to the neurons that actively contributed towards the generation of the solution and some of their topological neighbors). The final term encodes the dependency of the goal, the solution for which the agent was rewarded/penalized. This term allows the system to generalize the presence/absence of value field q_i for other goals. In other words, this term evaluates how much value a good/bad experience encountered in the past while performing a goal G_i (for which the additional field q_i was learnt) holds in relation to the currently active goal G.

Figure 11 shows a simple example of the learning of additional task dependent value fields (Q) based on user/self feedback. Panel A shows the solution generated by the robot while trying to reach a spatial goal G from the start position shown using the reward structure of equation 12. Since this was not the appropriate solution to allow any further arm related manipulation of any object placed at the goal, we penalize this solution. These rewards/penalizations can also be done by the robot itself, if we program a criteria for evaluation of success inside it (for this case, the criteria could be the final distance between the body and the goal once the system has settled to an equilibrium). Coming to the problem of penalization, we need to find a simple mechanism to distribute the reward received at the end, to all the intermediate neurons that contributed to the solution. This is reminiscent of the well known temporal credit assignment problem in reinforcement learning. Since we are dealing with high dimensional continuous state spaces, we tried to use some simple heuristics related to distribution of the end reward to the neurons in the

The goal dependency term in eqn.20 allows the system to generalize the presence/absence of reward field q_i for other goals. In other words, this term evaluates how much value a good/bad experience encountered in the past while performing a goal G_i (for which the additional field q_i was learnt) holds in relation to the currently active goal G.

SMS that proved very used both in the case of learning new constraints in the spatial map as well as for the case of pushing.

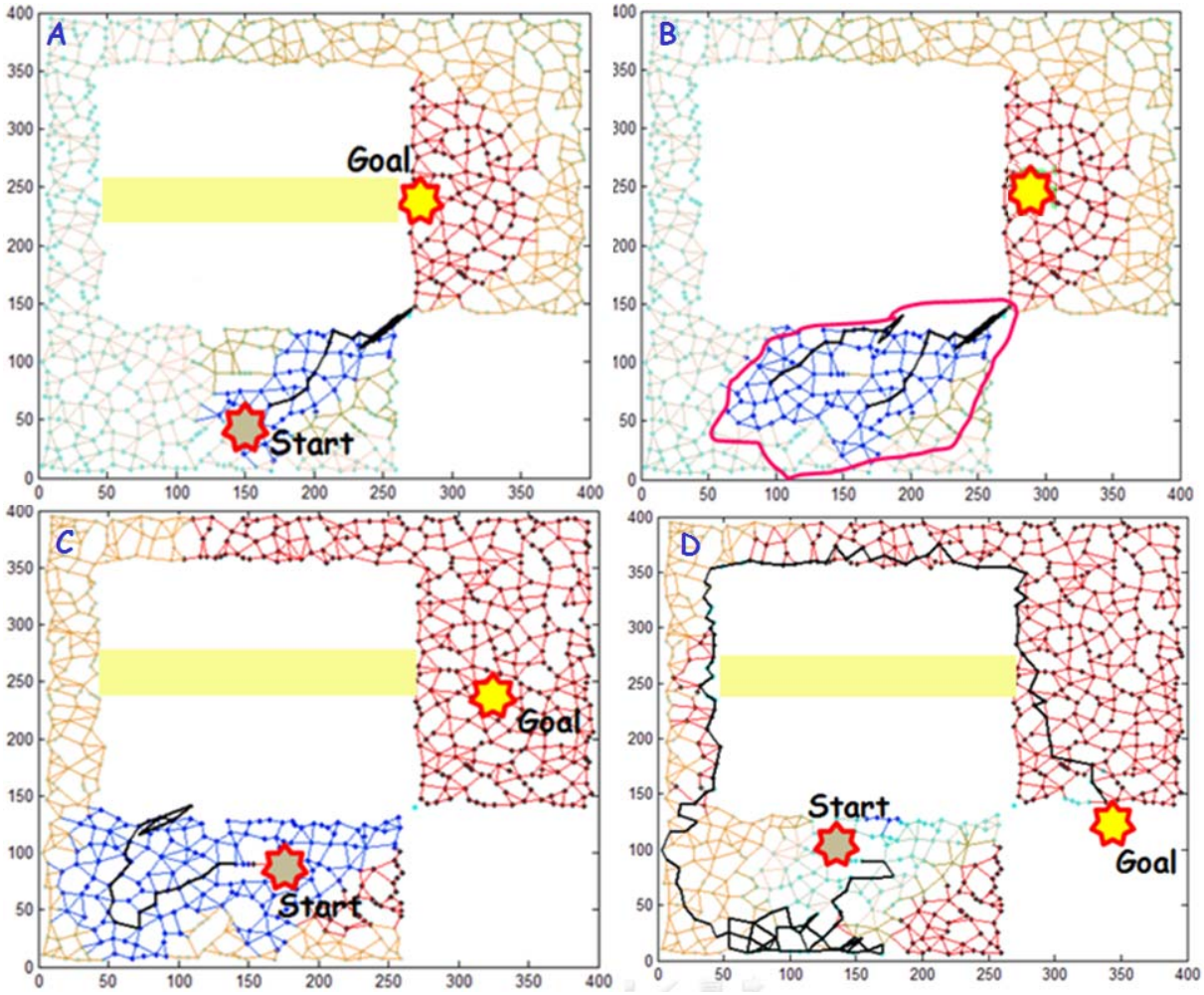


Figure 11 (A-D): Shows the learning of additional task dependent value fields (Q) based on user/self feedback. Four different runs of the system starting from different initial conditions to reach different spatial goals (yet preserving the nature of the problem), the penalization of the solution, the resulting value field and the resulting behavior under the influence of the new field structure are sequentially shown.

While computing in the SMS, it is always possible to keep track of the temporal sequence of activations in the neurons that lead to the solution. So we used the following procedure for distributing and rewards and penalties to the system. Let us consider that the set Z:(z_1, z_2..z_n) represents the temporal sequence of neurons in the SMS that were active in the SMS dynamics leading to the behavior that has to be rewarded or penalized. Let us also assume that U_i is the penalty or reward that needs to be distributed to the neurons that are members of Z. The simple heuristics we employed is the following one:

a) In case of a penalization, the most proximal neuron z_1 receives the maximum penalty U_i, and its topological neighbors to which it is laterally connected receive a scaled version of U_i as penalty.

b) During the distribution of a penalty, while moving from the most proximal neuron to the most distal neuron in the set Z, every neuron in Z receives a scaled version of the penalty U_i say $\varphi_n.U_i$ such that φ_n decreases by small amounts (in our case, from 1 to 0.4) as we move from z_1 to z_n. In the similar fashion, the neighbors of these neurons also receive scaled versions of the net penalty that the master received. All the neurons in the SMS, that are neither members of Z and neither neighbors of neurons that are members of Z don't receive any penalty. By using this mechanism what we have heuristically generated is a new penalty field that has a value on every neuron in the SMS. In a brief moment, will see how this field contributes in future behavior of the system to reach a new goal. For example, Panel A in figure 11 shows the resulting quasistationary value field on the SMS (after stabilization as in eqn.13) for the same spatial goal, after the new penalty field was added to the existing reward structure. As we can clearly observe, after the penalization of previous solution (marked in black), a new value field is added causing neurons in the neighborhood of the solution (blue region) to induce very less (or negative) reward excitations for spatial goals around region G. Panel 11 B shows another run for the same goal from a different initial condition under the influence of this value field. As evident, the system still tries moving to the right (optimizing energy) and settles at a sub optimal solution. It cannot move further to the right because of the changes in the value field due to penalizations in the previous run. This solution is also penalized using the same procedure outlined above and the resulting field is also shown in 11B. Now as the robot is asked to reach the same spatial goal (from a different initial condition) under the influence of this value field, we observe that (panel C, solution is marked in black) it initially moves in the correct direction taking the longer path, but later turns rightwards in a circular loop and settles to a new but still sub optimal solution. From the point of view of using the default plan of minimizing distance, the performance of the system is degrading, as it is moving farther away from the goal. This solution is also penalized and the resulting value field structure is shown in 11C (Red=high reward locations, Blue= very low rewards). As shown in panel D, after an initial phase of turbulent behavior in the region where the added penalization in the previous runs cause maximum influence, the system is eventually pulled towards the goal. Since the agent reaches the goal this time successfully , this solution is rewarded, resulting in a new addition in the value field Q2. It is easy to imagine that, this reward will have the effect of encouraging similar solutions, while moving from the neighborhood of the starting point, to the neighborhood of the goal region. During the process of

distribution of rewards to the contributing neurons, we follow exactly the reverse of the procedure we followed to distribute penalties i.e. the last neuron the set Z receives the full reward, and all the proximal neurons receive scaled versions of the full reward. This simple heuristics for distributing rewards and penalties to neurons in the SMS embodies a simple logic that in case of a problem or bad performance, the root is attacked and incase of success all the contributing elements get the rewards in ways such that elements higher up in the hierarchy earn more benefits than those at the bottom. In fact this simple heuristic of distribution of rewards underlies basic human nature of attribution of credits to any collective goal directed behavior. In case of problems, elements at the bottom of the hierarchy face maximum damage and in case of success elements at the top of the hierarchy reap maximum profits!

c) The third rule which we employed in the distribution of rewards an penalties to the neurons in the SMS that contribute to a specific behavior, was developed during

During the process of distribution of rewards to the contributing neurons, we follow exactly the reverse of the procedure we followed to distribute penalties i.e the last neuron the set Z receives the full reward, and all the proximal neurons receive gradually scaled versions of the full reward. Since the net work done by the system when it employs a circular solution is nil, the third rule relates to actively discouraging solutions that result in no net work. In fact this simple heuristic of distribution of rewards underlies basic human nature of attribution of credits to any collective goal directed behavior. In case of problems, elements at the bottom of the hierarchy face maximum damage and in case of success elements at the top of the hierarchy reap maximum profits!

Finally, the net resultant field that comes into play during the execution of any arbitrary goal is a superposition of the field representing the default plan and a set of experience related fields that the robot learnt in the past, scaled appropriately based on their relevance to the current goal.

experiments related to pushing. This had an effect of smoothening the resultant field structure. We can intuitively understand this rule, by once again looking at the initial turbulent behavior of the robot in 11d. We can also see many circular solutions, the same was observed even during explorative phase of pushing when the ball is pushed back and forth returning back to the same initial position instead of moving to the goal. Since the net work done by the system when it employs a circular solution is nill, circular solution are discouraged by a small penalty or they are not rewarded.

How these new learnt fields influence the behavior of the system for a different goal can be understood by considering equation 20. The net input of a reward field q_i learnt while realizing goal G_i, towards the a currently active goal G, is a scaled version of the correlation

between the two goals G and Gi. This implies the net resultant field that comes into play during the execution of any goal is a superposition of the field representing the default plan and a set of experience related fields that the robot learnt in the past, scaled appropriately based on their relevance to the current goal. In simple terms, the default plan always exists, the rest of the experience related fields are switched on and off based on the relevance of the experience with respect to the currently active goal.

Figure 12 (panel A) shows the effect of the only Q component in the reward equation in the value field, after the completion of phase of goal dependent value field adaptation. What the system has encoded through external/self penalizations and rewards is that, for goals approximately around region G, and initial conditions around region S, it is more rewarding to optimize for greater rewards at the end (spending more energy), than optimizing for distance and reaching a close enough, but not good enough solution. Panel B shows an almost smooth solution moving all the way round the table. Panel C shows 6 different paths travelled by the robot to the same goal from 6 different initial conditions.

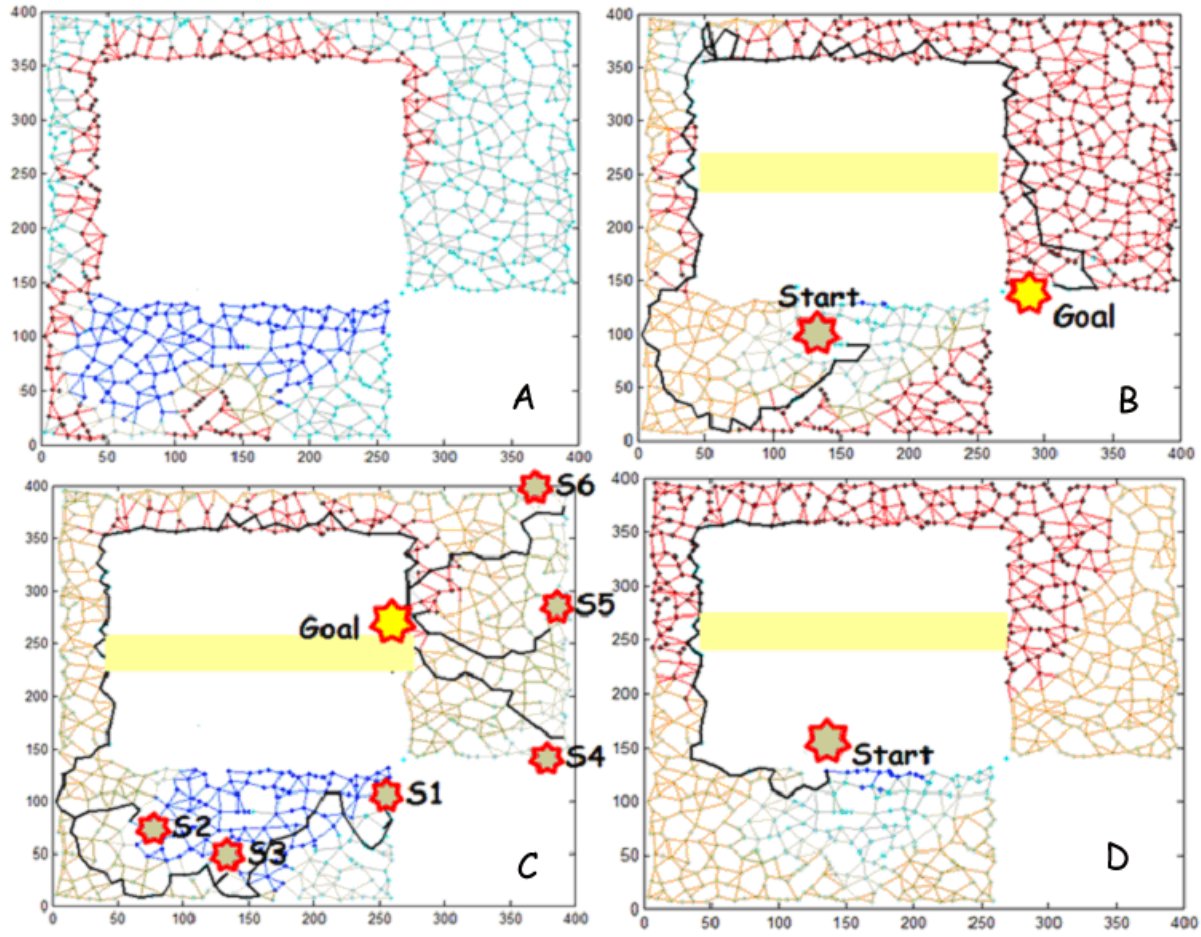


Figure 12.

We must observe that the new adaptations in the value field do not deteriorate the solution of the dynamical system for other initial conditions. For the target goal in panel D, we see that the learnt additional components Q in the value field (blue areas of panels A-C) do not

have much effect (or are almost switched off), and the system just moves along the field representing the default plan.

The goal dependency term in eqn.20 allows the system to generalize the presence/absence of reward field q_i for other goals. In other words, this term evaluates how much value a good/bad experience encountered in the past while performing a goal G_i (for which the additional field q_i was learnt) holds in relation to the currently active goal G.

4.7 Contradictions: the exploration-exploitation dilemma revisited

In section 4.6, we dealt with the issue of how the dynamics of the SMS can take into account task specific switches in parameters that need to be optimized while realizing a spatial goal. In this section, we will consider a simple simulation example related to the problem of taking into account dynamic changes in the world. While using classical reinforcement learning approaches (even in a simplest case of having a small and discrete state spaces), exploration has to be done at least till a time in future when reward is received, which is then used to assign values of intermediate states by solving the credit assignment problem. In a high dimensional and continuous sensorimotor space, it is easy to visualize that keeping on doing random exploration is not a feasible option. Is to possible to balance explorative dynamics and normal dynamics under the influence of an external force that dynamically pulls the system back to normal behavior when it senses that exploration is no longer necessary? In other words, if exploration is done in the presence of an already existing goal based attractive field, even if the reward may not be immediately available, a measure of success is immediately available (through the bifurcation parameter) that can be exploited to trigger the switch between random exploration and normal dynamics.

To illustrate this, let us consider a scenario where a new obstacle is introduced in the environment in ways such that it will interfere with the usual solution that the robot employs while moving towards a goal as in figure 13. The robot will move normally till the time it encounters the obstacle. As the robot comes closer to the obstacle, it is detected by the infrared sensors and a threat message is created. At the level of execution now there is a contradiction in the sense that a) the dynamics of SMS delivers the next incremental motion commands needed to reach the goal (that will result in collision with the obstacle, since the obstacle is not yet represented in the internal spatial map) b) the infrared signals sense the threat posed by the obstacle in front of the robot and generate a message regarding the threat involved in moving ahead in the heading direction.

As we mentioned in chapter 2, the Gnosys Agent has inbuilt mechanisms to arbitrate contradicting execution requests, and now veto's the decision of not executing the motion

commands delivered by SMS dynamics in the presence of the threat detected by the IR sensors. This now creates a contradiction in the level of sensorimotor space, since the anticipated new state of the robot $S_{Anticip}$ (eqn. 7 and 8) computed through virtual execution of the current incremental motor activations (using multiplicative modulation) is different from the incoming real sensory information S. A consequence of this contradiction is that the bifurcation parameter β_{if} drops to zero. The incremental actions now proposed by the SMS are small random motor signals (since $\zeta \to 1$) and hence cause the robot to wander randomly exploring the space around the obstacle as seen in the figure 13.

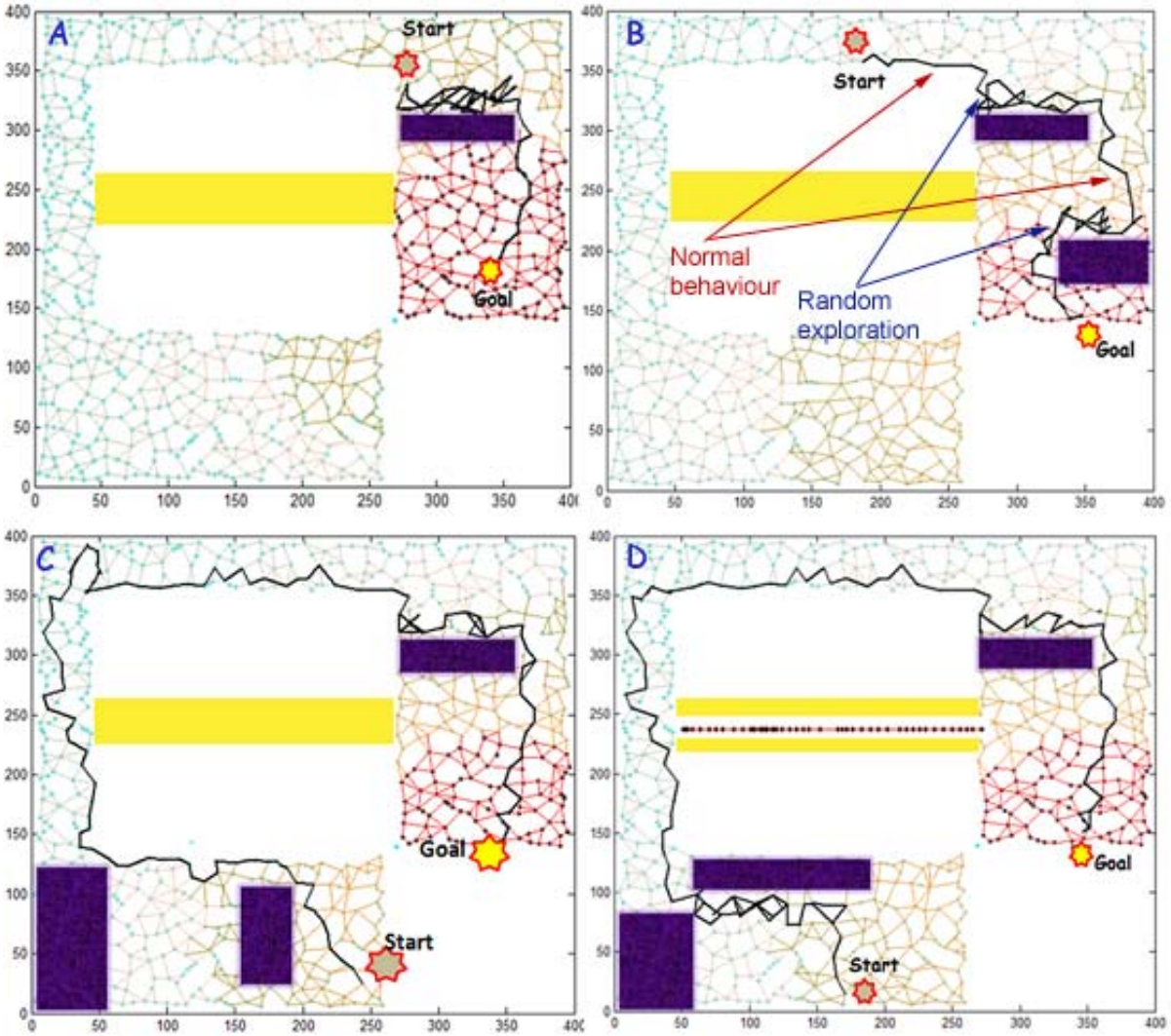


Figure 13. Behavior when obstacles are placed dynamically in the environment.

System behavior now is more explorative like the initial stages during the learning of the GNG and remains like this till the time the system is pulled back to normal behavior, so that the previously existing value field can take control again in determining the action. This is possible if the value of β_{if} increases again so that actions generated are based on the influence of the value field and not purely random motor excitations. We note that the value

of β_{if} is computed every time step based on the motor action that is computed (which determines the predicted sensory consequence) and the incoming sensory information. When the anticipated and incoming sensory information are close enough, it implies that there no need for doing random exploration (at least for the time being). 13 c-d also show solutions of the system for other dynamic obstacles placed in different locations in the playground. Hence using the lower level preprogrammed vetoing schemes in the gnosys agent that arbitrate contradictions at the level of execution and the switching between random exploration and normal behavior based on the contradictions at the level of SMS allow the system to flexibly take into account dynamic changes in the world.

4.8 Causality: On a growing neural gas for pushing

We now move on to the next internal model of action i.e. 'Pushing' that we need to create in GNOSYS in order to meet the demands of the different reasoning tasks considered in chapter two. To develop the pushing model we will use most of the computational mechanisms that we have discussed till now, mainly

a) the forward inverse models for reaching that we developed in the previous chapter to initiate the pushing action i.e. the sequence of reaching a stick placed on the table, grasping it, then reaching (now using the gripped tool) the goal object that has to be pushed;

b) the procedure of self organizing sequences of sensory and motor data that was outlined in 4.2 in order to develop the pushing SMS, the dynamics of the SMS that we described in 4.3, the use of value field dynamics covered in 4.4 to generate goal oriented motor sequences of pushing and learning new reward fields based on experience (section 4.6) in order to dynamically change the direction of pushing based on the presence of traps in the trapping groove.

Since the basic mechanisms of learning, dynamics, action generation etc have already been covered in the previous sections and considering that all the internal models presented in this thesis share the same common computational principles, we keep the discussion on pushing quite concise in relation to these issues, and focus more on the novel details that may be of interest to the reader.

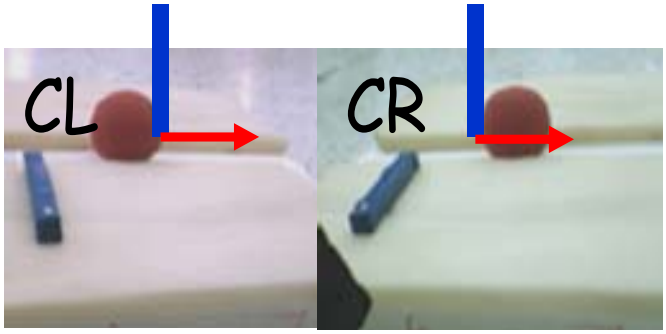


Figure 14. Pushing to the right in the case of CL will not induce any motion on the ball. Pushing to the right in the case CR will displace the ball based on the amount of force applied (i.e. approximately equal the displacement of the stick in contact with the ball along the trapping groove, based on the small random changes in the DOF $\theta 1$ and $\theta 5$ of the KATANA arm).

Similar to the development of the SMS for GNOSYS play ground, a growing SMS for pushing in the trapping groove was built. We simplify this scenario by considering pushing to be only functional along the x axis. The sensory information coded in the SMS is the location of the object (being pushed). This information regarding the CG of the goal object comes coming from visual recognition system and reconstructed suitably in 3D space using the computational model presented in section 3.4.1.

The motor space consists of the following variables (shown in figure 14)

a) Location of the tool with respect to Goal. In principle the robot could reach any point along the ball with the stick using the forward inverse model for reaching. Since we are pushing only inside the trapping groove, we considered only two possible locations around the object where the robot reaches with the stick (as shown in figure14). If x is the position of the ball, the robot brings the stick randomly to locations that are 6 cm away from x in both positive and negative directions.

b) The amount of force applied to the object. This is once again simplified and made proportional to the change in the DOF $\theta 1$ and $\theta 5$ of the KATANA arm as described in figure 2 (and invariant of the orientation of the body). After a few trial runs of pushing, we estimated that a relative shift of 45 degrees (+/-) in $\theta 1$ coupled with a shift of 60 degrees (+/-) in $\theta 5$ is sufficient (and also the safe limit so as to avoid accidents, damage to the gripper etc) to displace the ball from the center of the trapping groove to a location around the corner of the table, that could be visible by the cameras (Figure 14 shows the camera images of an example case CL is the left camera and CR is the right camera).

Figure 15 shows a typical pushing sequence on the robot. Even though the picture shows a trap, initial training phase for developing the pushing SMS and learning the reward structure (DP component in eqn. 18) did not involve any traps. We will delay the problem of the presence of traps in the trapping groove until the next section.

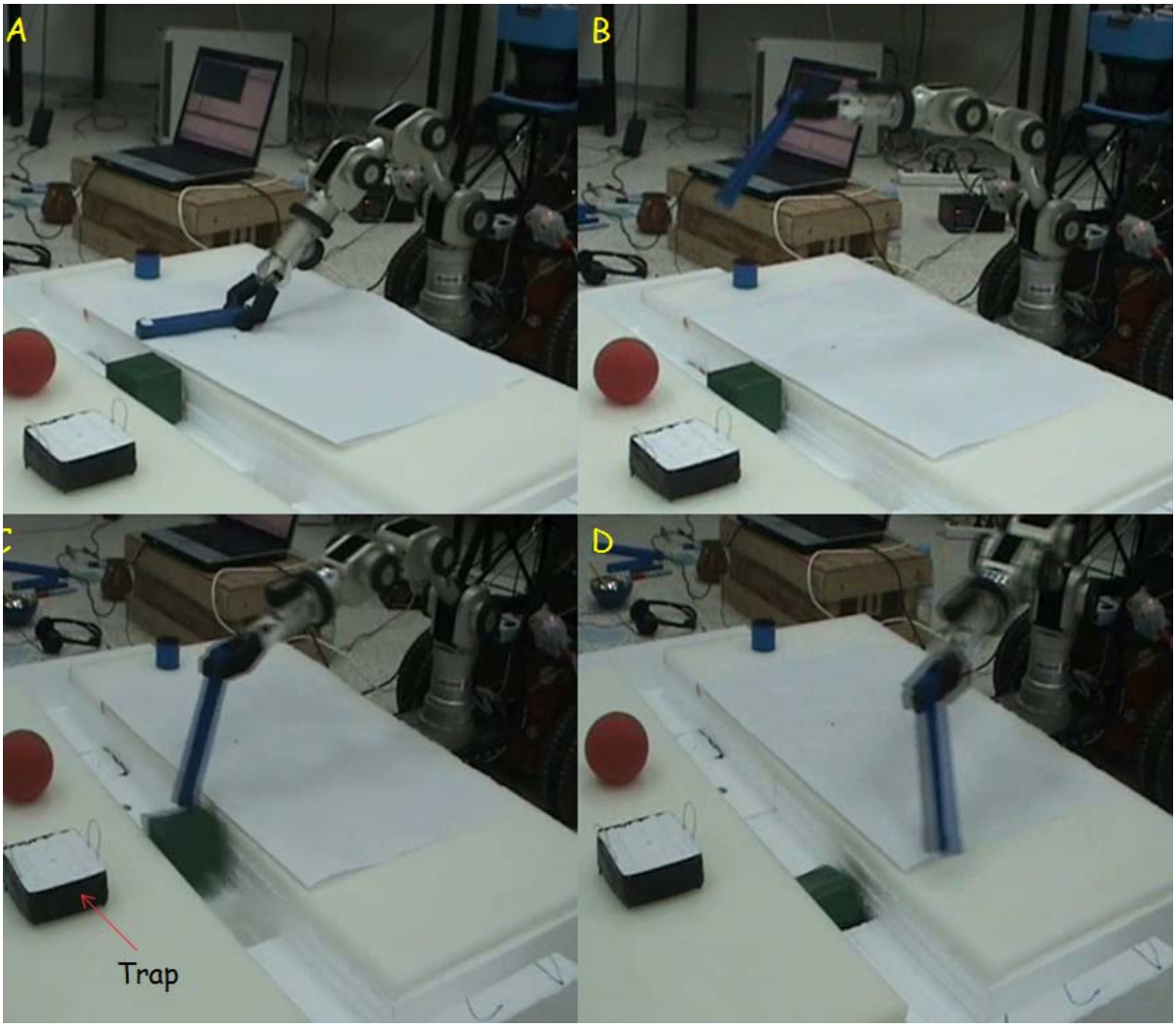


Figure 15. A typical test sequence of pushing in GNOSYS. The forward/inverse models for reaching is used to reach a stick placed on the table, and then to reach either sides of the goal object with the gripped stick. Then a small change in the DOF $\theta 1$ and $\theta 5$ of the KATANA arm is commanded so as to possibly slide the object smoothly along the groove.

Based on the data generated by repeated sequences of reaching a goal object randomly on the either sides and pushing in different directions (with different amount of force) and then tracking the new location of the ball, we trained the pushing SMS. Figure 16 a shows the trained growing neural gas for pushing on the trapping grove along with the spatial map that was developed in the previous section. As shown in figure 16 b, now the movement of the goal object along the trapping groove as a result of a virtually executed pushing action can induce reward excitations on the spatial map that can eventually cause body movements to be virtually initiated in order to move closer to be goal. In other words, we couple the

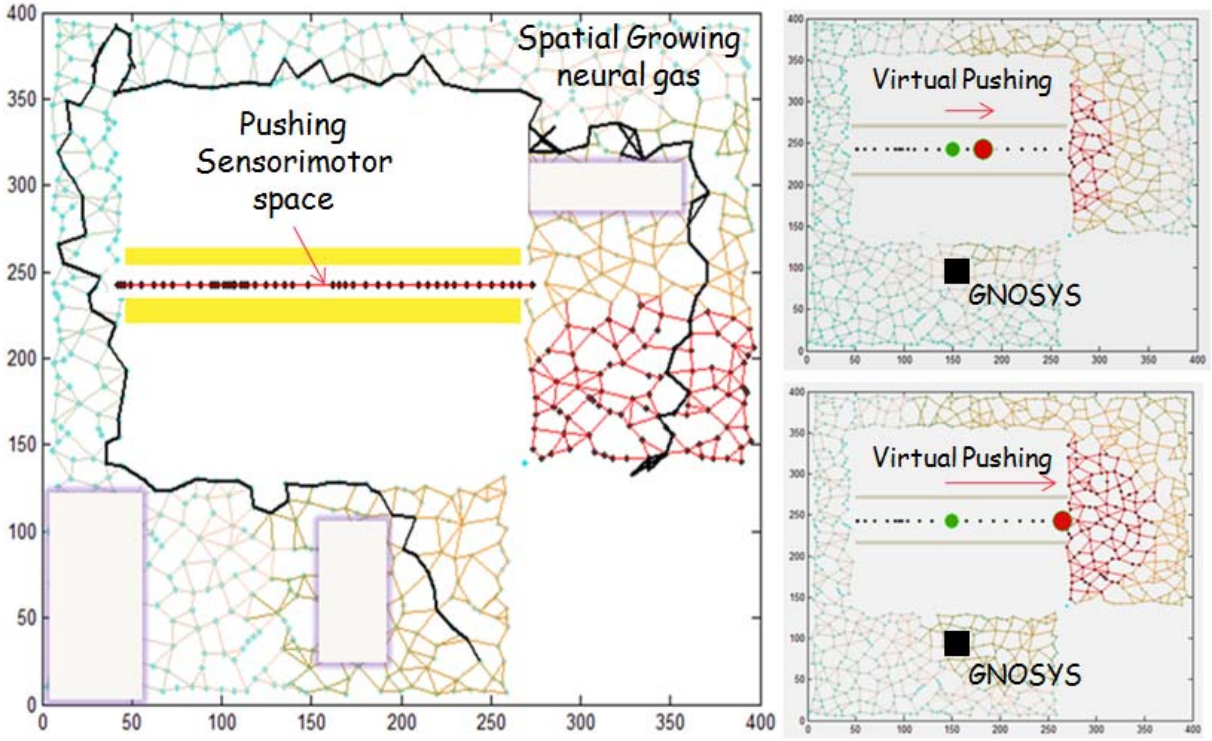


Figure 16. It shows the internal spatial map of the Gnosys playground along with the sensorimotor space for pushing in the trapping groove. A virtually executed pushing action (and its anticipated consequence in terms of the displacement of the goal) can now induce reward excitations on the spatial map allowing virtual body movements to be initiated by the robot so as to evaluate the possibility of approaching the goal from some other spatial location in the playground.

pushing SMS with the spatial SMS, the robot can carry virtual experiments like pushing the goal along different directions, predicting the sensory consequence of the pushing (estimated position of the goal after the push) through motor modulated lateral connections and then evaluating if the new spatial location of the goal is reachable through a process of navigation in the playground.

Even though in the first attempt, we represented the sensory input (the location of the goal object being pushed) in global frame, representing it with respect to body would have helped in generalizing the learnt knowledge about pushing to be applied anywhere in space. More over since we are pushing only along the x-axis wrt the table, and not along the y-axis (because of the constraints imposed by the arm), it would be extremely interesting to couple degrees of freedom of the body in the pushing action (i.e. in case of the need for pushing along the y axis, the robot then just needs to reorient its body appropriately and then initiate the normal pushing action). We hope to incorporate these extensions in the future versions of the pushing model.

In order to realize any high level goal that that requires a pushing action to be initiated, it is not just important to be able to simulate the consequences of pushing, but also to be able to

'push in ways that are rewarding'. In other words, after learning the pushing SMS, we now have the task of making the robot learn the reward structure involved in a pushing action, so that it can coordinate the pushing in a goal directed fashion. In the simple set up of pushing in the trapping groove, we can estimate that pushing the goal to the either edges of the table should be maximally rewarding, since it ensures that the robot can move around the table and grasp the goal. We note here that the reward structure (eqn.11) for pushing SMS need not be predefined (no need of default plan) and can be learnt by repeated trials of random explorative pushing of the goal in different directions along the groove, followed by an attempt to grasp the goal (by moving and pushing). These trials can also be done in the mental space the once the SMS for pushing is learnt (by co-activating the spatial dynamics after the pushing action). Following a trial, either the user can present the robot with a reward or else, the robot can self evaluate its success. As scaled version of this reward is distributed to all the proximal neurons in the pushing SMS that contributed to the pushing action in this trial (in the manner described in 4.6).

In order to realize any high level goal that that requires a pushing action to be initiated, it is not just important to be able to simulate the consequences of pushing, but also to be able to 'push in ways that are rewarding'. Energetic efficiency in the combined task of pushing and navigating can also be incorporated in the reward structure.

Once again energetic issues can also have their effects in the learnt reward structure, since there are multiple solutions to get the reward successfully. Influence of energetic issues in the reward field can be introduced by adding a decaying element in the promised net reward for achieving a goal successfully, which is a function of the amount of energy spent in the process of getting the goal. In every trial, the robot randomly pushes the goal in different directions and virtually evaluates the possibility of reaching the now displaced goal (using the GNG for spatial navigation, and the forward/inverse model for reach/grasp). For the effort, the robot receives a scaled quantity of the promised net reward as in equation 21, that is a function of both the final solution and its energetic efficiency . The received reward R_T is then distributed appropriately to all the neurons in the GNG for pushing.

$$R_T = R_{net} \quad if \quad Dist_{iter} < \delta$$
$$R_T = R_{nst} e^{-(Dist_{iter}/125)} \quad if \quad Dist_{iter} < \delta \qquad (21)$$

with $\delta = \frac{Goal - Initpos}{1.5}$.

R_T is the actual reward received in the end of the T^{th} trial in case of success, R_{net} is the net reward promised in each trial (we kept all promised rewards for success as 50), $Dist_{iter}$ is an approximate calculation of the distance navigated by the robot to get the goal, that is estimated based on the number of neurons in the spatial SMS that were active in the trajectory from initial position of the robot to the goal. We must note that this distance travelled $Dist_{iter}$ is a consequence of the pushing action that preceded navigation and not a result of the constraints in spatial navigation in the playground). In other words, if the robot pushed the goal to the right, it needs to navigate a much greater distance that it would have had to in case it had pushed the goal to the left. This is reflected in the number of neurons that are sequentially activated during the path from source to goal i.e. $Dist_{iter}$. Since navigation has a high cost in terms of battery power consumed and since navigating greater distances than that was necessary directly implies spending more energy than that was necessary, the term $Dist_{iter}$ is one of the parameter that helps in distributing rewards based on the energetic efficiency of the solution. The other term δ is the ratio of the shortest distance between 'the initial position (*Initpos*) of the robot from the final location of goal after pushing (Goal)' and 'representational density of neurons covering the spatial GNG ' that we conservatively approximated as 1.5.

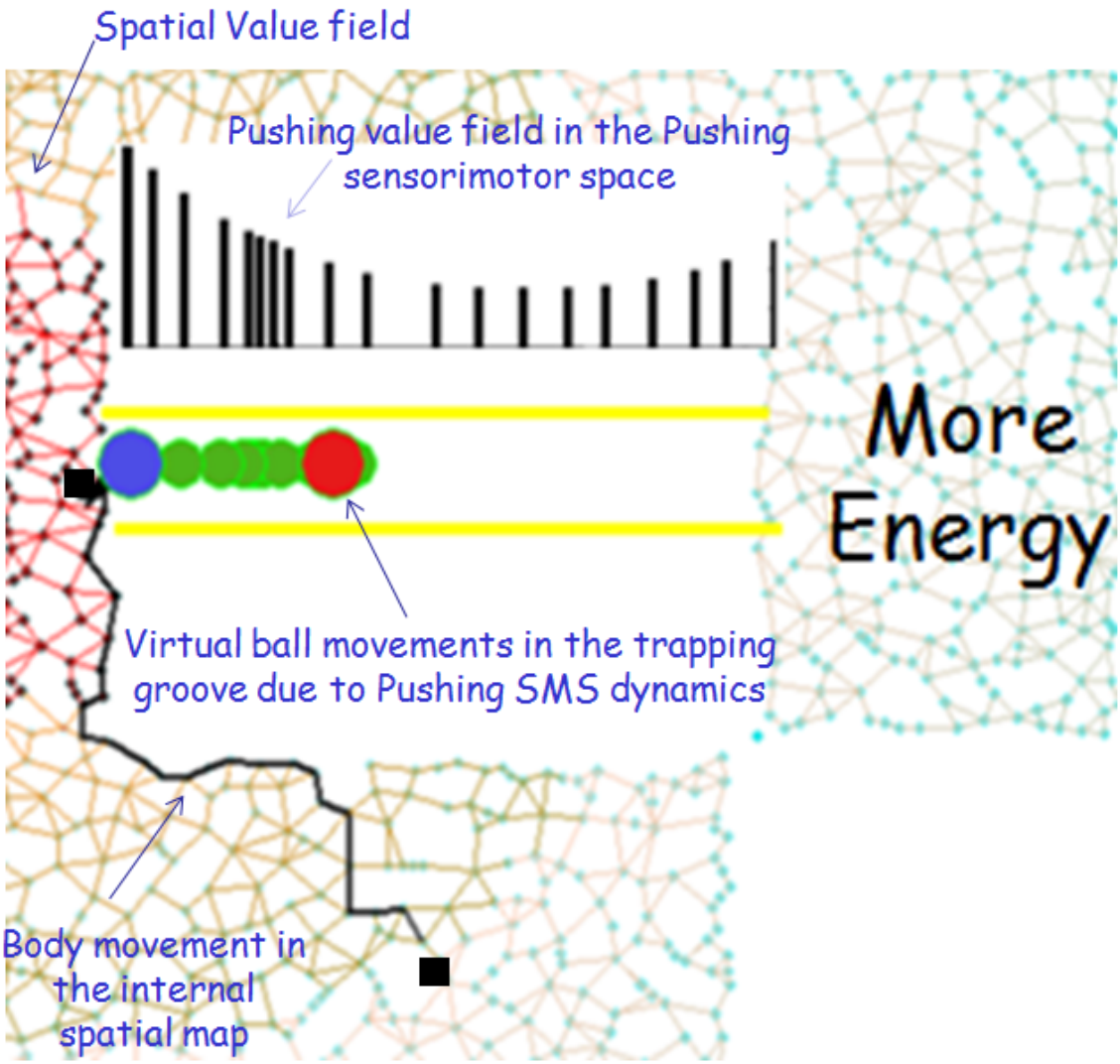


Figure 17. Combined sequence of pushing and moving in the mental space. Based on the pushing value field, the next incremental motor action for pushing is computed. Motor activations then modulate lateral connections in the pushing SMS and cause activity shifts in the pushing SMS corresponding to the new anticipated spatial location of the ball in the trapping groove as a result of the pushing. Now based on this nonstationary activation in the pushing SMS, and the pushing value field, the next incremental pushing action is computed and so on till the time the system attains equilibrium. The final anticipated spatial position of the ball after the pushing SMS dynamics is over in turn induces a quasistationary value field in the spatial map that triggers the spatial SMS dynamics so as to eventually pull the body towards it. We can also observe from the pushing value field encourages the robot to push towards the left since it is an energy efficient strategy and hence more rewarding. However, as we will see in the next section this may not always be true if there are dynamic changes in the world (like introduction of traps) during which always pushing the goal to the left may result in a failure to get the reward.

In simple terms, if the shortest distance between the displaced location of the goal and initial position is 46cm, δ is around 30, which means that not more than 30 neurons should

be active in the SMS during the journey from the initial position of the robot to the goal for an energetically efficient solution. After every trial , the reward received by each neuron in the GNG is added to its previous accumulated reward value. After about 50 trials, we averaged the rewards received by each neuron in each trial, in order to generate the final reward structure for pushing. This reward structure can now be used to compute the value field which then drives the pushing SMS dynamics. Figure 17 shows the value field on the pushing SMS computed based on the average reward structure learnt after 50 trials. The X axis shows the active neurons along the trapping groove, the Y axis shows the rewards received by each neuron in each trial. We can see that after a large number of trials, the neurons representing the left end of the groove elicit greater value that the right end. Based on the value field, the next incremental action for pushing is computed. This then modulates the lateral connections to cause a shift in activity that corresponds to the anticipated movement of the ball in the trapping groove. Now based on this nonstationary activation in the pushing SMS, and the value field, the next incremental pushing action is computed and so on till the time the system attains equilibrium. Once the final anticipated position of the ball is known after pushing SMS reaches its fixed point, the value field for spatial navigation can be computed in order to navigate towards the ball. The value field for spatial navigation and the corresponding trajectory generated by the spatial SMS dynamics is also shown in figure 17.

4.9 Foresight: Pushing in the mental space to 'push intelligently' in the physical space

In the previous section considered simple examples of foreseeing the combined effect of pushing and moving in the context of an otherwise unrealizable goal (like grasping a ball placed in the center of the table) using the dynamics of pushing SMS and spatial SMS. We also saw how the learnt reward structure for pushing encourages the robot to push more often towards the left in an attempt to reach an energy efficient solution (unless the initial position of the ball is closer to the right end of the trapping groove). We now introduce some more constraints on the pushing scenario by introducing traps randomly at different locations along the trapping groove.

Traps were indicated to the robot through visual markers (in both the tables we constructed) so that their location in the groove can be estimated by reconstructing the information coming from the visual recognition system.

Figure 18. Introduction of traps in the trapping groove.

When traps are introduced initially in the trapping groove, the behavior of the system is only governed by the previous existing value field. Hence the robot follows the normal strategy as in the previous section. As seen in the three trials after introduction of traps shown in figure 19, the ball is pushed as a function of the value field we learnt in the previous section shown in pink on top of the trapping groove and is always constant. This normal behavior continues till the time a contradiction is encountered between the anticipated position of the ball as a result of an incremental pushing action and the real location of the ball coming from the 3d reconstruction system. This is in fact similar to the earlier case of insertion of dynamic obstacles in space when the robot was navigating to reach spatial goals. An contradiction automatically implies that there are new changes taking place in the world whose effects are not represented internally by the system. As we saw in 4.7, such contradictions result in a phase of exploration at least till the time the system is pull back to the normal behavior by the existing value field. Now the robot initiates incremental random pushing in different directions till the time the ball begins to move as anticipated, in which case pushing is once again governed by the preexisting plan. The pathof the ball during random pushing and normal behavior is shown in figure 19 for four different cases . In the first case, since the initial location of the ball is close to the right end, following the normal behavior, the ball was pushed rightwards where it collides with the trap placed at around 220, this motion of the ball is shown in green with the white arrow. Now there is an active phase of random pushing for a while with the ball moving forwards and backwards, till the time it reaches a position from where it is the preexisting value field takes over.

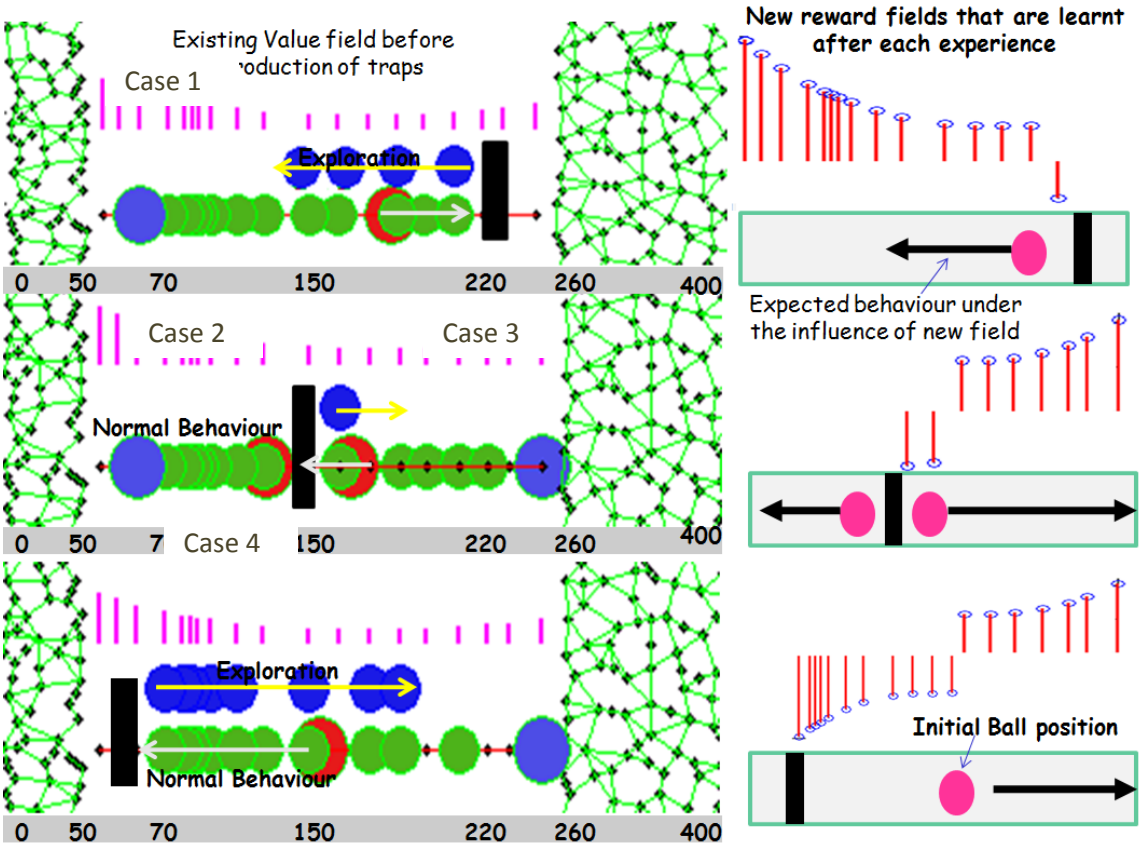


Figure 19. Three trials of pushing under the influence traps placed at different locations along the groove are shown in the figure. The panels on the right show the new reward components q_i learnt after being rewarded due to successful realization of the goal partly because of random explorative pushing. In every trial the robot has an experience, an experience of contradiction because of the trap, an experience of exploration which characterizes its attempt to nullify the effect of the trap so as to realize the goal and an experience of being rewarded by the user/self in case of success. This experience is represented in the form of a reward field in the pushing sensorimotor space. For example, in trial 3, what is represented is the simple fact that if the initial position of the ball is around 150 , and the position of the trap is around 65, it is more rewarding to push towards the right and navigate all around the table to reach closer to the ball. These experiences, based on their relevance to the goal being attempted will influence the behavior of the robot in the future.

The motion of the ball due to explorative pushing is shown in blue with the direction indicated by the yellow arrow. Once it is on the other end of the table, it can be easily reached. Case 3 and case 4 are also similar to the first case, however with a different environmental configuration. In the case 2 the trap was placed around 150 and the initial location of the ball shown is approximately 135. In this case there was no exploration at all because the previously existing value field automatically causes the ball to be pushed to the left and the goal was achieved. In fact, the robot was blind about the existence of the hole in the sense that it was not the hole that caused the direction of pushing but the preexisting reward field it had developed earlier. This may be also be a limitation of the approach

because the knowledge is represented more in the form of associations of experiences (like the capuchins, in fact a bit more intelligent as we will see in the next chapter) rather than a still higher level of understanding of the real physical causality. Is there such a still higher level of understanding or is it just associative rules learnt by experience that are exploited intelligently is still an issue of debate, which we will not enter in this section.

What should the robot do with these sequences of new experiences, the experience of a new environment, a contradiction which it did not encounter before while solving similar goal, a phase of exploration to try to find an alternative solution that eventually results in success and rewards? The approach used in this thesis suggests that it should represent them as a memory and in the form of the q_i components in the reward structure given by equation 19. For this the reward that was received on success needs to be distributed to the contributing neurons in the pushing SMS. This distribution is done using the procedure described in 4.6, i.e. in case of rewards, the most distal element receives the maximum reward and all contributing elements receive gradually scaled versions, circular solutions being actively penalized. The panels on the right show the new reward fields (qi) learnt after each trial. In case1 for example, the reward structure representing this experience reflects the fact that if the initial position of the ball is around 180 and the location of the trap is somewhere around 220, it is rewarding to push leftwards. For case 4, it reflects the fact that if the trap is somewhere around 60, and the initial position of the ball is around 150, it is more rewarding to push to the right. We also note that there is no need to predecide how many trials of such learning has to take place. Learning in the system takes place when it is needed, i.e. when there is a contradiction and things are not working as expected. After eight for eight different single trap configurations the behavior produced was intelligent enough that no further training was required.

A possible limitation of the proposed approach is that knowledge is represented more in the form of associations of experiences (like the capuchins, in fact a bit more intelligent as we will see in the next chapter) rather than a still higher level of understanding of the real physical causality. Is there such a still higher level of understanding or is it just associative rules learnt by experience that are exploited intelligently is still an issue of debate, which we will not enter in this section.
What should the robot do with these sequences of new experiences, the experience of a new environment, a contradiction which it did not encounter before while solving similar goal, a phase of exploration to try to find an alternative solution that eventually results in success and rewards?

These additionally learnt qi components of reward field also begin to influence the value field dynamics now and hence the value field structure is no longer constant like it was in figure 19. It changes based on the configuration of the problem. The net reward structure is a superposition of the default plan which was learnt in the previous section and the new experience related fields that were learnt after introduction of traps scaled

appropriately based on their relevance to the currently active goal. This case is similar to the discussion in 4.6, the only difference being that the relevance of an experience is computed based on the location of the hole.

$$R = R_{default} + \sum_{T=1}^{N} \sum_{E=1}^{m} R_E \cdot \frac{1}{\sqrt{2\pi}\sigma_T} \, e^{\frac{-(Trap_T - Trap_E)^2}{2\sigma_T^2}} \qquad (22)$$

Here $R_{default}$ is the pushing reward structure learnt in the previous section. T stands for number of traps. Even though in this section we have not dealt with multiple traps in the trapping groove , we will encounter them in the next chapter. We just note that for multiple traps no additional training was in fact required.

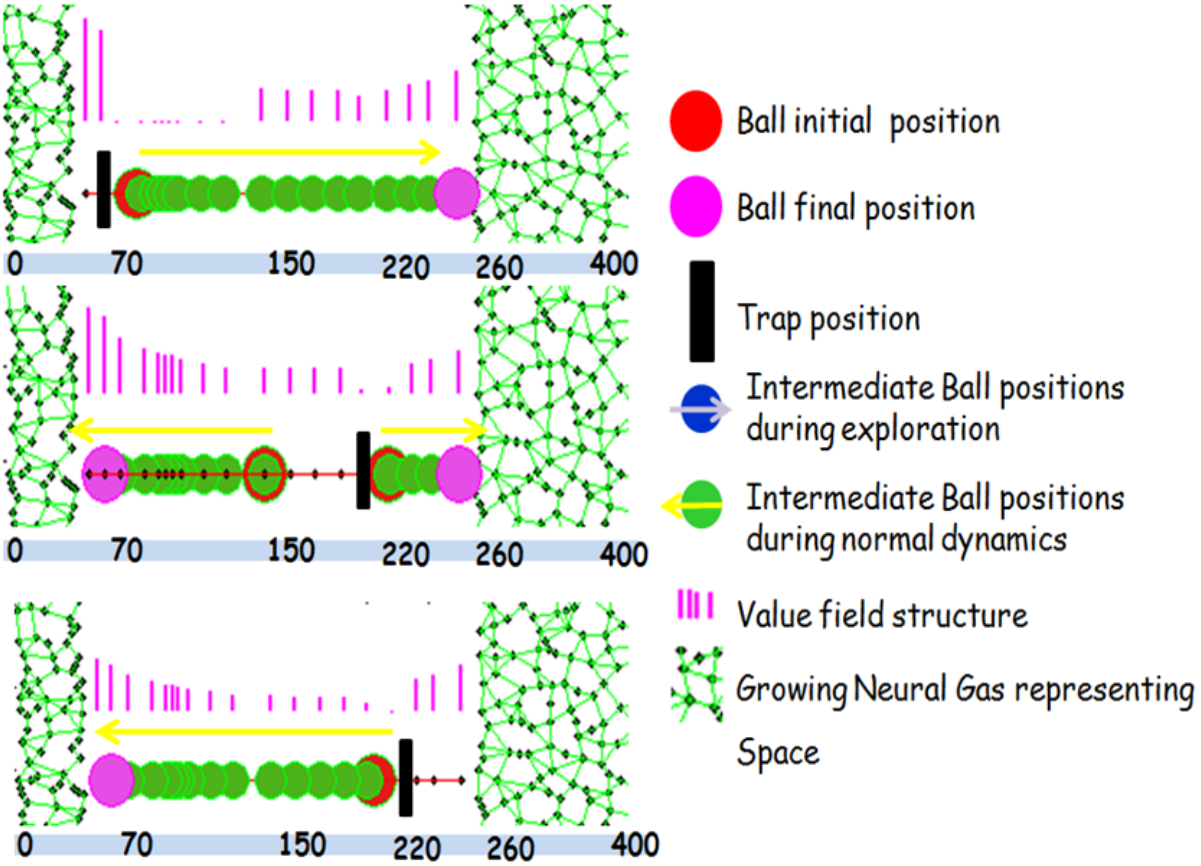


Figure 20. Pushing in the presence of traps in the trapping groove. In the previous cases of pushing shown in figure 19, the value field superimposed on the pushing sensorimotor space was constant. In this figure we can observe goal/trap specific changes in the value field. Experiences encountered in the past and represented in terms of fields are superimposed in a task relevant fashion, to give rise to a net resultant field that drives the dynamics of the system. Also we see that in this case pushing direction is a function of both the relative position of the hole, and the starting position of the reward/ball.

The resulting value field due to superposition of relevant past experiences automatically takes a shape that is sufficient to drive intelligent response in the presence of multiple traps in the trapping groove. E stands for the number of experiences during which new reward field were learnt (eight in our case).R_E is the E^{th} reward field. And the final term computes how relevant an experience E is with respect to the situation considering trap T present alone in the environment. Figure 20 shows examples of the pushing in the trapping groove for single trap configurations, after the learnt reward fields began contributing to the value field structure and hence actively influencing the behavior.

4.10 Unification: Pushing, Moving, Reaching all together

How do the internal models for reaching, spatial navigation and pushing cooperate in simulating a sequence of actions leading towards the solution of a high level goal ? To answer this question let us consider an environmental scenario as shown in figure 21. The goal of the robot is to grasp the red ball placed at the center of the table. Assuming that the robot has a way to figure out that it can create a long red stick using the two small magnetized sticks (we will deal with these issues in the next chapter), it already has an imaginary object in its mental workspace.

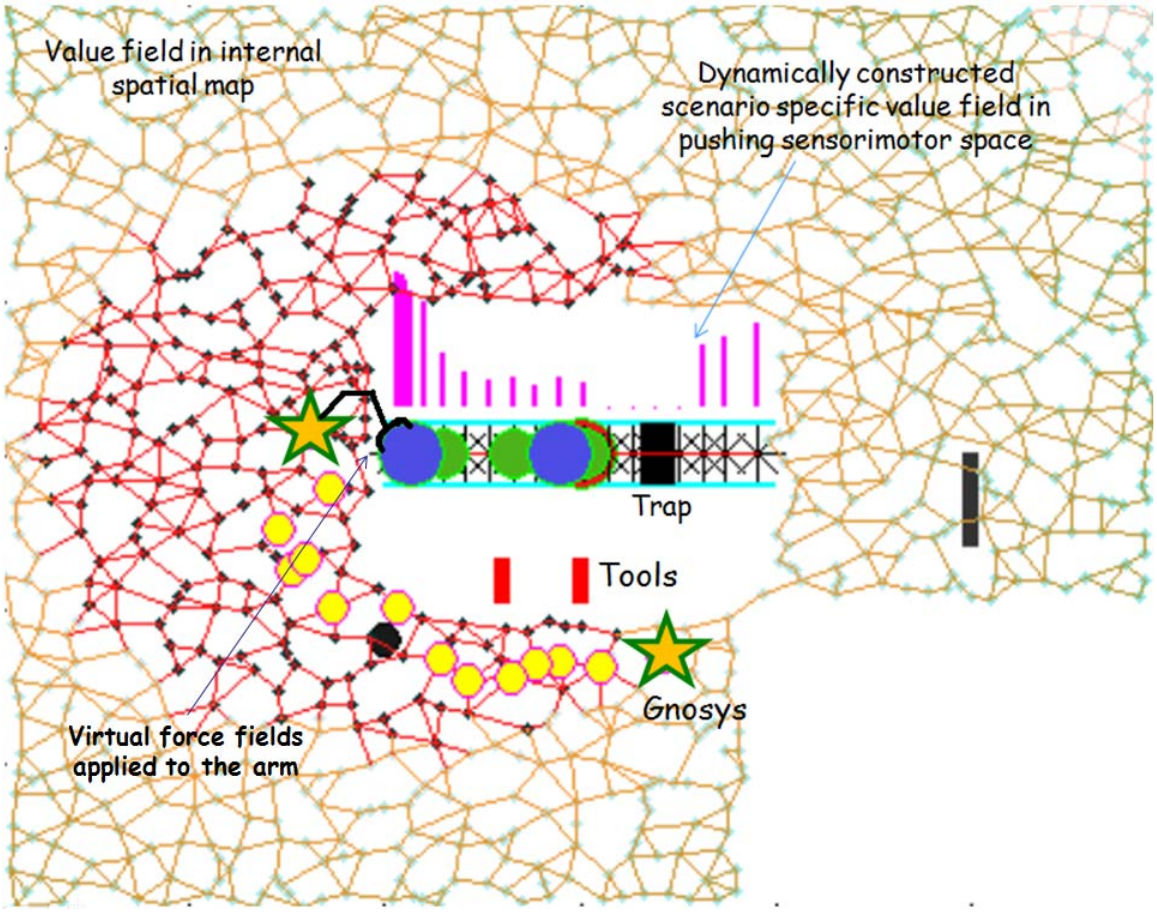


Figure 21. Simulating a simple goal directed sequence of Pushing, moving and reaching.

Assuming further that the robot has figured out that this imaginary object is useful in reaching the ball placed centrally in the table, a trap specific value field on the pushing sensorimotor space can be dynamically generated as shown in the previous section, through superposition of the default reward field and the weighted contributions from previous experiences encountered in dealing with traps. The value field in the trapping groove incrementally generates actions that are needed to push the ball in the most rewarding way. On the other hand these motor activations modulate the lateral connectivity in the pushing SMS and anticipate the position of the ball as the result of the virtual pushing. On reaching equilibrium, the output of the pushing internal model is a set of trajectories: the trajectory of the ball in the SMS and the trajectory of motor actions that is needed to push the ball, in the action space. The anticipated final position of the ball in the trapping groove induces reward excitations on the neurons in the spatial sensorimotor space and triggers the spatial dynamics. The spatial dynamics functions exactly the same way moving in a dynamically generated value field in the internal spatial map, taking into account the set of constraints that are relevant to the task. The output of the spatial dynamics is once again a set of trajectories: the trajectory of the body in the spatial sensorimotor space and the trajectory of motor commands that needs to be executed in order to move the body closer to the spatial goal (i.e. the anticipated final position of the ball which was the output of the pushing internal model).

Once the dynamics of spatial growing neural gas becomes stationary, Gnosys has the two crucial pieces of information needed to trigger passive motion paradigm (forward/inverse model for the arm): the location of the target (predicted by Pushing model), and the initial conditions (location of the body/end-effector predicted by the equilibrium configuration of the dynamics in the internal spatial map). As we saw in the previous chapter the output of the forward inverse model is also a set of trajectories: the trajectory of the end-effector in the distal space and the trajectory of the joint angles in the proximal space. Starting from a mentally simulated initial body/end-effector position (coming from spatial sensorimotor map), the robot can now mentally simulate a reaching action directed towards a mentally simulated position of the goal target (coming from the pushing sensotimotor space), using the forward inverse model for reaching (passive motion paradigm).

*S*tarting from a mentally simulated body/end-effector position (coming from spatial sensorimotor map), the robot can now mentally simulate a reaching action directed towards a mentally simulated position of the goal target (coming from the pushing sensotimotor space), using the forward inverse model for reaching (passive motion paradigm).

In other words, staying where it is, the Gnosys can simulate sequences of 'actions and perceptions' in multiple sensory motor state spaces in order to realize a high level goal in a

complicated environmental set up. As the activity in the SMS space shifts, corresponding motor commands to achieve the necessary perceptual shifts are simultaneously produced in the action space (Goal directed, Incremental forward-inverse activations in the mental space). Further, Pushing, Moving , Reaching, Grasping form a closely connected network, predictions of one slowly driving the other (or providing enough information to make the other mental simulation possible).

Be it the virtual trajectory of the ball simulated by the dynamics of pushing, be it the virtual trajectory of the body simulated by the spatial dynamics, be it the virtual trajectory of the arm simulated through the passive motion paradigm, the common link to all these dynamics is the 'high level goal' that the system eventually seeks to embrace and the several task specific 'blending' of forces that sculpt these trajectories and pull the system to equilibrium/harmony (as evident in the opening quote of this chapter and written by Taoist sage Lao Tzu).

Reasoning about Actions – Computing in the Mind

In fact I learnt this from my wife,
Despite all that I say to her, she does exactly the contrary,
'Yet without retaliation'
This is the **Strategy** of Non Violence.

MOHANDAS.K.GANDHI
(On the Strategy that won India Freedom)

The ability to reason, to orchestrate thought and action in accordance with internal goals, especially when inhabiting unstructured (and sometimes hostile) environments is a fundamental feature of any kind of cognitive behavior. The increasing complexity of our society and economy places great emphasis on developing artificial agents, robots, smart devices and machines which can reason and deal autonomously with our needs and with the peculiarities of the environments we inhabit and construct. We reason when we use experience to go beyond experience, we reason when we use induction to jump from one or two cases to a conclusion about all cases of the same kind, we reason when we make logical deductions, we reason about cause and effect, about situations that do not exist but could exist as a result of our actions in the world, we reason metaphorically from birds to aircrafts, and finally we reason about our own reasoning abilities. Amazingly, only the surface of this seamless ability to reason is reflected in our conscious awareness. Hence, the dual problem of explaining how thought emerges from distributed activity of billions of neurons in the brain has been one of the fundamental quests of neuroscience.

Coherently integrating the information from the bottom (sensory, perceptual, conceptual) with the drives from the top (user goals, self goals, reward expectancy), intelligent agents quickly learn to exploit possibilities afforded by the structure in one's immediate environment to counteract limitations (of perceptions, actions and movements) imposed by one's embodied physical structure. A major part of this process of transformation 'from affordance to action' is known to take place in the mental space wherein the agent, with the help of an acquired internal model, executes virtual actions and simulates the usefulness of their consequences towards realizing an active goal. In the

previous chapters, we came across internal models that basically generate a set three different goal dependent virtual trajectories (in both perceptual and action spaces):

a. the virtual trajectory for an end-effector (and trajectory of motor commands) corresponding to a goal of reaching an external object, taking into account a range of internal, external and temporal constraints (generated at runtime and represented as superimposed force fields)

b. the virtual trajectory for the body in space corresponding to a spatial goal, once again taking into account different environmental constraints that the system learnt based on the rewards/penalties it received for its behaviour (and represented as superposition of different value fields based on their relevance to the current goal) .

c. the virtual trajectory for an external object as a result of the pushing action, taking into account the effect of traps placed dynamically in the trapping groove (once again the net quasi-stationary value field is computed dynamically and represented as a superposition of different 'trap specific experiences')

The computational complexity in the problem of realizing an user goal like 'Reaching a Red Ball' in a dynamic and changing environment results from the fact that before reaching the red ball itself with end effector, there may be several intermediate sequences of real/virtual 'Reaching', 'Grasping', 'Pushing', and 'Moving' etc directed at 'potentially useful' environmental objects, information regarding which is specified by the root goal itself (which was just 'reach the red ball'). So before realizing the root goal, the robot has to 'track down' and 'realize' a set of useful subgoals that 'transform' the world in ways that would then make the successful execution of the root goal possible. Hence, even though the high level goals are simple, the complexity of the reasoning process and actions needed to achieve them can increase more than proportionately with the complexity of the environment in which they need to be accomplished.

So how can the robot reduce/distribute a high level goal into temporally chunked atomic goals for the different internal models? How can the robot do this flexibly for a large set of environmental configurations each having its own affordances and constraints? What happens if the constraints in some environments do not allow the goal to be realised ? Can be robot mentally evaluate the fact that it is in fact impossible to realize the goal in that scenario ? Will it Quit without executing any physical action at all? If yes, does it have a reason to Quit ? and Can we see the reasons that caused the Quitting by analysing the field structure? In this chapter, we seek intuitive, algorithmic and computational answers to some of the questions above.

5.1 Abstraction Yet Again: From 'Force-Flows' to 'Situation-Plans'

All the internal models described in the previous sections essentially had the same structure for acquisition, organization, transformation and use of information. The only difference was that they operate on different sensorimotor variables, moved towards different local goals, under the influence of different value fields and using different body or environmental resources. Further, the mechanisms to deal with constraints, organize goal oriented behavior were also similar, using superimposed fields. Further, as we have seen from the results presented in the previous chapters, every internal model discussed was involved in simulation of an action at the detailed level of forces, flows (displacement), efforts, constraints, geometry, shapes, visual information etc and had its own level of complexity, adaptability, flexibility and capability to deliver intelligent response towards realization its local and well specified goal ('well specified' when seen from the abstract sensorimotor space, when seen from inside the internal model itself, this well specified goal is generally underspecified, for example the goal of reaching a target in space is always underspecified for a redundant robot that is doing the reaching).

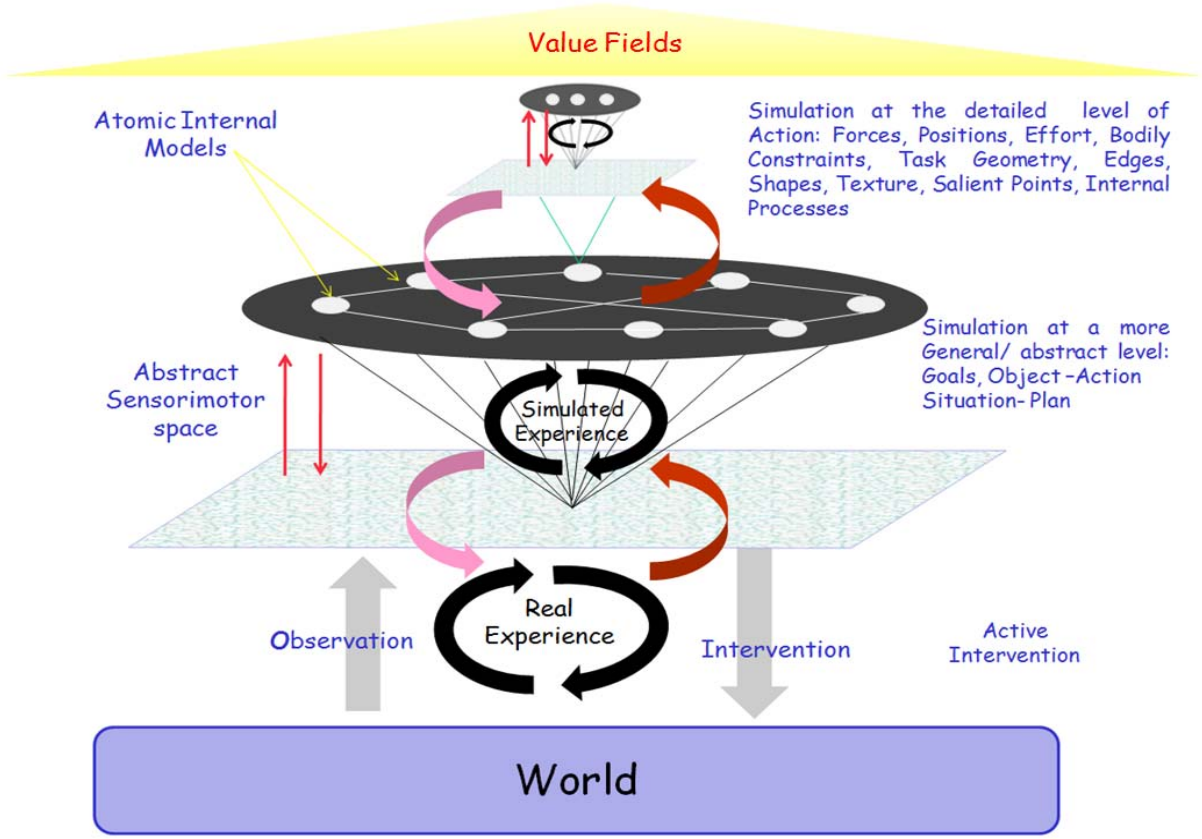


Figure 1. 'Abstraction Yet Again':- The abstract reasoning layer uses the same mechanisms for learning and goal directed behavior organization like the atomic action models. The only difference is that it reasons at the level of situation-plan (or abstract force flows) and not physical force-flows. Hence the various internal models for action act as just variables when seen from the level of the abstract sensorimotor space.

However the task of handling underspecified user goals in ever changing environmental scenarios (like the ones described in chapter 2) requires one more higher level cognitive layer that (when ever needed) segregates the 'useful components' from an unstructured environment and utilizes this information to decompose an underspecified user goal into a sequence of sub goals for the different internal models (and directed towards the discovered useful components), execution of which is expected to change the world, in ways that should aid the realization of the root goal.

As evident from figure 1, in a computational and algorithmic sense, this higher level cognitive layer (abstract sensorimotor space, ASMS hereafter) does exactly the same thing that the atomic internal models do i.e. operate on a new (more abstract) set of sensorimotor variables, grow the state space, lateral connectivity through random motor exploration, move neural activations bidirectionally from SMS to action space (hence forming a trajectory in both perceptual and motor spaces) using the nearly same dynamics, the movement of activity being a function of a goal induced value field in the ASMS. The only difference is that all this occurs at a higher level of abstraction i.e at the level of objects and actions that could be done with them, situations and plans that could be executed from those situations (and not at the level of joint angles, translations, forces, displacements). In this sense the internal models for action act as variables when seen from the level of ASMS. Just like the user/motivation system issues goals to the ASMS, the ASMS issues goals to the internal models for action, which are then either virtually or physically executed by them using information from the environment (for example F-I model for reaching needs the salient points of the target object through vision and so on). If it was a virtual execution, lateral connections in the ASMS are modulated by the abstract motor action, hence causing a shifts in activity in the ASMS equivalent to the perception of an abstract sensory state similar to that which would have occurred if the action was really executed. In case of a physical execution, real sensory information arrives at the ASMS and can be used for growing the map, building lateral connections, comparing with the simulated sensory

> Managing the changing world requires very general and efficient management of the information that it keeps generating, and this information keeps changing every time the robot intervenes in the world, which then has an effect on the ongoing reasoning process, sub goals scheduled for operation, free tools in the world and the overall success in the realization of an active goal. Despite similarity in structure and function, it is only this factor that makes the abstract reasoning system orders of magnitude more complex than the very flexible and adaptive internal models that we encountered in the previous chapters.

information to detect contradictions etc. This leads to the first assumption in the design of the ASMS:

Assumption: Before learning the ASMS, the internal models for action generation must be functional and relatively well developed. There is no restriction on future adaptations of the internal action models. All of them can keep adapting (locally in their sensorimotor spaces) in future based on changes in the world (like the spatial map can be modified due to change in environmental set up, pushing model can incorporate new reward fields, new kinematic chains, force fields can be added to the forward inverse models for reaching etc). Well developed only means that the robot must be able to move autonomously to spatial goals, reach and grasp different objects, use sticks to reach objects, push objects with sticks etc (all these capabilities were demonstrated through the results in the previous chapters). Of course knowing to push, reach, move etc does not imply that the robot also knows how to sequence them when encountered with a high level goal in a random environment, that's the job done by ASMS dynamics.

Managing the changing world in addition requires very general and efficient management of the information that it keeps generating, and this information keeps changing every time the robot intervenes in the world, which then has an effect on the ongoing reasoning process, sub goals scheduled for operation, free objects in the world, and has a direct impact on the overall success in the realization of the active goal. Despite similarity in structure and function, it is only this factor that makes the abstract reasoning system orders of magnitude more complex than the very flexible and adaptive internal models that we encountered in the previous chapters. In the next section, we will enter into the microstructure of the ASMS, analyze what are the various sensorimotor variables, what new connectivity structures are needed in the abstract sensorimotor space that were not present in the previous sensorimotor spaces, how the interaction between these new connectivity structures take place, how information coming from the world (external and internal due to virtual actions executed that gives virtual sensory information) is organized and what further assumptions are made in the design.

5.2 The Microstructure of Reasoning

We begin with the brief discussion on the sensorimotor variables involved in the ASMS.

A. Action Space

Figure 2 shows the basic structure of the action space in the abstract reasoning system. When the abstract reasoning system (ARS) is turned on, there exist nine primitive actions as shown in the figure, using which the robot can begin its motor exploration to acquire the ASMS. Every primitive action in the action space is also given an unique identifier i.e. a six

dimensional vector that is used to refer the action (for all software related information communication) in the abstract sensorimotor space. Even though we begin with nine primitive actions, there is no theoretical assumptions on the number of actions that could exist in the action space or even about their nature (they can merely be software processes defining internal actions like interrupting a goal etc). Usually plans (temporally chunked sequences of primitive actions) enter into the action space as new actions. This occurs when a particular plan is executed so repeatedly that it becomes a waste of computational resources to use reasoning every time the plan is being performed. It is not very difficult to define a criteria for the addition of a new sequence of actions as a primitive element in the action space. Further this pre-definition by the designer as to when a plan should become a primitive does not compromise with the autonomy of the system, in fact it aids it. We will now briefly explain some of the primitives that we have not dealt with in the previous chapters.

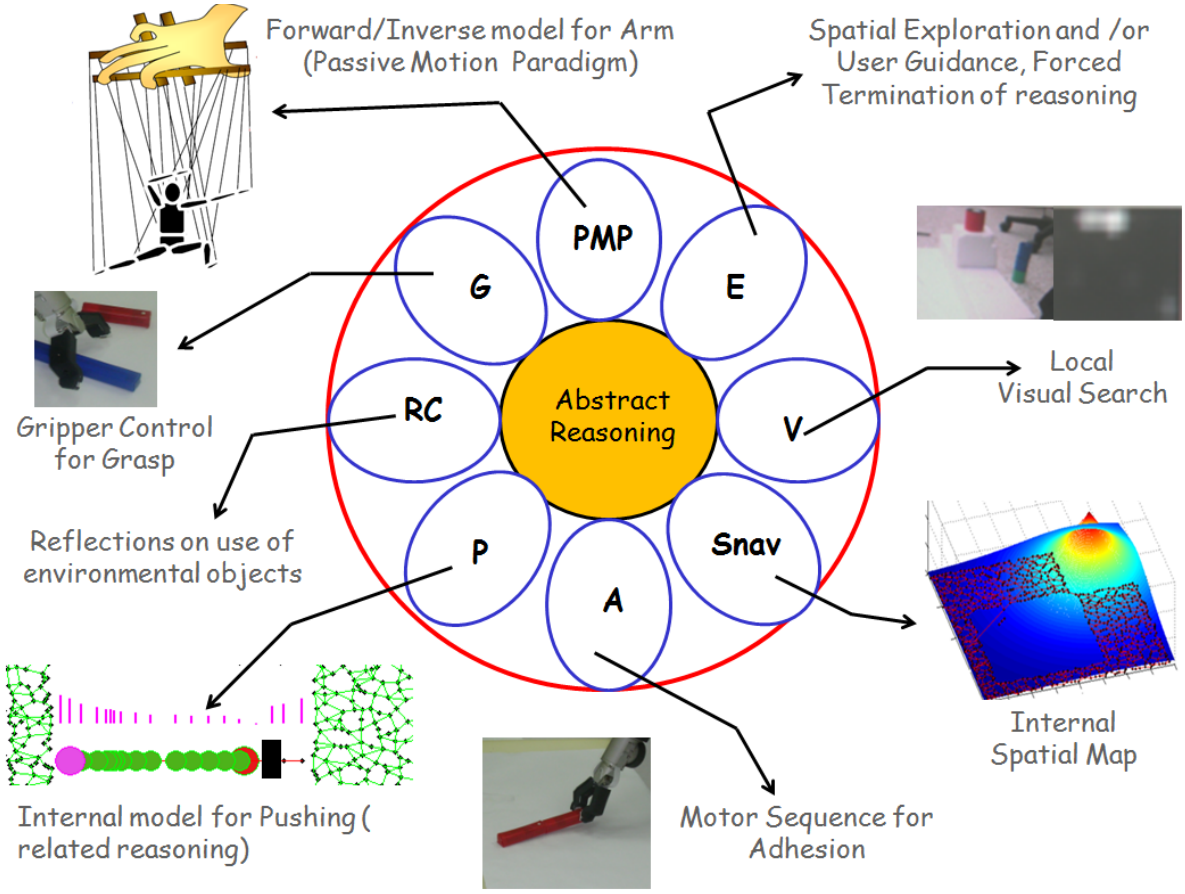


Figure 2. 'Action' related degrees of freedom in the Action space abstract reasoning system (ARS). Every action operates autonomously on its sensorimotor space, is adaptive, plastic and capable of dealing with multiple constraints related to its local scope. There are primitive actions (learnt to an appropriate level in the past through exploration using their local motor degrees of freedoms, like in the previous chapters) and exist before ARS is switched on.

a) PMP: This represents the forward/inverse models for reaching. In addition to its general function of computation of motor commands to reach a goal using the end effector, it is also used to perform virtual experiments to estimate the utility of a tool with respect to a current active goal. A typical virtual experiment would be like, 'can I reach the otherwise unreachable ball using this long blue stick?' In order to verify this fact, the robot must virtually perform this action using the FMF/IMC pair. Virtually performing this action is itself not trivial and requires estimating the length of the stick, updating the kinematics of the arm, doing a virtual reach with the tool and evaluating the convergence, all this assuming that the stick itself will be reachable when the robot has to physically execute this action. The forward/inverse models are also used to do experiments related to the wrist orientation with the a tool must be grasped in order to provide the appropriate length extension to the arm (grasping a same stick different wrist orientations can result in different length extensions, and knowing the best possible orientation is an asset, further based on the wrist orientation with which the tool was grasped the kinematics of the (arm+tool) forward model changes).

b) Grasp: This functionality is not complicated for the GNOSYS robot and just involves closing or opening the gripper whenever the appropriate message is communicated.

c) Pushing: Functionality of this model was described in detail I the previous chapter.

d) Spatial Navigation: This model is responsible for all space related planning, can be triggered through reasoning for execution of a spatial goal, can be triggered internally by the reaching system (PMP) in case reaching has to be done as a combination of body and arm movement, can be triggered by the pushing internal model like the example discussed in section 4.10.

e) Local visual search: This action defines a visual search in the vicinity of the robot (without spatial navigation, like in front of the table for example). This search can be for a specific object or can be just a request for updating the place map with new information through vision (like observing what happens when the robot initiates some explorative action, like trying to couple two small magnetic sticks) . At the level of reasoning this action has three functions 1) trigger the visual perception system with a request for scene analysis, on reception of which images of the present scene is taken, analyzed and the object descriptors are returned 2) Once the object descriptors are retuned, the next task of this action is to update the local place map based on the received new information. Objects not existing in the previous place map are added, the information about overlapping objects are refreshed 3) In case the visual search was for a specific object (like a green ball), then this process

additionally checks the recent place map for this object and returns a message conveying the outcome of the search.

f) Spatial Search 'E': We call it as spatial search in order to not to confuse with 'exploration' (which in our terminology means a random change in the permissible degree of freedom in the action space (joint angle, speed, abstract action whatever applicable) followed by a step of observation and self organization). Spatial search is simplified in the current set up and it involves moving to pre defined spatial locations and doing a local visual search so as to update the place map. So the basic utility of this action is that whenever there are no demanding user goals, the robot can go around exploring in space in order to find what objects are available(that could be used in reasoning scenarios once place map is aware of their availability). Of course in reality the robot must learn areas in space that are rewarding in terms of availability of tools by defining some landmarks, but we preferred not to enter into these issues and focus our efforts on integrating the already sufficiently complex reasoning system on the robot. The reasoning system being highly modular, a more complex model for searching in space can be easily integrated without affecting any of the existing interfaces and functionality.

g) Reflections on Environmental options: This may be thought as an inbuilt drive to experiment (physically/virtually) with objects in the environment. Almost all animals inhabiting in complex environments are known to have this kind of curiosity to play with objects common to their habitats (like twigs in the case of new Caledonian crows, sticks, stones and several other objects for the case of the chimpanzees). The robot uses this action to evaluate the use of any available object in the place map (cylinders, sticks) to reach otherwise unreachable objects in the environment. Of course, having an inbuilt drive to use objects in the environment to reach other objects just deals with the software related communication to make this action sequence possible (taking a random object (say T1) from the place map, extracting its length, updating the kinematics of the arm based on this information, and then initiating the FMC/IMC relaxation to reach the location of another object (say T2) with T1 as the end effector). However, it does not (and cannot) specify the utility of any object for any scenario at any point of time and the robot has to discover the utility on the spot through virtual tests with different available objects (as scenario is never constant, different objects may be present at different times and may have or not have any utility value). More over it is also not specified when to use this action in the first place, it is up to the robot to randomly try this action during the execution of any root goal (like reaching an unreachable ball) and if successful in reaching the ball with a tool detected after using this action, be rewarded for its attempt. This

experience can be generalized in ways such that this behavior will be encouraged in similar situations.

h) Adhesion: This is another inbuilt motor sequence available as a primitive action applicable to any group (of two) objects from the environment (say T1 and T2). This is the sequence of first grasping T1, then extracting the orientation (with respect to the body) of the second T2 (different objects can be placed at different orientations in the table) and then reaching the second object with T1 held in the gripper such that the edges are close by (so as to allow magnetic coupling if the both the objects T1 and T2 are magnetized). This action encourages the robot to try coupling different objects and see what happens and is analogous to Betty's playing with flexible plastic pipe cleaners an year before she was tested on the task that involved making a hook out of the straight wire to pull out her food basket form the transparent vertical tube. In the similar fashion, the robot also has a motivation to randomly pick couples of objects and try this action as shown in figure 3. It is during this playing with sticks of different types, the robot can discover (and represent in the ASMS) the additional affordance of the small magnetized red sticks i.e. to make still longer sticks (and also discover that if it tries the same action on a red and blue stick (or any other object fro that instance) nothing eventually happens and the world is still the same i.e. no new objects are found in visual analysis). We wish to note here that, even after gaining an experience of making a long red stick using small magnetized sticks and representing it in the ASMS, there is a further problem of retrieving this experience in the context of a high level goal (like Betty used its previous experience to make a hook out of the available tool during the time of the experiment) and then seeing the consequence of this action (creation of an imaginary object in the place map through motor modulated connections), seeing the utility of this imaginary object in the context of a the active goal (like the possibility of successfully reaching, pushing a ball with this imaginary object by using action h), then physically creating this object for real use.

Figure 3. Two trial examples of the execution of Adhesion action.

With respect to the atomic actions, we also assume that even though all of the primitives can be randomly triggered during random exploration while learning the ASMS, only those actions that have the sufficient (sensory) arguments necessary to be physically executed are triggered. For example, If an object is not visible then it cannot be reached (reaching needs the location of the object that has to be reached). Similarly, if an object is not reachable then it cannot be grasped. In such cases explorative reach and grasp actions cannot be executed on the objects respectively and some other action is randomly selected for execution (like going for a spatial exploration, quitting which is generally discouraged by penalty if selected).This assumption may seem trivial for a human, but it's necessary to make this explicit in software in order to initiate randomly only those actions that are possible from a particular sensory situation.

B. Sensory Datagram structure for ASMS

After describing the primitive action degrees of freedom in the ASMS, we now move on to the sensory information structure in the abstract sensory motor space. The sensory state vector in the ASMS is a combination of both information coming from the environment and the body and summarizes the complete state of the robot at any point of time. Further, the state vectors are allowed to represent both situations as well as objects. The basic structure of the sensory datagram is shown in figure 4.

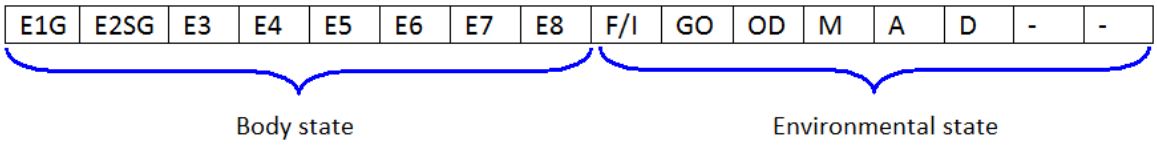


Figure 4. Sensory datagram structure in the ASMS.

As seen in figure 4, the instantaneous state vector is a composition of 16 inputs, 8 each coming from the body and the environment. The environmental frame can accommodate a maximum of eight arbitrary environmental objects at any time. Every environmental object has its own unique identifier (based on its shape, color and size as returned by the vision system) that is used to represent that object. The coding scheme we followed for the environmental objects was very simple and is illustrated below:

- Color: is detected by the visual systems and is coded as follows (Red=1, Blue=2, Green=3)

- Shape: is a feature detected by the vision system and is coded as follows (Stick=1, Ball=2,Cylinder=3, Cube=4)

- Size: This parameter is dynamically computed by the 3D reconstruction system presented in chapter 3. Since the estimate is only approximate, in order to have uniformity in representing size, the estimates are then appropriately rounded off to either 10,15,20 or 25 cm. So if an approximate estimation of size was 13.5 cm it is rounded off to 15 cm.

Using this simple coding scheme, a small red stick will be represented as 1110 (Color, Shape, Size); a red ball will be represented as 121 (size for all balls is 1, and only color changes); a long blue stick is represented as 2120 and so on. One important thing to note here is the fact that since there are multiple objects of the same kind, they all are represented similarly. For example, since we have two small red sticks, both are represented as 1110 only. The only way to know that there are two of them available is through the spatial location maintained in the place map (which can also be fooled if the user intervenes and keeps the same red stick in two different locations while the robot is exploring to update its place map). The other way to know that there are two red sticks is a past memory of playing with two of them. We will deal with this issue in the coming sections, because red sticks are extremely useful objects from the view point of the robot.

This invariance was introduced based on the simple observation that the plan for reaching a red ball (in some environment) will be exactly the same as for reaching a green ball in the (same environment), provided that the robot is reasoning with the same knowledge.

Till now we saw how objects are recognized by the reasoning layer based on the object descriptors returned by the vision system. While representing objects in the ASMS is important to form object action complexes (in the form of a hetero associative memory, like a stick can be used to push etc), considering the fact that there are many objects that can appear in any combination at any time in the environment, we need to further constrain the sensory state vector to acquire some form of invariance when it comes to valued states that are used for goal directed planning. Valued stated are sensory states in the ASMS that are associated with the virtual simulation of a goal directed action sequence under the influence of a value field (like in previous chapters). This invariance was introduced based on the simple observation that the plan for reaching a red ball (in some environment) will be exactly the same as for reaching a green ball in the (same environment), provided that the robot is reasoning with the same knowledge. In other words, the representation in ASMS should be made independent of the 'goal object' being actively pursued by the system, since plans will have the same structure independent of the object being pursued, provided the environment is constant. Hence for any ASMS state involved in value field based goal directed planning, the first element in the environment state is constant (and equals to 1 and not the real object descriptor of the

goal). In this case from where can the actions receive/send more information about the goal object? (for example a visual search for any object needs the object descriptor of the goal it is looking for and not a constant value like 1). Whenever the first element in the environment tree is constant, it indicates that the real object descriptor for the goal is stored in the goal space (another memory structure we discuss shortly) and is the first element in the goal space if A=0 in the body tree, that indicates the goal being attempted is the root goal. A=1 indicates that the goal being attempted is a sub goal generated by the reasoning system to support a root goal, and hence the location of the object descriptor is the last element in the goal space. In simple terms, in this way even if the goal is to grasp a red stick or a green ball or whatever, the representation in the ASMS that is used for planning is independent of all these details, all that it indicates is the location in the goal space where the real object descriptor for the goal can be found. We will present a simple example of this once we deal with the body part of the state vector (that is needed to understand how presence of sub goals are reflected in the state vector).

We wish to note here that all we have done with respect to the sensory datagram representation is define the structure in which the information coming from the world is organized to form the state vector. We have not defined/controlled the content which is a function of the environment itself.

The first element in the body vector is the state of the forward/inverse model for reaching, the second element is the gripper status, third is related to vision, fourth is an input coming from the motivation system indicating the engagement/termination of the goal, the fifth term is the activity vector that indicates the presence of sub goals in the system. The activity flag once again helps introducing invariance in the ASMS representation at the level of a goal and a sub goal. This comes from the fact that at the level of reasoning and execution, there is no difference between a goal and a sub goal, and the reasoning process employed to reason for a root goal of grasping a red ball, is the same that is employed for a subsequent sub goal of grasping a blue stick that was generated during the process of reasoning for the root goal. The only fact the ASMS must represent is that if it is currently executing a sub goal, then there are other goals in the goal stack that needs to be realized in order to realize the root goal. This is represented by the activity flag. If the activity flag is '0' that means that there are no sub goals linked to the root goal. If the activity flag is 1, that means that there are sub goals linked to the root goal, and the current sub goal being executed is related to the bottom most object located in the goal space. How many sub goals are linked to the root goal is a function of the environmental complexity and cannot be known/predicted in advance.

Finally we wish to note here that all we have done with respect to the sensory state representation is to define the structure in which the information coming from the world is organized to form the state vector. We have not defined the content which is a function of the environment itself. This implies that there is no hand coding of any representation in the system. Representation are created directly based on the sensory information arriving from the environment based on which the ASMS grows with respect to time (like the SMS we dealt with in the previous chapter). The only thing we defined was the structure in which the information is arranged, which makes computing on the representation easier and efficient. Before we proceed further, we will present a few possible examples of state representation and explain what they mean (light blue = body state; yellow = environmental state).

1) There are two red sticks

E1G	E1SG	E1	E1	E1	E1	E1	E1	F/I	GO	OD	M	A	D	-	-
1100	1100	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-	-

2) There is a long red stick

E1G	E1SG	E1	E1	E1	E1	E1	E1	F/I	GO	OD	M	A	D	-	-
1120	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-	-

3)The goal object is reached successfully by the end effector

E1G	E1SG	E1	E1	E1	E1	E1	E1	F/I	GO	OD	M	A	D	-	-
1	-1	-1	-1	-1	-1	-1	-1	0	-1	1	-1	0	-1	-	-

Note that the first element in the environment tree is 1 that indicates a valued state in the presence of a user goal. What the goal object was can be found by looking at the topmost element in the goal space. As we mentioned earlier this way of representing valued states independent of the object descriptor of the goal object was motivated by the fact that, the plan learnt/outputted by reasoning system for a goal of reaching a red ball and reaching a green ball will be the same (and independent of the goal object) if the goal was executed in the identical environment, with identical knowledge. Also note what end effector reached successfully means in the system: It means that the goal object was detected by vision (OD) and the F/I model converged successfully in reaching the target.

4)The sub goal was reached using the end effector

E1G	E1SG	E1	E1	E1	E1	E1	E1	F/I	GO	OD	M	A	D	-	-
1	-1	-1	-1	-1	-1	-1	-1	0	-1	1	-1	1	-1	-	-

The only difference with respect to the previous example is that A=1, which indicates that the goal being attempted is a sub goal generated to support some root goal, and the object descriptor of the object on which this goal is attempted is the last element in the goal space. It is the last element because, during reasoning sub goals are formed top down, but during execution they are executed bottom up, every sub goal supporting the goal above it in hierarchy.

5) There is a red ball, a small red stick and a long blue stick

E1G	E1SG	E1	E1	E1	E1	E1	E1	F/I	GO	OD	M	A	D	-	-
121	1100	2115	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-	-

It should be quite clear from the few examples presented, that even though the structure of organization of information is well defined, the content is not defined by the end user, and based on what content fills the structure, the meaning of the state vector will be different. This also implies that the ASMS is domain agnostic and growing in nature. Based on the nature of the high level goal and the subsequent explorative experience strategies can be learnt by the robot to handle them in a flexible fashion.

Based on the nature of the representations in the ASMS, we must also note the fact that for the abstract sensorimotor space, there is no direct mathematical relation between any two sensory states unlike the internal models presented in the previous chapters (for example : there is no direct relation between a sensory state in ASMS representing a goal to reach a red ball and another state vector in ASMS representing the presence of a long blue stick in the table, even though this state may be valuable in the context of the goal of grasping the red ball , or equivalently the relationship between a banana and a stick for the case of the chimp).

To sum up, the sensory state is constructed using information coming from the body and the environment. All components can take any value based on the corresponding state of the body and the environment in which the state is constructed. In the case of a valued state, that represents a situation arising as a result of an execution of an user goal (and not just an arbitrary collection of objects), the representation in the ASMS is independent of the object descriptor representing the goal object. All that the representation indicates it the location in the goal space (top most or bottom most position based on the root goal or a sub goal, that is indicated by the activity flag) where goal object and its various features can be found.

Based on the nature of representations in the ASMS, we must also note the fact that for the abstract sensorimotor space, there is no direct mathematical relation between any two sensory states unlike the internal models presented in the previous chapters (for

example : there is no direct relation between a sensory state in ASMS representing a goal to reach a red ball and another state vector in ASMS representing the presence of a long blue stick in the table, even though this state may be valuable in the context of the goal of grasping the red ball , or equivalently the relationship between a banana and a stick for the case of the chimp). Hence in the ASMS it is the indirectly learnt value of any state (in the context of the active goal), that defines the closeness between two sensory representations.

C. Monitor process

Since there can be many sub goals running in the reasoning system during the process of realizing the root goal (based on the complexity of the environment that is not in our control) we need a parallel process running along with reasoning that keeps track of the progress in the ASMS regarding the realization of the each of the goals. This information is crucial in

a) managing priorities of the different sub goals scheduled for execution in the goal space (to handle temporal aspects of plan sequence)
b) terminating the goal when it completes it execution (virtual) and handling the control of the ASMS to the next scheduled goal/sub goal with highest priority
c) to initiate the learning processes in case the goal was being attempted through random action exploration (in which case the reward given by the user must be distributed to the various sensory states I the ASMS that contributed towards the solution), the global knowledge space must be updated etc.

Whenever a new user goal/sub goal is created during reasoning, the termination conditions for that goal is initialized in the monitor system. Every goal has a separate identifier called as the goal number that is used by the monitor system to identity the goal. Goal number is just a scalar, the root goal is always 1 and the sub goals that are generated will be numbered in the ascending order. Since the user goals are currently limited to reaching and grasping different objects (in any environment), the sensory inputs involved in the definition of the termination conditions are shown in figure 5.

From the perspective of reasoning, notion of a goal is sensorimotor as well. It is sensory because it refers to an abstract perceptive state in the ASMS that the reasoning dynamics should converge at, as a result of the actions executed in the motor space (random trial and error during exploration or planned during reasoning). It is motor because, the goal can be thought as an under constrained version of an action itself, for example the goal 'reach red ball' already indicates the action that the system needs to use to achieve the desired sensory state.

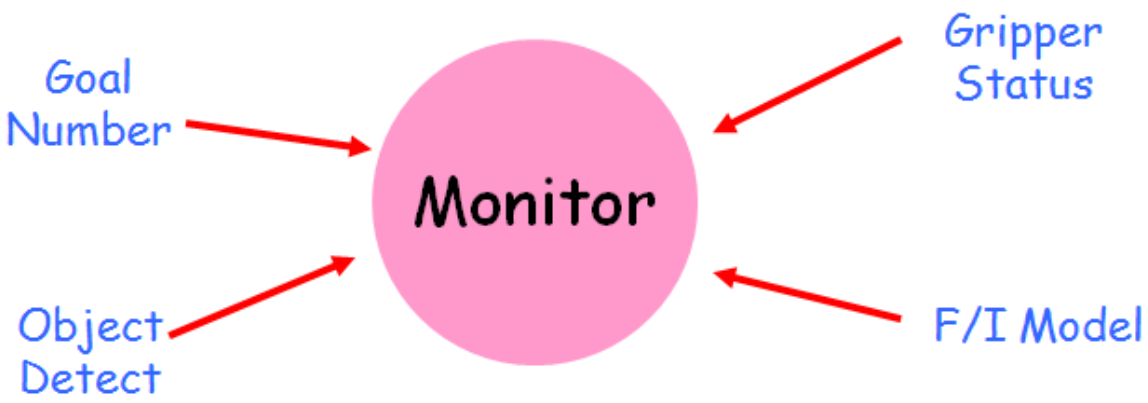


Figure 5. Definition of termination conditions for a goal.

A reaching goal is terminated if the message from vision signals the successful detection of the target object, the reaching system converges to equilibrium and signals success. A grasp goal in addition to the above two conditions also requires the gripper status to be checked to verify if the object was grasped successfully. If the system had two goals of reaching small red sticks placed at different locations, what differentiates the two goals is the goal number. Even though our priorities were to initially attempt only reaching and grasping goals in arbitrary environmental scenarios that involve sequences of tool use/tool making, there are no theoretical restrictions in the architecture that limit us from having more complex goals in the future versions of the architecture. One new root goal we have introduced recently is the stacking goal, that in simple terms is composed of sequences of reaching, the termination conditions of which additionally require the measurement of the height of the stack achieved, using 3d reconstruction system. This goal is quite general and can be to stack 6 small cylinders, 3 large cylinders whatever. More over the components (cylinders) may be placed in the trapping groove in which case the task of collecting the components itself is equivalent to a complicated sequence of reasoning (using sequences of reach, grasp like goals defined earlier).

From the perspective of reasoning, notion of a goal is sensorimotor as well. It is sensory because it refers to an abstract perceptive state in the ASMS that the reasoning dynamics should converge at, as a result of the actions executed in the motor space (random trial and error during exploration or planned during reasoning). It is motor because, the goal can be thought as an under constrained version of an action itself, for example the goal 'reach red ball' already indicates the action that the system needs to use to achieve the desired sensory state. However, it is the locally available environment that imposes constraints on weather this action can be directly executed or will there be several sequences of sub actions that need to be executed before executing the reach action itself.

In this sense, the goal of reasoning is to discover the constraints imposed by the locally available environment, and transform an under constrained action hinted by the goal into a

set of environmentally feasible actions that when physically executed will nullify the constraints imposed by the environment/body towards realizing the action suggested by the root goal.

D. Place Map

Any change occurring in the world as a result of physical actions initiated by the user or the robot and mental actions initiated in the world by the reasoning system is reflected in a dynamic memory structure called the place map, shown in figure 6. The place map has a matrix like structure, every row representing an object and its known attributes at that point of time. For every object, its description (in terms of color, shape and size), the salient points of the object in the left and right camera images (5^{th} to 8^{th} elements), the approximate length extension that the object gives after grasping (9^{th} element), a piece of information indicating how close the object is with respect to the current position of robot, required for plotting Venn diagram like representation of the world in the GUI (10^{th} element), the temporary utility value of the object (11^{th} element) and the global spatial location of the object in the playground(12-14^{th} element).

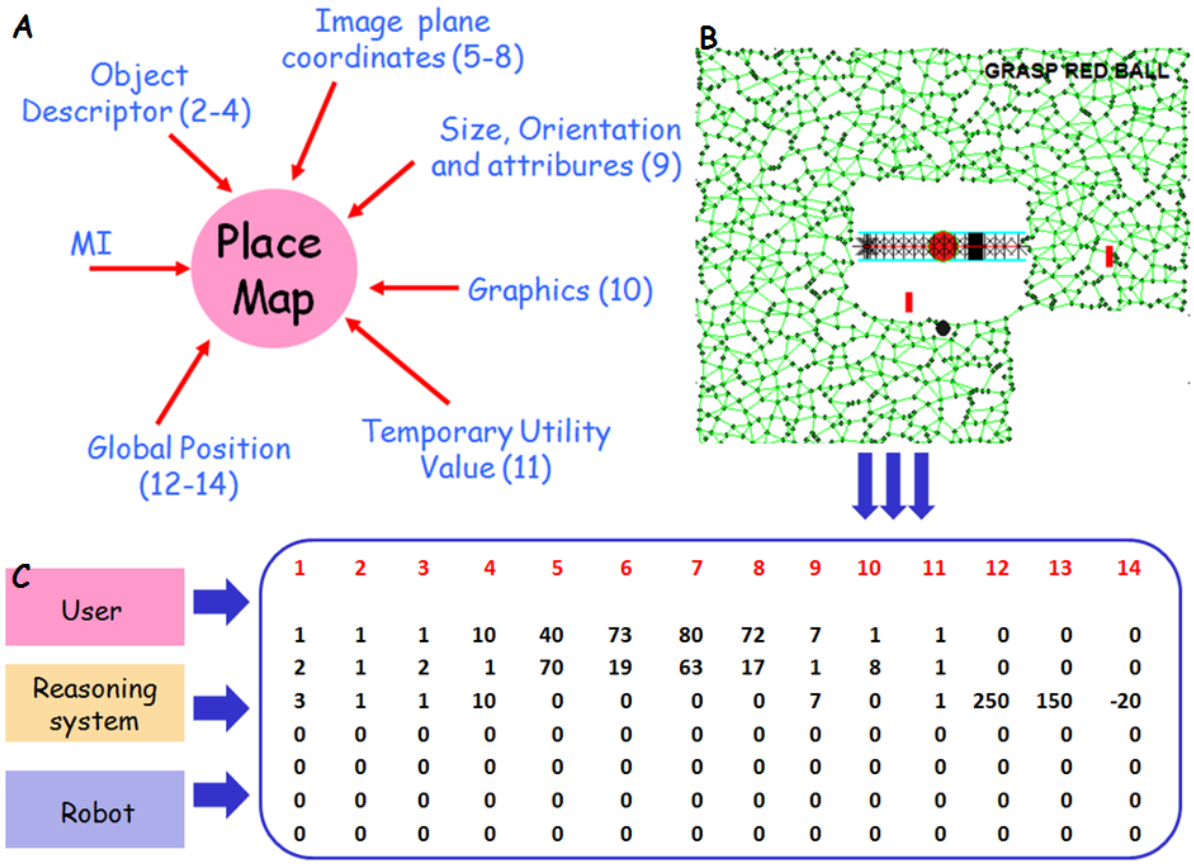


Figure 6. Panel A shows the different streams of information entering into the place map. Panel C shows the snap shot of the place map for the environmental scene shown in panel B. Place map is a dynamic memory reflecting changes in the world based 'n the actions initiated by the user or the robot. Every element in the structure can change with respect to time, there is no restrictions on all information to be known at all times. As we can see in panel C, for the second red stick the arguments corresponding to the location of that object in the image plane of the cameras is not known (or are just 0's).

The temporary utility value is either defined by the user in case he wants to prohibit the robot form using certain objects in the world (as a means to encourage it try using something else or make a tool for example) or is modified by the reasoning system if some object is already under use (physical/mental) by some goal (for example if goal 1, sees the utility of a stick, it takes possession of it by rewriting the TUV as 0, which means that some other goal running simultaneously cannot get access to the stick, till the time the object is freed again by the executing goal).

As seen in figure 6, the camera plane coordinates of the second red stick (3^{rd} row) is not known which implies that no arm related manipulation on the object can be done unless this information is available. It is the task of the reasoning system that when some arm manipulation needs to be done on the second red stick, it initiates a spatial navigation using the spatial map, moves closer to the second red stick, trigger a visual search and update the place map to provide the necessary arguments to the forward inverse models to carry our arm based manipulation on the red stick. Even though this memory has a simple 2d structure, it is one of the sites in the reasoning system where any minute change initiated on the world by any process executing inside the robot or the user is immediately reflected. Hence the place map remains in continuous observation by the reasoning system at all times. From the view point of the reasoning system, place map is also the structure that holds a concise description of what the environment affords/does not afford and hence is also a site for reflections on possible environmental objects that could be used as tools.

E.Goal Space

Actions generated by the reasoning system are directed towards realizing goals. Goals (and sub goals) are directed towards objects in the world (organized by the place map). Since we have no control over the world, we have no control over the amount of goals (sub goals), that need to be directed at objects in the place map/world in order to realize the goal. Hence we need a special dynamic memory just like the place map, that organizes/holds information related to goals under execution in the reasoning system. In simple terms, just like the place map organizes information acquitted about all objects present in the world, the goal space organizes information about all the 'useful objects' in the world/place map on which actions need to be initiated in order to realize the high level user goal. Figure 7 shows the structure of the Goal Space.

The goal space is empty when the reasoning system is initialized with a new user goal. We note here that the place map is not empty when the system is initialized, because it just holds the state of the world that presently exists and may have several objects registered (of which the robot knows information) in it based on the previous visual searches conducted (or actions executed) in the environment. In this sense the entry and exit of information in the place map is continuous time and is independent weather the system is executing a new goal or doing something else.

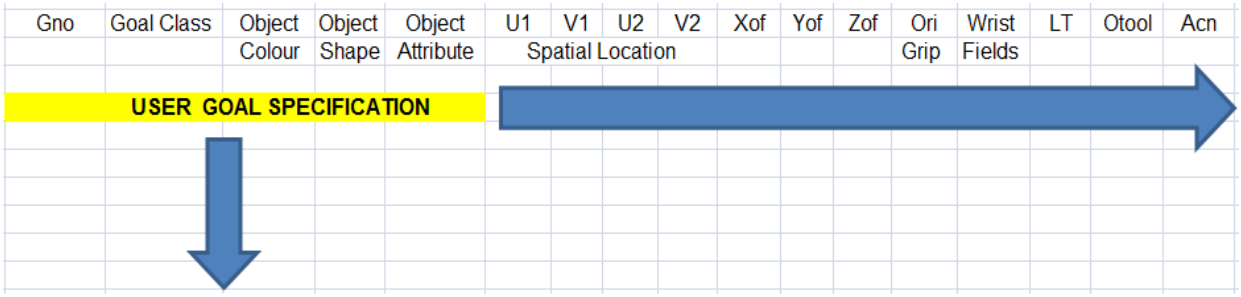


Figure 7. Structure of the Goal Space. Just like the place map dynamically holds information about all the objects in the world, the goal space holds information about the set of 'useful objects' in that world on which sub-goals have been set by the reasoning process. The goal space is the complete execution scope of the reasoning system, all actions executed physically or virtually on useful goal objects generate new information regarding that object when then keeps accumulation in the goal space, which then affords some other action to use this new information as its arguments and run its virtual execution, generating more information and so on. For example, a visual search run on a potentially useful object, say red ball, will generate information about its spatial location, which enters the goal space: the forward/inverse models need exactly this information to run the PMP.

In this sense, in an unstructured world, while the place map holds world specific information the goal space holds goal specific information, and the whole process of reasoning is about transforming the world specific information in the place map into useful goal specific information in the goal space. Reasoning clusters the world into those which are potentially useful, and those which do not afford an utility value at that point. Useful objects become subjects of sub goals, and enter into the goal space.

As a new user goal (root) is presented to the robot, the goal space is initialized with this information (and shown shaded in yellow in the figure). A typical example of such an initialization is '1 Grasp 1 2 1' simple means, this is goal number 1, the goal class is grasp (can be communicated linguistically to the goal space) and 1 2 1 means the red ball. This is all that is presented to the system. All remaining information 'horizontal and vertical' are subsequently filled by the process of reasoning and the actions generated by the system. The columns represent information about the particular goal object, and rows represent sub goals generated by the reasoning system on potentially useful objects in the place map, in

order to realize the root goal. In the simplest possible scenario like reaching a red ball that is reachable and is present in front of the robot, there won't be any more rows in the goal space, as the is no need of a sub goal. But as the environment becomes more and more complex information increases both horizontally and vertically in the goal space.

In this sense, in an unstructured world, while the place map holds world specific information the goal space holds goal specific information, and the whole process of reasoning is about transforming the world specific information in the place map into useful goal specific information in the goal space. Reasoning clusters the objects present in the world into those which are potentially useful, and those which do not afford an utility value at that point. Useful objects become subjects of sub goals, and enter into the goal space. Since all actions executed by the system are goal directed (and initiated on useful/rewarding objects), all actions take information from the goal space and send information to the goal space (as shown in figure 8).

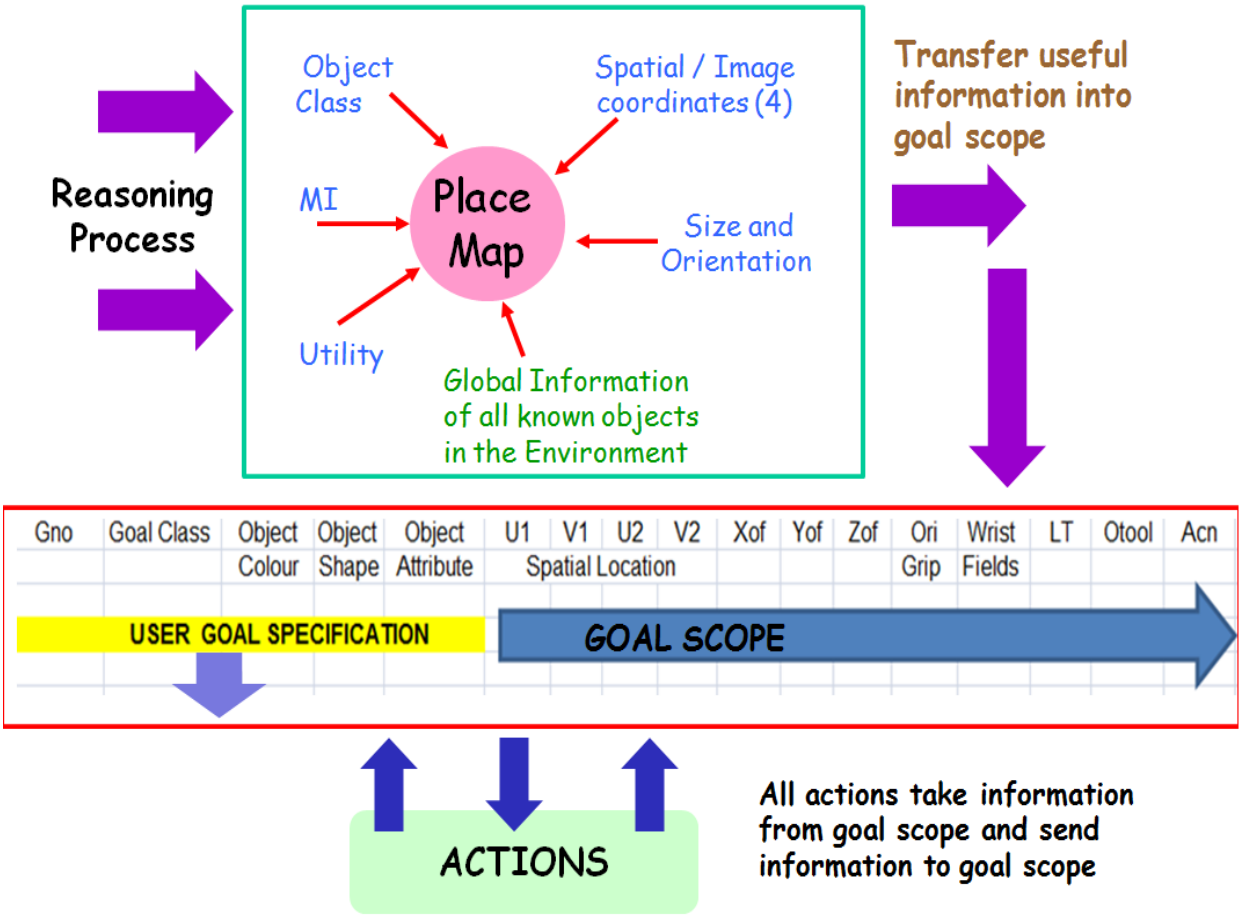


Figure 8. Interactions between the Reasoning process, Place map and the Goal space. Reasoning clusters the objects present in the world /place map into those which are potentially useful, and those which do not afford an utility value in the context of the currently active goal. Useful objects become subjects of sub goals and enter the goal scope, from where they become subjects of different actions, that may eventually lead to their successful realization, that changes the world in ways that aid the realization of the root goal and so on. In this way the

global system behavior with respect to information management does not change even if the world (internal and external) is continuously changing.

Goal in this sense is the global information organizing/binding agency in the architecture. What we gain is a well structured mechanism to useful move information from an ever changing world into the local scope of the active goal. Another important advantage gained is the fact that even if the world is continuously changing, goals are changing, internal models are undergoing adaptations, the global system behavior with respect to organization of information does not change .

F. Global Knowledge space

In the previous five sub sections, we described the basic sensory and motor variables on which the abstract reasoning system operates (ASMS) and how information coming from changing world is routed to different dynamic memory structures that aid the operation of the reasoning system in terms of efficient information management. In this section, we will look into how activity moves bidirectionally between the ASMS and the action space when a user goal is issued to the reasoning system. As we saw in the previous chapter, the lateral connectivity, sensory weights and the motor modulated lateral connectivity were together responsible for moving activity in the spatial SMS and Pushing SMS under the influence of a value field generated by the goal. Since the ASMS also computes in the similar fashion as the previous internal models (with the only difference being that the sensorimotor variables are more abstract) it should be quite clear that these connectivity structures play an important role in the dynamics of the ASMS also. In addition to these connectivity structures that were already present in the previous internal models, we introduced two more types of connections between neurons in the ASMS in order to achieve greater flexibility while reasoning in the abstract world. All the previously existing connections i.e sensory weights, lateral connections and the motor weights that induce top down multiplicative modulations in the SMS are maintained and have the same functionality in the ASMS. The newly introduced connections are as follows:

Intermap connections.

Intermap connections associate sensory states present in the ASMS to the motor actions that are possible from that situation. In simple terms, intermap connections aid formation of object-action or situation –action complexes. Unlike the motor modulated lateral connections that project top down and cause shifts of activity in the sensorimotor space based on current motor activations (similar to the pushing SMS and spatial SMS models), the intermap connections project upwards from the ASMS to the action space and associate with the set of possible motor actions from a currently active sensory situation. While action selection based on a value field dynamics does not explicitly require intermap connectivity as

we saw in the previous chapters, introduction of intermap connections in the ASMS was necessary because there can exist many isolated experiences involving 'object-action-consequence' loops in the ASMS that may not have any value associated with them initially, as they were just learnt/observed accidentally through explorative play (while trying out primitive actions randomly). A typical example is shown in figure 9. During the phase of randomly truing out primitive actions on the available objects, whenever the robot initiates the adhesion action on a couple of red sticks, it will discover the additional affordance provided by the group of red sticks. Now we need some mechanism to store this experience as a memory trace, that could be retrieved whenever necessary (based on a sensory cue).In simple terms, we need a way through which when given a cue the robot can estimate/remember the set of actions that are possible with that object (based on its previous experiences with that object) i.e for example a blue stick or a large cylinder can be used to push objects, two small red sticks can be used to make a long red stick, it cannot do anything with a small blue stick and a small red stick and many more similar cases.

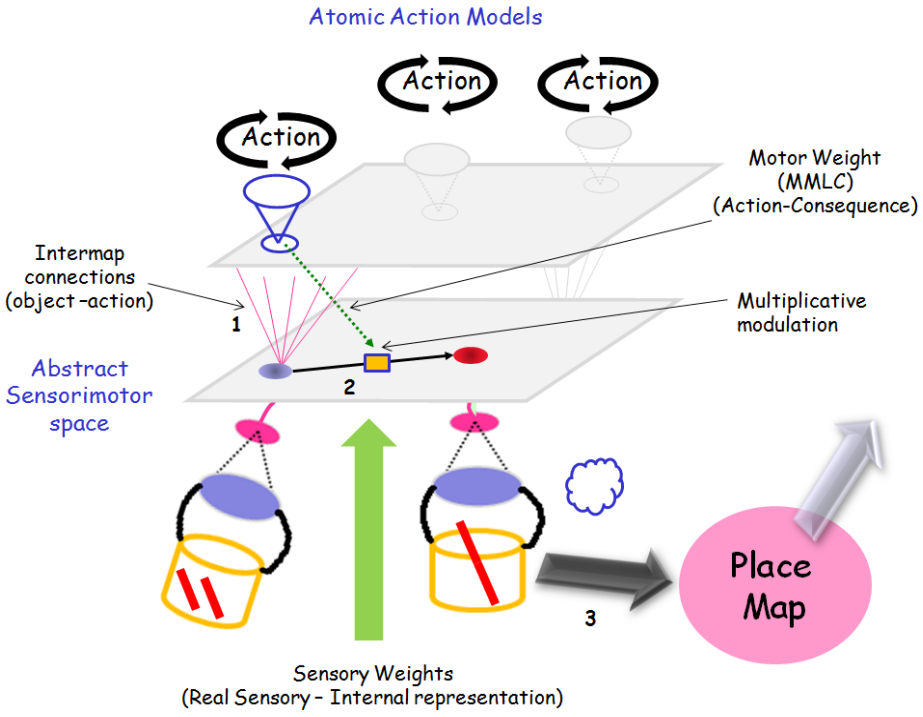


Figure 9. Unlike the motor modulated lateral connections that project top down and cause shifts of activity in the sensorimotor space based on current motor activations (similar to the pushing SMS and spatial SMS models), the intermap connections project upwards from the ASMS to the action space and associate with the set of possible motor actions from a currently active sensory situation. While action selection based on a value field does not require intermap connectivity as we saw in the previous chapters, introduction of intermap connections in the ASMS was needed because there can exist isolated experiences involving 'object-action-consequence' loops in the ASMS that may not have any value associated with them initially, as they were just learnt/observed accidentally through explorative play (while trying out primitive actions randomly).

We also note here that we only need some mechanism to make the bottom up (Situation-action) association. The motor modulated lateral connections that are already defined in all the previous models can be used to foresee the consequence (this can be easily done by making the motor weights connecting the situation and the consequence equal to the action that caused the transition). Hence for moving from situation to action, the intermap connectivity was incorporated in the global knowledge space (GKS) of the Abstract reasoning System. These representations of situation-action –consequence, since they are known through trial and error do not have any value when the robot represents them. The value is known only when these observations can turn out useful while realizing a reward fetching goal. Hence there was a need to separately take into account the problem of storing isolated experiences in the ASMS.

As shown in a pictorial representation in figure 9, using these intermap connections the robot can retrieve the set of actions that are possible from any given situation. For example, from the state of having two small red sticks (we described how they are represented in the sensory datagram in the previous subsection) it can estimate that adhesion is a possible action, virtually executing the adhesion action then modulates the lateral connections and cause a shift in activity corresponding to a ASMS representation equivalent to the situation of having a long red stick (i.e the consequence of coupling two magnetic sticks). Now this object that the robot encountered in the mental space enters into the place map as a new entry. Once an object becomes available in the place map it can be used as a possible tool in the context of some other active goal. For example, the forward/inverse models for reaching now have a new imaginary object available in the place map to try using as a tool to virtually reach otherwise unreachable goals. We also note that there are no limitations on the number of actions an object or a situation can associate with (this depends on the sensorimotor experience the robot has had with the object), and during retrieval the complete set of possible actions can be retrieved.

Conceptual lateral connectivity.

This was the second new connectivity structure we introduced in the ASMS and has the function of forming 'situation-situation' or 'object-object' associations. While the introduction of the conceptual lateral connections is not needed explicitly for the solution any of the artificially reconstructed scenarios from animal reasoning (note that, intermap connections on the other hand is needed to retrieve a past experience of making a tool like the case of Betty), presence of the conceptual lateral connectivity offers the possibility to try out some more dynamic environmental scenarios of reasoning on the robot. Humans are very good at connecting different concepts, in fact moving effortlessly from concept to an another one that is related to the previous one in some way, often leant through one shot learning. For example we can easily move from the 'A' –'Apple'- 'Apple Pie' –'Keeping a doctor away'-may be even 'Gravity' for that instance. Conceptual lateral connectivity has the

function of introducing this kind of fluidity in the ASMS in a very primitive way, thereby allowing activity to move from one situation to another if they have some relation (the connection developed through experience). The crucial difference between the conceptual lateral connectivity (CLC) and the other sets of connections is the fact that, the CLC allow activity spread in the ASMS without the intervention of any motor action. Since introducing this new connectivity was done at a preliminary level we assumed that whenever the robot observes something novel can be made out of a situation representing a set of environmental objects, this situation is connected with the situations representing the individual objects (that were the members of the composite situation) with a conceptual lateral connection. A typical example is the case of playing with two red sticks to make a long red stick. In this case we develop a CLC between the situation representing the presence of two red sticks to the state where there is only one small red stick in the environment (that was a member of the previous composite situation). An example scenario is shown in figure 10.

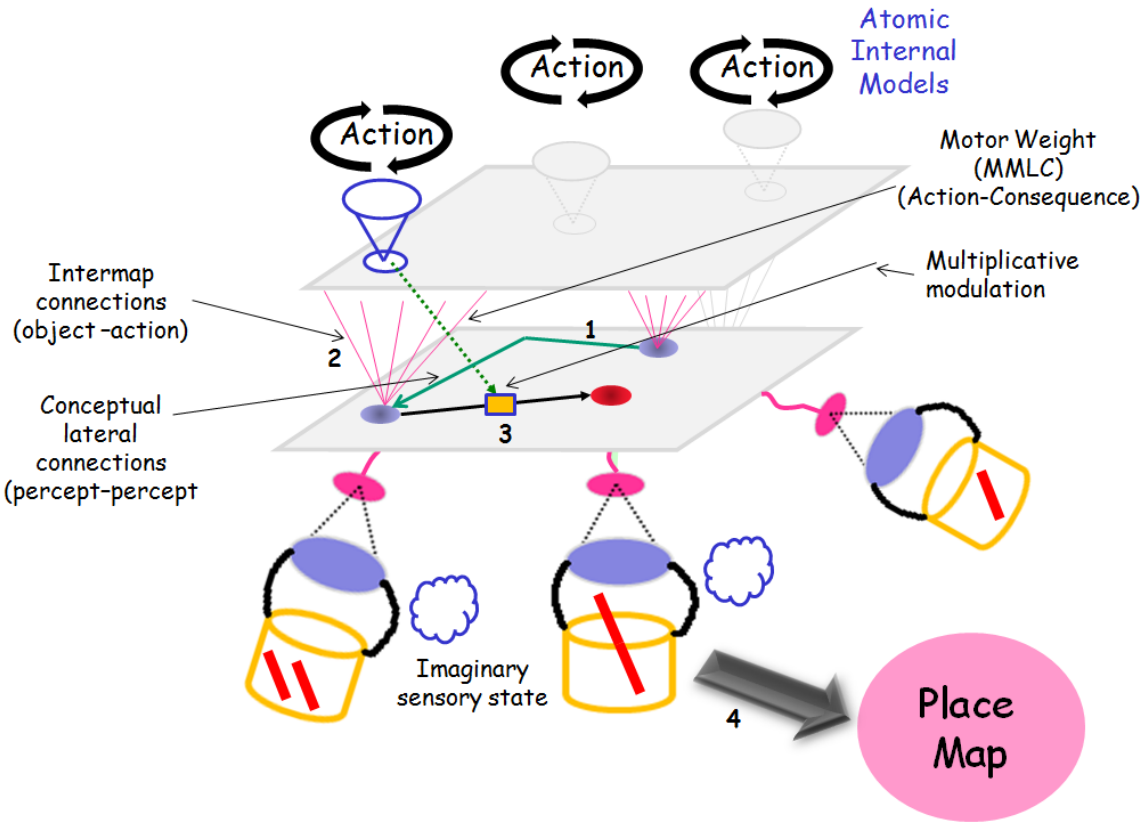


Figure 10. An example illustrating the use of three types of the connectivity structures in the abstract sensorimotor space. Let us consider that the robot wants to evaluate the possibilities that a small red stick affords. It can evaluate this using the intermap connectivity that associate objects with actions (or situation action) based on previous experiences. Let us further assume that nothing really useful is possible with the small red stick. Using the conceptual lateral connectivity that allow activity shifts between related percepts in the ASMS (without intervention/virtual execution of a motor action), it can move to a composite situation that

represents a sensory situation corresponding to having two red sticks. Now using intermap connectivity from this situation, it can see the possibility of using the Adhesion action from this situation. Virtually executing the adhesion action, modulates lateral connections through MMLC (that associate action with its sensory consequence) and cause one more shift in activity in the ASMS now leading to an imaginary situation of having a long red stick. This object can now be available for use, for further simulation about its utility in the context of an active goal (like the forward inverse model for reaching can simulate reaching otherwise unreachable objects using this new imaginary object as a tool in the gripper). Of course if this imaginary tool is of use, the robot must physically create it before is uses it, but a lot of information as to what components are needed, what actions are to be executed are all already available, only now the planning and execution must be done at a detailed level in the physical world, which the internal models for action generation do very efficiently.

The advantage of using the CLC is that, while trying to estimate about what is possible with a particular object (using inter map connections), the robot can also reflect on what is possible with situations that are related to this object (using intermap connections associated with those situations). In other words, it is possible to move from component to composite in the ASMS using CLC, and then move from the composite to the action space using intermap connections, in order to determine what actions are possible with the composite, what are their consequences (using MMLC).

G. Bidirectional interactions between ASMS and Motor space

In order to incorporate the effects of the new connectivity structures in the dynamics of ASMS, we need to minutely modify the expressions for the dynamics in comparison with the ones we used for the internal models in the previous chapter. However, the general information flow while moving from an initial state to the goal is the same i.e. Goal to computation of the Value field on the ASMS, value field and intermap connections are used to select the next most valuable action in the context of the active goal, the action when virtually executed modulates the lateral connectivity and shifts the winning unit in the ASMS. This bidirectional movement of activity between the ASMS and the action space continues till the time the monitor process detects the successful realization of the goal or in the pathological case where valuable actions cannot be determined (because intermap and reward structure does not exist) in which case behavior is switched to exploration.

Value computation

$$\dot{v}_i = -v_i + Q(x_{ASMS}, i) + \gamma_{goal}(W_{ij} v_j)_{\max} \qquad (1)$$

where v_i is the instantaneous value of the i^{th} unit in the ASMS at time t+1, Q is the instantaneous reward that could be obtained for moving to the i^{th} unit in the ASMS from the currently active state x_{ASMS}. These instantaneous rewards that can be expected for making a state transition in the ASMS are learnt using standard Q learning (since states in the ASMS are more discrete and small in number, standard reinforcement learning methods are quite

efficient in the ASMS). For different goal types (either reach goals or grasp goals right now, but can be extended for any other goal also) there are different Q matrices, and those are loaded appropriately based on the currently initialized goal. We will deal with the learning of reward values, and assigning rewards to intermediate states in the ASMS in the next section. The third term is the expected future reward, that could be obtained in the future, if a transition to the state 'i' was made at time t+1, weighted suitably by a discount factor that we set as 0.6 . In sum, the instantaneous value field over the states in the ASMS is a function of the instantaneous rewards that comes from the Q matrix, the current activations in the ASMS and the discounted future reward.

Values to Actions

The action to be executed at time t+1 is a function of both the value field in the ASMS and the intermap connectivity (between situation-actions) that associates a set of possible actions with a sensory situation. The motor action that needs to be virtually executed in order to cause a transition from the current active state x_{ASMS} to the most valuable state 'i' in the ASMS can be computed easily from the motor weights (exactly similar to the previous chapters). More precisely, the gradient $v_i - v_{xASMS}$ in the value field indicates how desirable motor activations M_{jxASMS} encoded in the motor weights, are when the current state is 'x_{ASMS}'. Hence the identifier representing the next motor action (internal model) that needs to be executed can be computed as follows:

$$A = \frac{1}{Z}\sum_{j=1}^{n} w_{ij} v_j M_{ij} \qquad (2)$$

where A is the six dimensional identifier for the action in the action space that needs to be executed. Recall that every primitive action is represented with an unique six dimensional identifier for all purposes of computation at the level of ASMS, and it is the identifier tags that are learnt by the motor weights M. So what equation 2 does is very simple to understand, once the value field stabilizes different units connected laterally with the presently active state in ASMS, say 'i' will have different values that indicate their relative importance in the presence situation. Equation 2 just averages the motor weights of all the connections that emerge from the currently active unit i, to all other units in the ASMS scaled based on their instantaneous value (where Z is chosen so as to normalize A=1). Now what we have is the 6D identifier of the action that needs to be executed in the action space in order to make the most valuable transition. For ASMS states that already have a value associated with them for a particular goal, the value field and the motor weights are sufficient to determine the most useful action at that point of time, for that goal. However as we mentioned earlier, there are also object action complexes in the ASMS that may not have any value associated with them, since they are merely learnt through explorative play. In such cases, the value field is null and hence it is not possible to move from ASMS to action space using equation 2. However, we can exploit the intermap connectivity in this case, to

find the complete set of actions that are possible or have been tried out by the robot in the past form that state in ASMS. The simplest way to estimate if an action j is possible from a state in ASMS say 'i' is to check its intermap connection C_{ij}. If C_{ij} =1 then action j belongs to the set of possible actions from the sensory state i. Of course if there are no intermap connections, it does not mean that a particular action cannot be executed, it can be still executed through random exploration provided the arguments it requires to run its execution are available in the place map. And in case it is executed successfully in a random exploration, then new intermap connections can be developed. It is also possible that there may be multiple actions in the set of possible actions that could be executed from a stat in ASMS .In case there are no values to arbitrate which action is the most valuable, one out of the possible action set is selected randomly. The set of actions in the action space possible from the current activity in ASMS can also be found out using a simple projection rule given in equation 3.

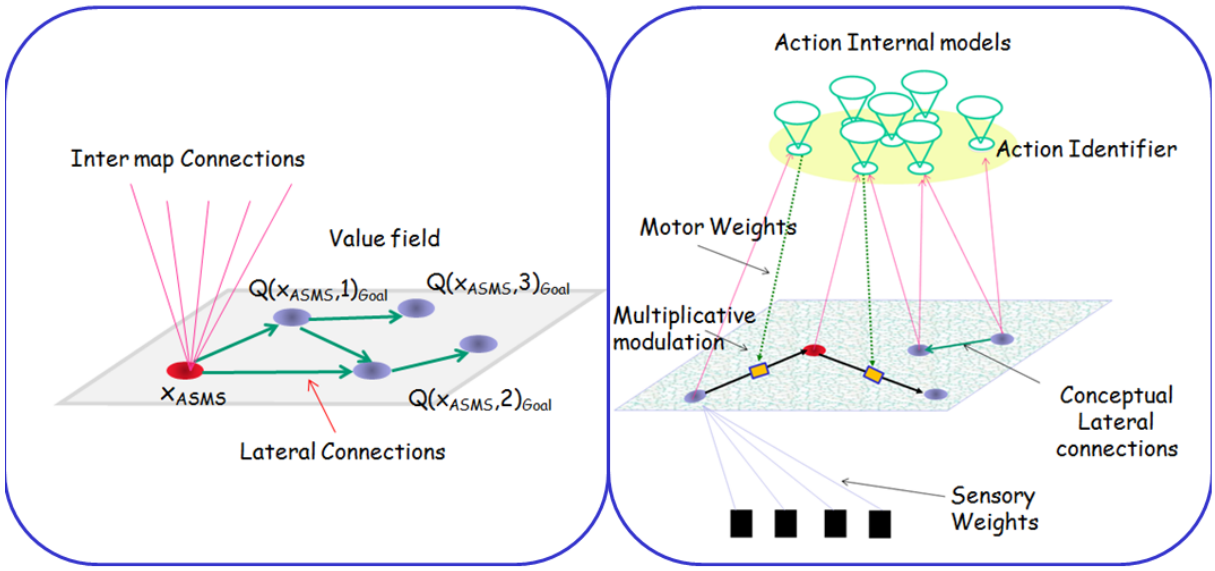


Figure11.Bidirectional interactions between the ASMS and the Action space. While the intermap connections (what are the possible actions), goal induced value field (what actions are valuable) and motor weights (what are the identifiers for the valuable actions) contribute in the Goal-Initial Condition-Action loop (Inverse model in a very general sense), the sensory weights (feed forward inputs that transmit reaction of the external world to the internal world), multiplicative motor modulation (prediction of the consequence of virtual execution of an action) and the conceptual lateral connections form the Situation-Action –consequence loop (Forward model in a general sense).

$$A_{Set} = \frac{\sum_i C_{ij} S_i(s)}{\sum_j \sum_i C_{ij} S_i(s)} \qquad (3)$$

where A_{set} is a vector of the dimension of the total number of actions in the action space (j), the value of n^{th} element in the A_{set} indicating the possibility of executing the n^{th} action in

the action space. C_{ij} is an elementary intermap connection between i^{th} unit in the ASMS to the j^{th} unit in the action space. $S_i(s)$ is the current activity of all the i units in the ASMS.

So to summarize, the reasoning process can move from the ASMS to the action space using the value field and motor weights, similar to the spatial SMS and pushing SMS described in the previous chapter. In case the value field is null, because the value is unknown at that time instance, it can still project the activity in the ASMS to the action space using the intermap connections to find the set of actions that were tried out and were possible from the current situation in the past, even if they had no value at that time. In the worst possible case where there are no intermap connections also, no potentially useful actions can be suggested by the reasoning system and either taking user guidance or performing a random exploration are the only options ahead for the robot. This pathological case (often possible if the state in ASMS was newly encountered in which case there are no values or lateral connections) is indicated by a message from the reasoning system to the user, in order to seek his approval to enter into explorative phase and randomly try any executable action or quit the goal.

We now come to the other loop in the abstract reasoning system i.e. the situation-action –consequence loop or moving from the virtual execution of an action to its sensory consequence in the ASMS. This loop is exactly similar to the previous computational models. The execution of an action in the action space, modulates the lateral connections in the ASMS and causes the next incremental shift in activity, corresponding to the sensory stimulus that would have occurred if the action was actually executed in the world. The dynamics of ASMS is exactly same as equation 4 of chapter 3, with just a new addition of taking into account the CLC introduced in this chapter.

$$\tau_x \dot{x}_i = -x_i + S_i + \beta_{if} \sum_{i,j} (M_{ij} W_{ij} + \alpha Co_{ij}) x_j \qquad (4)$$

Similar to the dynamics of the Pushing and Spatial SMS, the activity of any unit in the ASMS is a function both feed forward inputs (right now the sensory datagram) and the contribution from activity in the other units (in this case through both CLC and MMLC). M_{ij} in the case of ASMS are the 6D identifiers for the actions and not the actions themselves, which are internal models with their own dynamics and operating on their own sensorimotor spaces. This way of representation normally leads to abstraction in the system, with all the complexity of the dynamics at the level of an action hidden from the abstract reasoning system. We also note that the contribution of conceptual lateral connections are switched on only in the case of reflections on (tool use) i.e. what actions are possible on an object (situations related to an object) and are switched off during normal goal/value induced bidirectional shifts in activity in between the ASMS and the action space.

5.3 The Macrostructure of Reasoning

After describing the internal structure and function of different subsystems that form the microstructure of the reasoning system, mainly the sensorimotor variables on which the abstract reasoning system computes, the memory structures that help organizing the diverse information coming from a changing world (and the reasoning processes under execution) in a well defined and efficient frame work and the connectivity structures that mediate the bidirectional interactions between the sensory and motor variables to generate goal directed action sequences, in this section we will algorithmically analyze the global issue of how all the subsystems presented in the previous section interact when presented with a high level user goal.

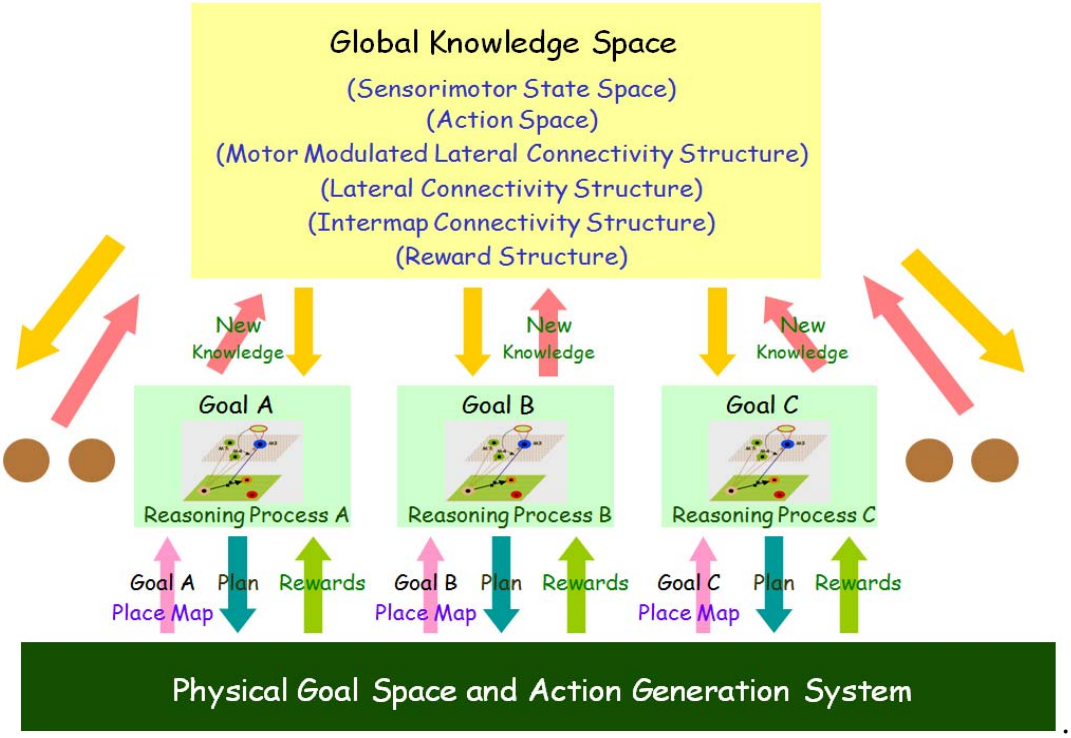


Figure 12. Interface between of the reasoning system, the global knowledge space of the robot and the external world. As seen in the figure, there can be multiple goals running in the system at any point of time, every goal running its own reasoning system. This is one more level of abstraction in the architecture, just like internal models for action presented in chapter 3 and 4 act as variables to the reasoning system, the reasoning system itself acts as a variable at the level of the robot (a cognitive agent with multiple competing goals) i.e every goal creates a new instantiation of the reasoning system (with all their complexity, knowledge abstracted away) .All the goals keep receiving refreshed updates of the world state through the place map, and rewards for their actions. All the goals also access the global knowledge space (GKS, that we discussed in the last section). Any new knowledge learnt during the process of realizing any goal, goes on to update the GKS. The output of the reasoning system is a plan (once again a (more abstract) trajectory in sensory/motor space, like the previous computational models, and is sent for execution in the physical space to realize the goal).

As seen in figure 12, the basic external input interface to the reasoning system comes from the user goal, the place map which organizes the information from the world in a structure that the reasoning processes and its associated actions identify with, and the rewards given by the user after realizing the goal (which is used for learning the values of internal states in the ASMS). The internal input interface to the reasoning system is the global knowledge space, which is the collection of sensorimotor variables (ASMS and the Action space) and connectivity structures that influence the interactions between sensorimotor variables in the reasoning system. The output of the reasoning process is a plan i.e. a temporally chunked sequence of actions directed on different useful objects in the environment using the internal action models, so as to eventually realize the active goal and fetch the reward. We will not enter into the issue of abstraction and having multiple goals for the time being. It will be a subject of discussion in the next chapter.

Learning Extended:

Prior to outlining the basic flow of information in the reasoning system when an user goal is issued, we will further extend the discussion about learning the connectivity and reward structures in the ARS. Even though the learning of sensorimotor variables and connectivity structures through motor exploration has been extensively dealt with in the previous chapter, we will briefly discuss issues that are specific to the abstract reasoning system. Similar to the pushing SMS and spatial SMS, the variables that need to be learnt in the ASMS are

a) Sensory (Self organizing and growing the ASMS and learning the Sensory weights)

b) Sensory Motor (Intermap, lateral connections and the motor weights)

c) Reinforcement (Reward structure)

d) Accidental learning of object action consequence complexes (like use of adhesion action on a couple of red sticks and their resulting consequence)

The abstract reasoning system when implemented on any robotic embodiment can be initialized either with Zero knowledge or with previously existing (learnt in the past) knowledge in the ASMS. Before learning the ASMS, it is however necessary for the robot to learn the different internal models presented in the previous sections so that it can use them for motor exploration while learning the ASMS. While learning the ASMS, the robot must learn both what actions it can do with different objects and what the consequences are (by trial and error) and what actions are useful to reach a goal from a particular situation in ASMS. Hence we may split the types of learning experiences into two kinds:

1) Accidental Trial and Error: Randomly try different action primitives on different available objects. Grasping a stick, Coupling two sticks, observing the consequence, Reaching other objects with sticks, cylinders, Pushing objects with sticks (we dealt with this issue in the last chapter, and also in the presence of traps, so pushing in this

chapter is just a variable P with all its complexity abstracted away). We must note that a large amount of knowledge can be gained by this phase of trial and error learning. In fact the additional affordance of the small red sticks is learnt and represented during this phase.

Learning in the presence of an User Goal: Considering that no knowledge (states, connections, rewards) exist in the system initially, the system must be bootstrapped initially with simple user goals (Reach/Grasp in simple environments) and once some initial knowledge exists, slowly the environment can be made more and more complex. Note that the reasoning process to solve any goal (Ex Reach Ball) can become more and more complicated based on the environment in which it has to be solved. As complexity increases, the agent encounters new states and discovers/codes new plans to solve them using the primitive actions available. These new states, their consequences, values are subsequently represented in the GKS. An example of how the complexity of the environment can be gradually increased for a goal Reach a ball is shown below.

a) Simple: Reach a ball in an environment where the object is in front of it and reachable.

b) Reach a ball when the ball is not immediately visible in which case use of visual search needs to be inserted in between as a valuable action

c) Reach a ball when the ball is not there in the environment at all, hence leading to low motivation and goal termination

d) Reach a ball that is reachable only with a tool, two tools or making a new tool (in which case the robot learns the value of reflecting on what is possible with different objects in the place map using the forward inverse models). Of course assuming that the robot has learnt the use of small red sticks earlier by trying the adhesion action on them, in future whenever it investigates what it can do with the small red sticks (using the intermap /conceptual lateral connections), it will remember the possibility of making a long red stick (a sensory consequence of this action). Now it is the task of the forward inverse models to mentally simulate reaching an unreachable object using this imaginary tool.

Considering the fact that the state space grows with time, the reward structure and the Q matrix must also grow in order to accommodate the new states and their values. So in the ARS, they are 3D in structure as shown in figure 13 (Top Panel). The first two dimensions code the state transitions (It's the actions that cause state transitions, subsequently coded by motor modulated lateral connections for use in future). The third dimension incorporates

the Goal dimension (different goals have different termination conditions, reward structure, strategies and the fact that same action can have different Q values in different goal contexts). For example if the goal was to reach, after the reaching is completed successfully, the interrupt action has the maximum value. But if the goal was to Grasp, after reaching its the grasping action that has greater value and not the interrupt action (even though both actions are possible from the same situation for both goals, in case the robot uses the reaching Q matrix for a grasp goal, it will interrupt the execution after reaching and fail to realize the goal that was issued to it and hence end up being penalized). Hence reach goals and grasp goals have different reward and Q matrices that are loaded appropriately when the goal is initialized.

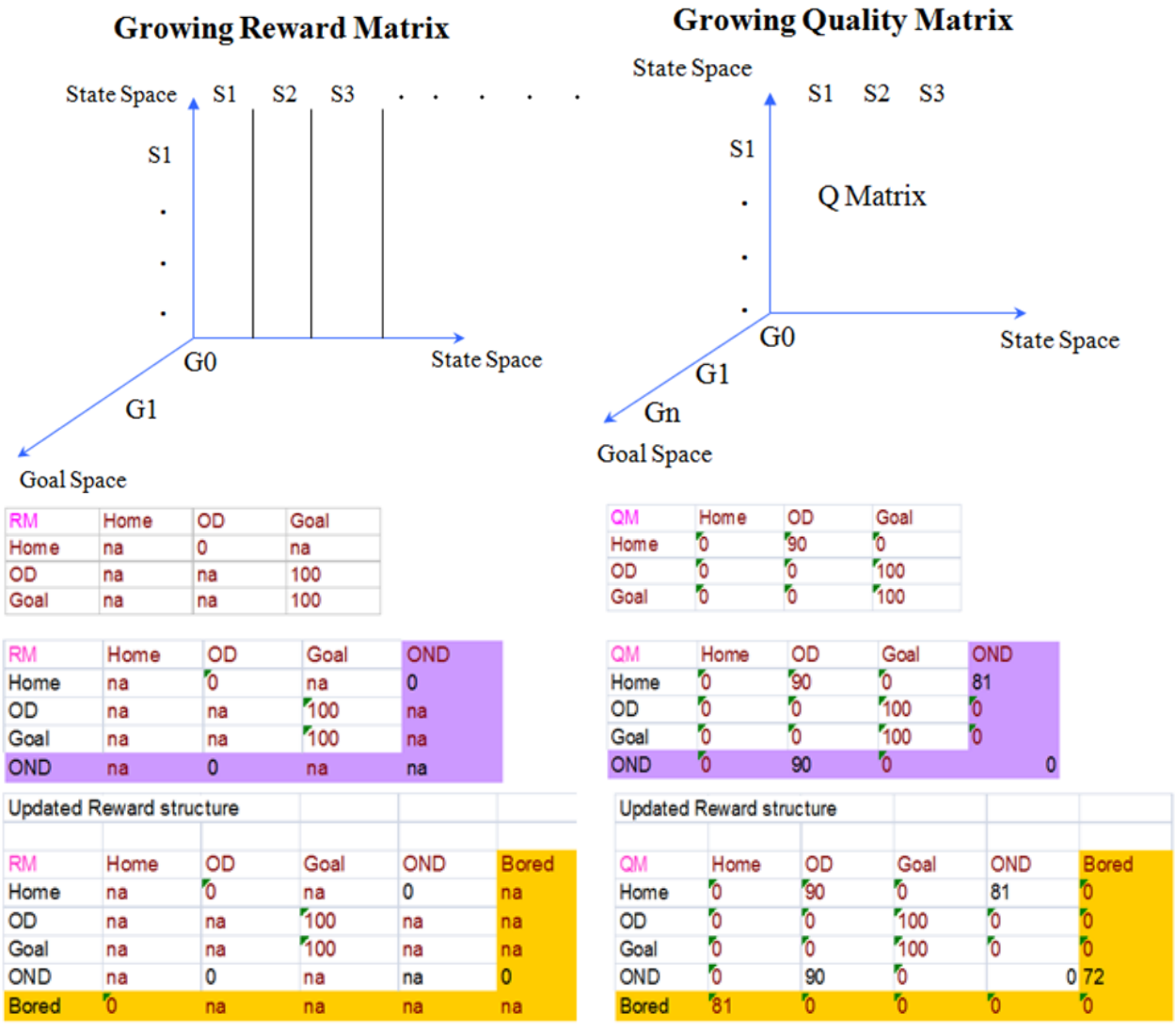


Figure 13.

The top panel of figure 13 shows the three dimensional and growing structure of the reward and quality matrices used to compute the values of different states in the ASMS. The first

two dimensions code the state transitions (It's the actions that cause state transitions, subsequently coded by motor modulated lateral connections for use in future). The third dimension incorporates the Goal dimension (different goals have different termination conditions, reward structure, strategies and the fact that same action can have different Q values in different goal contexts). For example if the goal was to REACH, after the reaching is completed successfully, the interrupt action has the maximum value. But if the goal was to GRASP, after reaching is completed, it's the grasping action that has greater value and not the interrupt action (even though both actions are possible from the same situation for both goals. In case the robot uses the 'REACH goal' Q matrix for a 'GRASP' goal, it will interrupt the execution after reaching and fail to realize the high level goal that was issued to it and hence end up being penalized). The bottom panel in figure 13 shows the growth and learning of the reward and Q matrices during the progressive training of the robot for solving the high level goal of reaching a target object under increasing levels of environmental complexities. In the first case, the goal is most straight forward to solve: i.e. the object is immediately visible and also immediately reachable. The ASMS has no states present inside it other than the home state (all elements in the sensory datagram are 1 in the home state, and is the only state defined by the user) with which the reasoning system is initialized when any goal is issued to it. As new states are encountered through motor exploration, the ASMS grows, new intermap, lateral and motor weights are also developed in similar fashion as in the previous chapter. Randomly initiating a visual search from the home state, leads to a new state corresponding to the successful detection of the object through vision and consequent updation of the place map and goal space. This now makes reaching action possible and when the forward inverse model is triggered (by random selection) the goal is reached successfully. Once the goal is reached after randomly trying out primitive actions in the proper sequence, the robot is also rewarded. This reward is distributed using Q learning (a standard reinforcement learning technique developed by Watkins, 1989):

$$Q(St, Action)_{Goal} = R(St, Action)_{Goal} + \gamma \, Max[\, Q(\text{next state, all actions})_{Goal}] \quad (5)$$

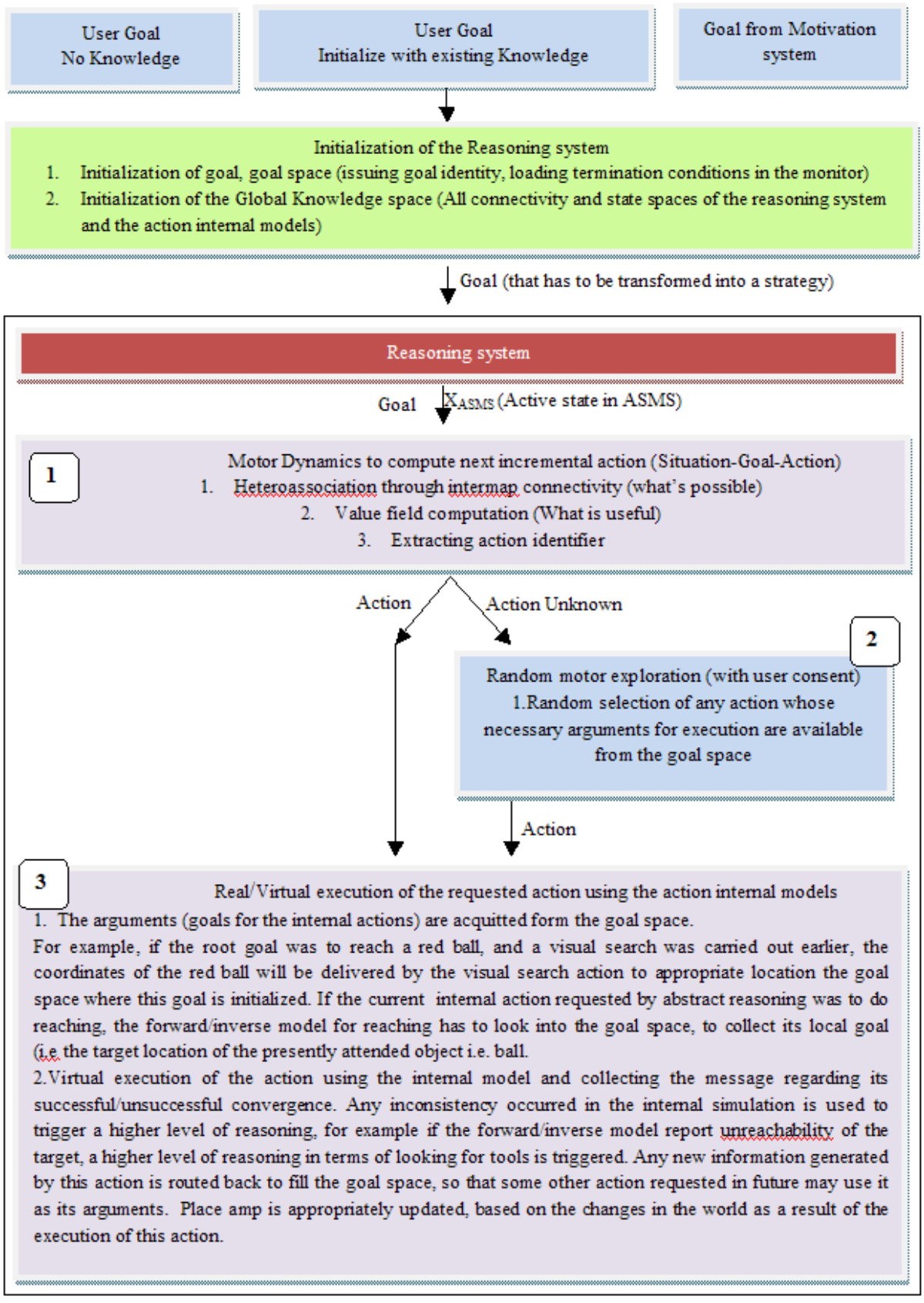
where γ is the discount factor (0.9 in our case). Once the goal is reached in this very simple scenario by randomly trying out possible motor actions and representing the new sensory states, new Intermap connections are developed and Q matrix is updated using 5. As we can see a state transition from the home state to the intermediate state representing successful detection of the goal object now has a Q value of 90. In the next training episodes the environment is made little bit more complex in the sense that the agent must do a spatial exploration to search for the object in the second case, and the goal object is

not present in the environment at all in the third case (leading to a loss in motivation and termination). In the second case we must note that the system can initially use the previously developed connectivity and Q matrices to trigger the visual search from home state, which now returns a failure or object not detected through vision (ond), since this was a new state encountered and there are no known actions for the system from here and the robot switches from reasoning to exploration. If after trying out spatial exploration the robot detects the object through vision, it now comes back to a familiar state in state space (od) representing successful detection of the goal object through vision. Now the robot need not do a random action selection from here because it already knows what action to execute from the state od in order to realize the goal. Once the goal is reached by triggering the reach action (that has maximum value from od , and which was learnt in the past episode), the robot is rewarded again and the Q matrix is updated again. We must note here that while attempting to realize the same goal in a slightly complicated environment, action selection through exploration was sandwiched between two iterations of action selection through value based computation. In other words all previously gained knowledge can be used by the robot in more complex environments of the same goal. Only it must switch back to exploration when it encounters a new sensory state. We can also see that the robot already has a nice chain of actions at its disposal now. For example, it knows that it going for a spatial exploration is the most promising action if the object is not detectable through vision. In this manner, as more and more complex environment is encountered, the robot uses both previous experience and explorative drive to learn the state space, connectivity structure and reward structure. When a new state is encountered, it initially has no lateral connections or reward values. The robot has to now engage in episodes of randomly trying the primitive actions and observing the consequences till the time it reaches the goal and is rewarded in which case the Q values for the intermediate actions are updated again. Just like the case of development of the growing neural gas for the spatial map, this training also being cumbersome was done in MATLAB and the developed Q and Reward matrices were incorporated into the global knowledge space of the real robotic platform.

We must note that while attempting to realize the same goal in a slightly complicated environment, action selection through exploration was sandwiched between two iterations of action selection through value based computation. In other words all previously gained knowledge can be used by the robot in more complex environments of the same goal. Only it must switch back to exploration when it encounters a new sensory state. We can also see that the robot already has a nice chain of actions at its disposal now. For example, it knows that it going for a spatial exploration is the most promising action if the object is not detectable through vision.

Figure 14 describes one cycle of reasoning from the time a new goal is issued to the system and a plan is outputted by the reasoning system to the execution system, that

unbinds the plan descriptor and sends the execution commands to the various actuators. As seen from figure 14, there are three ways to enter into the reasoning system: user goal with no initial knowledge in the system, user goal with the existing knowledge loaded and the case when there are no user goals, so the system randomly executes action on objects and observes the consequences. The net outcome of the initialization process is that a root goal enters into the goal space and the control is transferred to the reasoning. Hence the system enters into exploration through the reasoning, whenever the current knowledge existing in the system is not sufficient to compute an action to be initiated in the next time step. In this way the system when initiated from Zero knowledge will immediately into exploration (zero knowledge means with respect to the reasoning system only, knowledge of primitive actions learnt in the previous chapters are retained and are necessary to learn the ASMS and its connectivity).


User Goal
No Knowledge
User Goal
Initialize with existing Knowledge
Goal from Motivation system
Initialization of the Reasoning system
1. Initialization of goal, goal space (issuing goal identity, loading termination conditions in the monitor)
2. Initialization of the Global Knowledge space (All connectivity and state spaces of the reasoning system and the action internal models)
Goal (that has to be transformed into a strategy)
Reasoning system
Goal
X_{ASMS} (Active state in ASMS)
1
Motor Dynamics to compute next incremental action (Situation-Goal-Action)
1. Heteroassociation through intermap connectivity (what's possible)
2. Value field computation (What is useful)
3. Extracting action identifier
Action
Action Unknown
2
Random motor exploration (with user consent)
1.Random selection of any action whose necessary arguments for execution are available from the goal space
Action
3
Real/Virtual execution of the requested action using the action internal models
1. The arguments (goals for the internal actions) are acquitted form the goal space.
For example, if the root goal was to reach a red ball, and a visual search was carried out earlier, the coordinates of the red ball will be delivered by the visual search action to appropriate location the goal space where this goal is initialized. If the current internal action requested by abstract reasoning was to do reaching, the forward/inverse model for reaching has to look into the goal space, to collect its local goal (i.e the target location of the presently attended object i.e. ball.
2.Virtual execution of the action using the internal model and collecting the message regarding its successful/unsuccessful convergence. Any inconsistency occurred in the internal simulation is used to trigger a higher level of reasoning, for example if the forward/inverse model report unreachability of the target, a higher level of reasoning in terms of looking for tools is triggered. Any new information generated by this action is routed back to fill the goal space, so that some other action requested in future may use it as its arguments. Place amp is appropriately updated, based on the changes in the world as a result of the execution of this action.

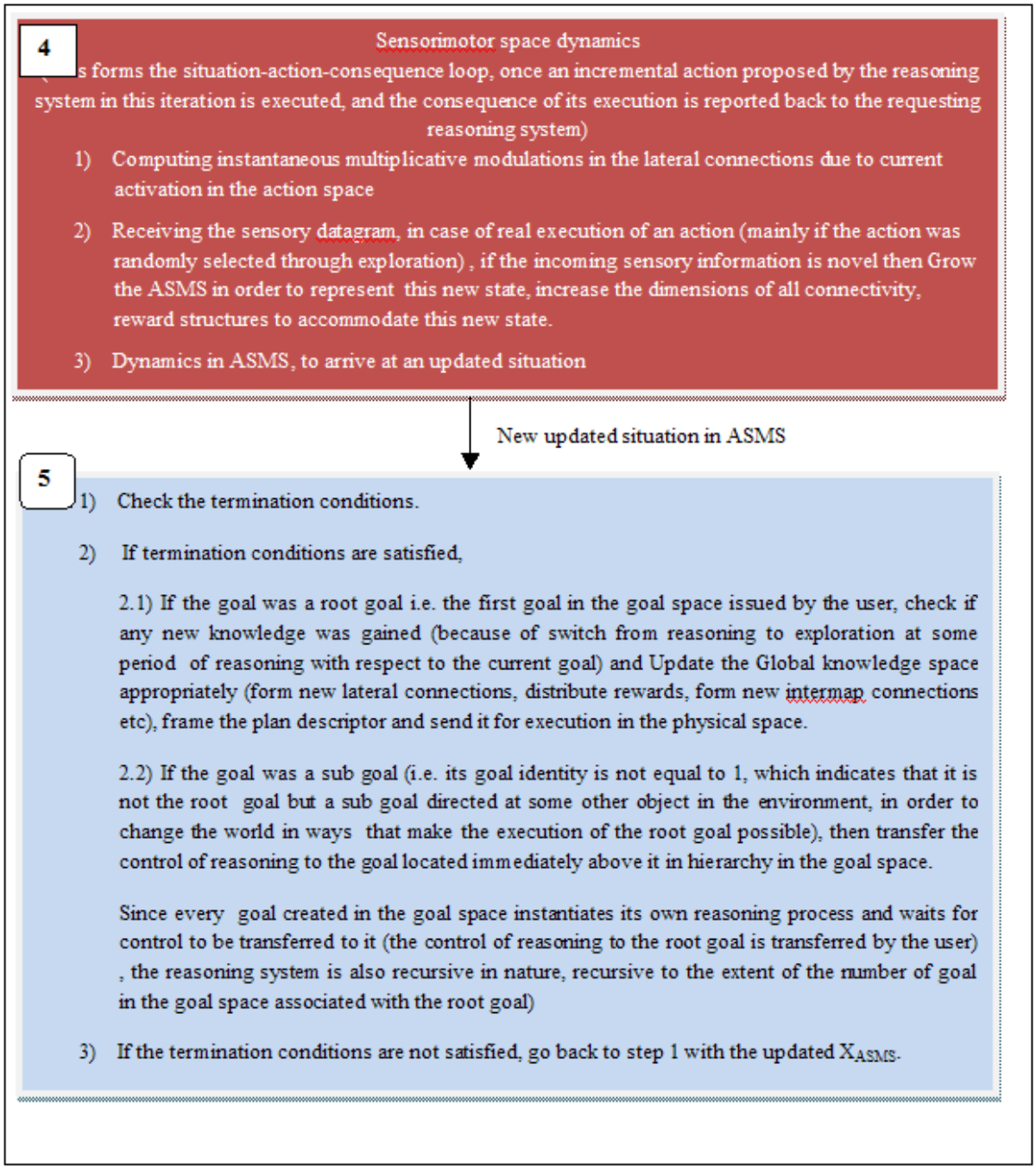


Figure 14.The flow of information through different computational models, memory and information management structures, knowledge/connectivity related structures and software message framing processes during one loop (or micro-cycle i.e. moving once from perception to action to perception) of reasoning, from the time a new goal is issued to the system. The complete solution to a high level goal in a complex environment may involve several such instantiations of reasoning processes acting on different potentially useful objects (tools) on the environment, each running several such microcycles, and executing actions so as to change the world in ways that make execution of the root goal possible.

Even when the system is initialized with the existing knowledge, it can enter into exploration through reasoning, in cases where the existing knowledge is not sufficient to find actions in a

complicated environment. In this way reasoning and random motor exploration are always weakly coupled and based on the complexity in the environment, in case the motor dynamics fails to converge to an action, the system moves into exploration and chooses a random primitive action to try (whose arguments are defined based on the existing information in the place map, or goal space).So the message that goes to the action space is the identifier of an action that needs to be executed, which can either come due to value based computation using the existing knowledge of the system, the goal and initial condition, or it can come as a result of a random selection.

The action space now either executes the action virtually using the internal models if it was requested as a result of value based computation or physically executes the action if it was triggered as a result of random selection during exploration (in which case the robot has to observe the real sensory information). When ever, visual search is requested, first the place map is searched and if the object does not exist in the place map, then a visual analysis is done in the physical space. All other actions (reaching, spatial planning, pushing, reflection on objects as possible tools, adhesion) can be virtually executed to the lowest level of detail, using the internal models learnt in the previous chapters. We also note here that all these internal models have their own complexity, goals, constraints and dynamics all of which is totally abstracted away at the level of reasoning. At the level of reasoning, what matters is the outcome of the virtual simulation of the internal models, new information generated as a result of the virtual execution of these actions, and the changes in the world that are caused (that is reflected in the place map).

After the real/virtual execution of an action using the internal model and reception of the message concerning its outcome, the next step is to estimate the consequence. This is done by the ASMS dynamics. We now have an updated situation in the ASMS, reflecting the consequence of the execution of the action on the world (through motor modulated lateral connections), if the action execution was mental using the internal model. If the action was physically executed on the world, then the incoming sensory information may also be new, corresponding to a situation that the robot has never encountered before. In this case the ASMS is grown to accommodate this new state, the dimensionality of all connectivity and rewards structures are increased appropriately. Now we are set with the fifth major task of determining if the sensory situation reached in the ASMS satisfies the rumination conditions defined for the goal. For that we must use the monitor process. The monitor process than does a series of actions like updating knowledge in case of successful termination, transferring control to some other goal in the goal space in case the goal that was terminated was not the root goal , but a sub goal. We note here that there are as many reasoning processes running in the robot, as many as the number of sub goals in the goal space linked to the root goal. In this sense reasoning system calls itself and is inherently recursive in nature. The resulting computational scenario analogous to a recursive block diagrams of figure 14 is pictorially illustrated in figure 15. The advantage of having recursivity

in the architecture is mainly a great amount of generalization, one need not specify all the details to all the goals, the reasoning system takes care of dealing with the details. As an example, let us consider the 2 sticks paradigm. It may be possible to insert n sticks of decreasing lengths in between the robot and the ball, and give it a goal to grasp the ball. Since the reasoning process is recursive in nature, the knowledge regarding trying to reach a ball with a long stick, is automatically generalized to trying to reach an unreachable long stick with another shorter stick good enough to reach the long stick and so on. Since the robot instantiates reasoning recursively for every goal (be it the user goal or a sub goal generated because an object in the environment was found useful during the execution of the primitive action of reflecting on environmental objects I the context of the active goal), and the knowledge regarding the strategy is represented independent of the goal object (ball, stick, cylinder etc), the arguments related to the goal always being available in the goal space, a great mount of generalization and flexibility towards handing changing environments is incorporated into the architecture.

Similar to the Pushing and Spatial SMS, the output of the reasoning SMS is also a trajectory, a trajectory in the sensorimotor space (more abstract in nature though) and a trajectory in the (atomic) action space. However a small point to note here is that unlike the previous internal models that work on well specified goals issued by the reasoning system, run their dynamics and output their virtual/real trajectories of sensorimotor variables needed to realize their local goal, the reasoning system is recursive in nature (calls itself) and hence different sub goals get priority to execute their actions such that some other sub-goal or goal can be realized successfully in the modified environment that resulted from their actions. Hence the difference in the case of reasoning system is that there are as many abstract sensorimotor trajectories, as the number of separate instantiations of the reasoning processes made in order to realize the sub goals that help realizing the root (user) goal.

Before we conclude this section, we will briefly describe how the output of the reasoning system i.e. the plan descriptor is structured. Similar to the Pushing and Spatial SMS, the output of the ASMS is also a trajectory, a trajectory in the sensorimotor space (more abstract in nature though) and a trajectory in the action space. However a small point to note here is that unlike the previous internal models that work on well specified goals issued by the reasoning system, run their dynamics and output their virtual/real trajectories of sensorimotor variables needed to realize their local goal, the reasoning system is recursive in nature and different sub goals get priority to execute their actions such that some other sub-goal or goal can be realized successfully in the modified environment that resulted from their actions.

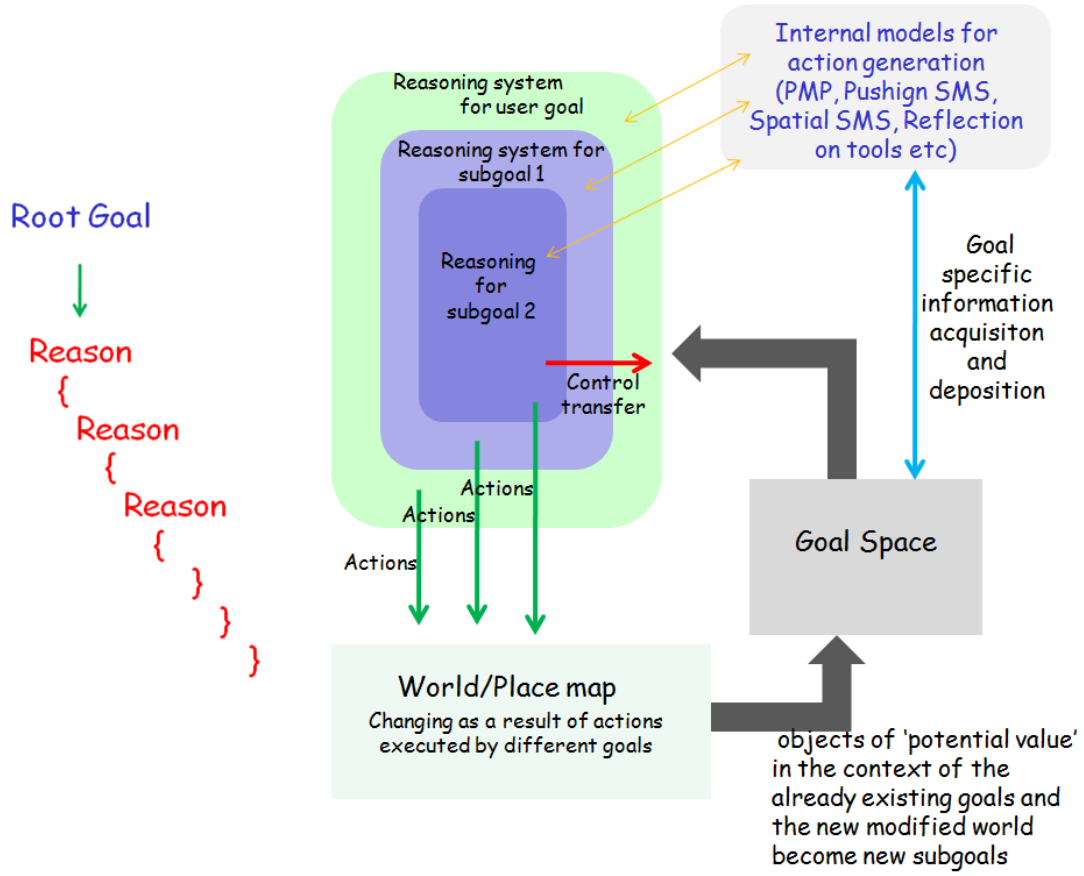


Figure 15. Recursivity in the reasoning system. In complex environments, the task of realizing an user goal may not be straight forward, and before attempting the user goal itself, the robot must act on other objects in order to transform the world in ways that lead to the successful realization of the root goal. Every time a potentially useful object (and the action on it that makes it useful in the context of the active goal) is found by the primitive action of reflecting on environmental objects (using the forward/inverse models, conceptual lateral connectivity etc), this intermediate object of high value (and the action that makes it useful) becomes the sub goal of the system. Since the reasoning system (with all the internal models inside it) itself is recursive in nature (and can call itself), every sub goal enters into the goal space exactly like the user goal (has its own identity, termination conditions) and instantiates its own reasoning system. Hence the task of realizing the sub goal successfully is now a process of reasoning itself, and nothing needs to be specified to the system in order to achieve it, based on the existing knowledge in the GKS and the complexity of the environment, this reasoning process will also evolve with time. For example, it a sub goal was to make a long red stick through adhesion, and only one red stick is visible, the robot need not be told to go for a spatial exploration (in other words all knowledge is represented and utilized in a goal dependent manner and not object dependent manner). Let us consider further that the other red stick was inside the trapping groove, we need not tell the reasoning system about searching for a new tool from the environment (resulting in a new reasoning system instantiation if anything useful was found), or we need not tell the robot about the direction of pushing (which is a local reasoning task for the pushing internal model under the scope of the goal/sub goal that triggers it). The number of reasoning threads, sub goals that exist, the changes occurring in the world as a result of the interventions made by the actions initiated by the different executing reasoning processes cannot be predicted in advance and are a function of the complexity of the environment itself. In case actions needed to achieve the goal could not be found (as no knowledge exists, new states are encountered), useful tools are not found, or if the internal models for action anticipate that the environment itself does not allow some particular action to be executed in ways that may be rewarding (like is the ball is paced in between two traps in the trapping groove, even if the robot has a long enough stick and has

the capability to push, the environment is such that the ball cannot be retrieved) the robot either goes into exploration mode (with user consent) or quits the goal due to low motivation (and is ready to handle some other rewarding user goal).

Hence the difference in the case of reasoning system is that there are as many abstract sensorimotor trajectories, as the number of separate instantiations of the reasoning processes made in order to realize the sub goals that help realizing the root goal. More over some actions are virtually executed and some of them have to be sent to the execution system for real execution (temporally sequenced in a proper way) . In order to manage the change at the level of plans that is dependent on the changes at the level of reasoning which is dependent on the environment, every action model that is triggered by any reasoning process has to deposit its outcomes to the plan descriptor (if it is a real execution) or to the GUI if it was a virtual experiment. A pictorial representation of the plan descriptor structure is shown in figure 16, for an example of using a blue stick as a tool to reach a green ball.

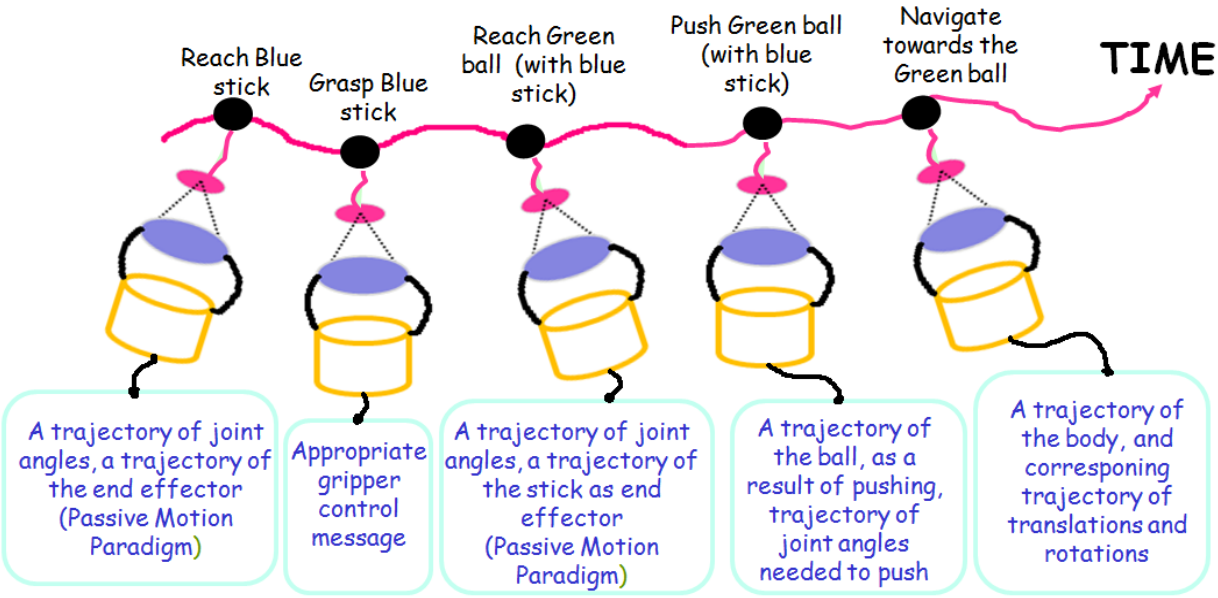


Figure 16. The dynamic plan descriptor structure. The plan descriptor is constructed dynamically along with the evolution of the different reasoning processes. One nay visualize the plan descriptor as a fine thread in time, on which many empty baskets are attached inside which every action that has to be physically executed identifies itself and empties its results (trajectories of sensorimotor variables corresponding to the local goal for which it was triggered). The plan descriptor in itself is self contained in the sense that it holds all the necessary information starting from an abstract chain of actions sequenced in time, to the detailed level of motor commands that needs to be sent to different actuators to perform the action.

5.4 The Fabric of reasons and actions

In chapter 2, we presented a general overview of the environmental set up we constructed for validating the reasoning-action generation system of the GNOSYS robot. Several scenarios that could be validated on the robot were analogous to artificial reconstruction of

experiments related to physical cognition conducted on a large circle of animals (birds, chimpanzees and children).The next two chapters were dedicated to the development of the internal models that drive the real/mental action generation capabilities of the robot. In chapter 2 we focused on the problems of run time generation of composite forward inverse models for simulation/execution of a variety of reaching movements with /without tools and taking into account subtle constraints imposed by different tasks may have to be executed in different environments. Several simulation results and performance of the forward/inverse models on the GNOSYS robot and iCub were presented. In chapter 3 we aimed to mentally simulate one sequence of pushing-moving-reaching using the different internal models. In order to achieve this, the spatial map of the playground and a model for pushing was learnt through random motor exploration, and subsequently adapted to take into account a range of constraints in their respective sensorimotor spaces (energetic, dynamic obstacles, traps etc). Taking the set of internal models for action developed previously as variables in the motor space, in this chapter we developed the abstract sensorimotor space, dynamics of which generates highest level trajectory of the system i.e. the plan descriptor. In this section, we will analyse the 'Reasoning-Action generation system' driven behaviour of the robot when it is issued user goals in different environmental scenarios we outlined in chapter 2. Even though the kind of environmental scenarios that can be created is only limited by the imagination of the experimenter himself, we will analyse four different cases that will reflect different aspects of performance of the reasoning system on the robot.

5.4.1 Direct tool use: 'Grasping a Green ball'

To begin with, we consider a scenario in which the robot is issued a user goal to grasp a Green ball. As shown in Panel 1 of figure 17, in the initial environment the ball is placed in the center of the trapping grove, unreachable form any direction. In addition one trap is placed in the trapping groove as an additional constraint. After the initialization process of issuing an identity to the root goal in the empty goal space, and initializing the connectivity structures in the abstract reasoning system and the different internal models, the robot initially searches for the green ball and refreshes the spatial information related to its location in camera coordinates, that is reconstructed into Euclidian space using the 3d reconstruction system, this information also enters the goal space, inside the scope of the root goal. This implies that the temporary utility value of this object in the place map is modified to 0. This implies that any other user goal cannot utilize this object form the environment unless this goal frees the object once again by remodifying the place map. In this sense the information in the place map (that basically reflects the state of the world) in the short term memory of the robot is global and available to all the high level goals existing inside the Gbrain. Whenever any goal takes control of any environmental object (both in the mental space and physical space), it makes that object unavailable for other concurrently running goals.

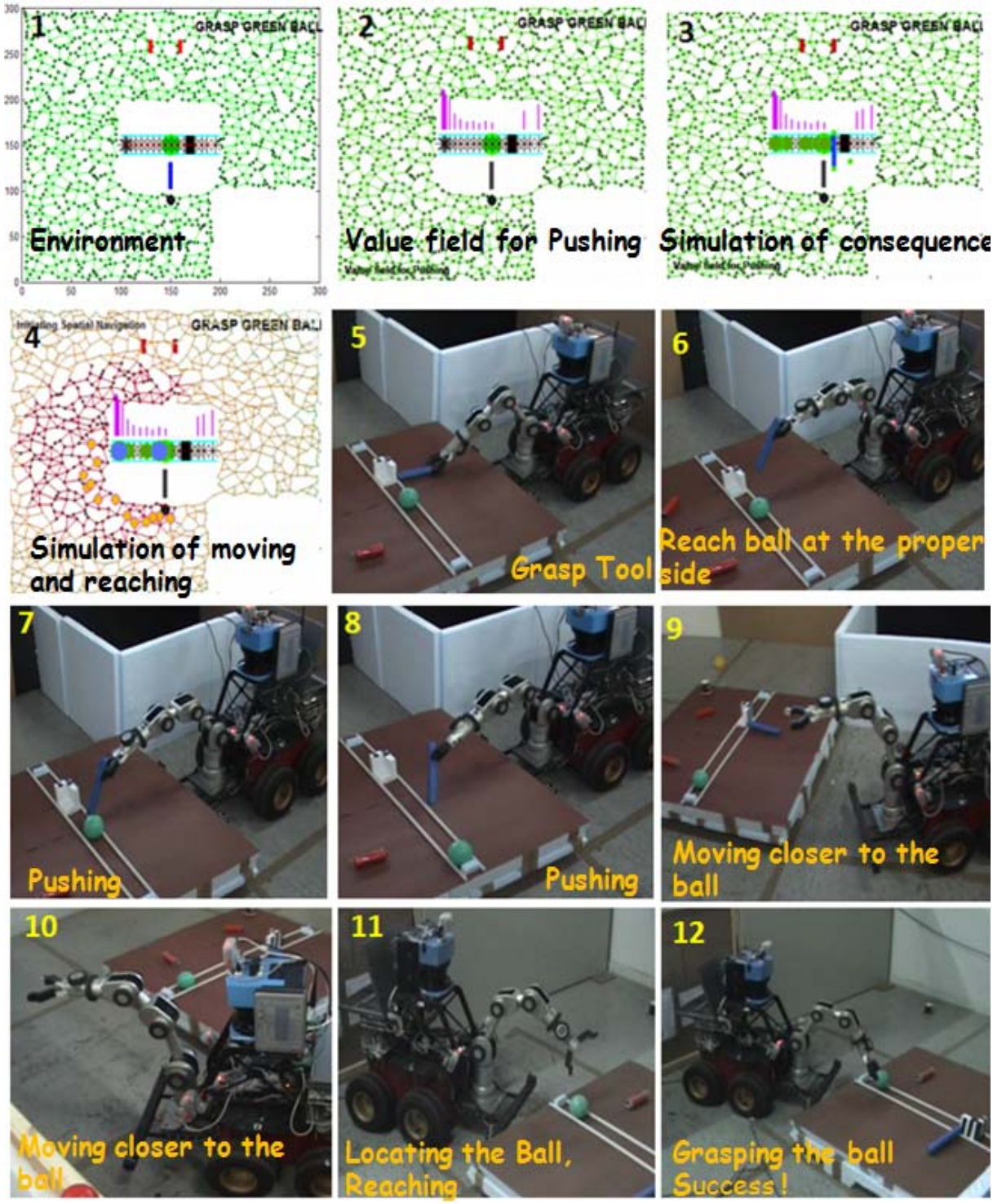


Figure 17. The goal issued to the robot is to grasp the Green ball in the given environmental scenario. Panels 1-4 show the mental simulations initiated by different actions in order to reason about the possibility of using the blue stick as a tool to realize the goal. Panels 5-12 show the sequence of real actions initiated by the robot to make the situation in the external world similar to the simulated situation in its internal world, hence changing the environment in ways that make the user goal now achievable.

Now the action of virtually reaching the ball using the forward inverse model is initiated, that collects the information it needs for its execution from the goal space and triggers the passive motion paradigm. Since the forward/inverse models predict that the ball is unreachable and report the consequence of their simulation to the action space of the abstract reasoning system. Now the action and its consequence modulate the lateral connections in the ASMS leading to an updated situation in the ASMS, from where the motor dynamics to compute the next incremental action is initiated. Now the robot begins to reflect on the other objects available in the place map to evaluate their affordances in the context of the present unsuccessful goal. Finding the long blue stick, it runs a FMC/IMC simulation of trying to reach the green ball using the blue stick as an extension to the end effector, which results in a success. Detecting some utility value in the stick, a new sub goal to grasp the blue stick (that in all computation terms is exactly similar to the root goal itself) is created and initialized in the goal space, which instantiates a new thread of the reasoning system. The reason for running a new reasoning process for the blue stick itself is motivated in order to deal with the changing world. In other words, the world is not constant. Even though the blue stick is useful, in some environment it may be inside the trapping groove, it may be located somewhere in space (in which case the robot must go searching for it) and so on. Hence before executing any physical action in space, the robot mentally rehearses the complete chain of actions on the world that may be needed to realize the user goal (if the environment permits). Now the reasoning process for the blue stick progresses similar to the earlier case editing the TUV of this object in the place map and triggering the PMP which reports successful simulation of reaching, and subsequent grasping (all in the mental space). Now the robot knows for sure that the blue stick has some utility value in terms of reaching the green ball, and the fact that the blue stick itself is reachable immediately (there can be situations where there can be multiple tools equally useful, but acquiring them may be expensive in terms of energy, that we do not consider here). Now the robot progresses with is evaluation of the use of the blue stick in terms of realizing the user goal. The Pushing system generates a value field as shown in panel 2, taking into account the trap (as we saw in the previous chapter). Simulation of the pushing action now gives the robot an estimate of the relative displacement of the ball as a consequence of pushing (Panel 3). Once the pushing dynamics stabilizes, the location of the Green ball in the goal space is modified to the new simulated location, and not the previously existing location in the center of the table. Movement of the ball to a spatially accessible location in the world now induces spatial rewards in the spatial map and the spatial dynamics generate the possible trajectory of the body in order to spatially move closer to the simulated location of the ball (Panel 4). Once the dynamics of spatial growing neural gas becomes stationary, Gnosys has the two crucial pieces of information needed to trigger forward/inverse model for the arm to reach the Green ball once again, however in a world that has changed quite a lot in its mental space (Goal Space) due to its previous virtual actions. They are the predicted location of the

Green ball (coming Pushing model), and the initial conditions (location of the body/end effector predicted by the equilibrium configuration of the dynamics in the internal spatial map). Now trying to virtually reach the ball is successful and hence the utility of the blue stick is confirmed. We note that till now there were no physical actions executed by the robot. In case the robot finds out in its simulated world that it is not possible to realize the goal in the current environment, it will either quit the goal or move to some other user goal (whose priority is expected to increase) or move to exploration seeking the consent of the user.

Since in this case the virtual experiments predict that the user goal can be realized if the world is appropriately modified, the robot now begins to initiate real actions in the world shown in panels 5-12. Of course executing each of the motor actions (reaching, wrist orientation, reaching with stick, pushing, moving, note that the last reaching of the ball from the other end of the table is quite complicated for the F-I model to plan in terms of the joint limits) seen in the panels have their own complexity that was dealt with in the previous chapters. On realizing the goal, the blue stick is freed for use once again in the place map, the goal space is cleared and the robot is ready to handle the next user goal, now in a modified world.

5.4.2 Betty's tool making task revisited: 'Grasping a Red ball'

We now move to a slightly more complicated version of the artificial reconstruction of Betty's tool making task. The environmental scenario is shown in panel 1. The goal of issued to the robot is to grasp the red ball, once again placed in the center of the table like the earlier case. However, the twist in the environment is that now, there are no long enough tools directly available for the robot to use, like in the previous case. Moreover it is a slightly more complicated version compared to Betty's task in the sense that Betty had the raw material needed to make the tool available in front of it (i.e. the straight wire), but in this case only a part of the raw material is available in front of the robot (which is not sufficient to make the tool itself). Similar to the earlier case, the robot uses the forward/inverse model to try reaching the red ball directly (Panel 2, the color of any object under consideration by the reasoning system is shown shaded in the GUI), making appropriate amendments to the place map and the goal space. Inconsistency in the simulation of the PMP now triggers a higher level of reasoning in terms of the evaluation of possibilities afforded by the immediately available environment. Making a virtual experiment to reach the ball using the small red stick ends in failure and there are no other tools immediate located in the environment. Spreading the activity in ASMS using the conceptual lateral connectivity, the system now attempts to recollect past experiences with the available object with the hope of finding a way out. Since the conceptual lateral connectivity allow activity shifts between related percepts in the ASMS (without intervention/virtual execution of a motor action), the

system now moves to a composite situation in the ASMS that represents a sensory situation corresponding to having two red sticks (as illustrated in figure 10).

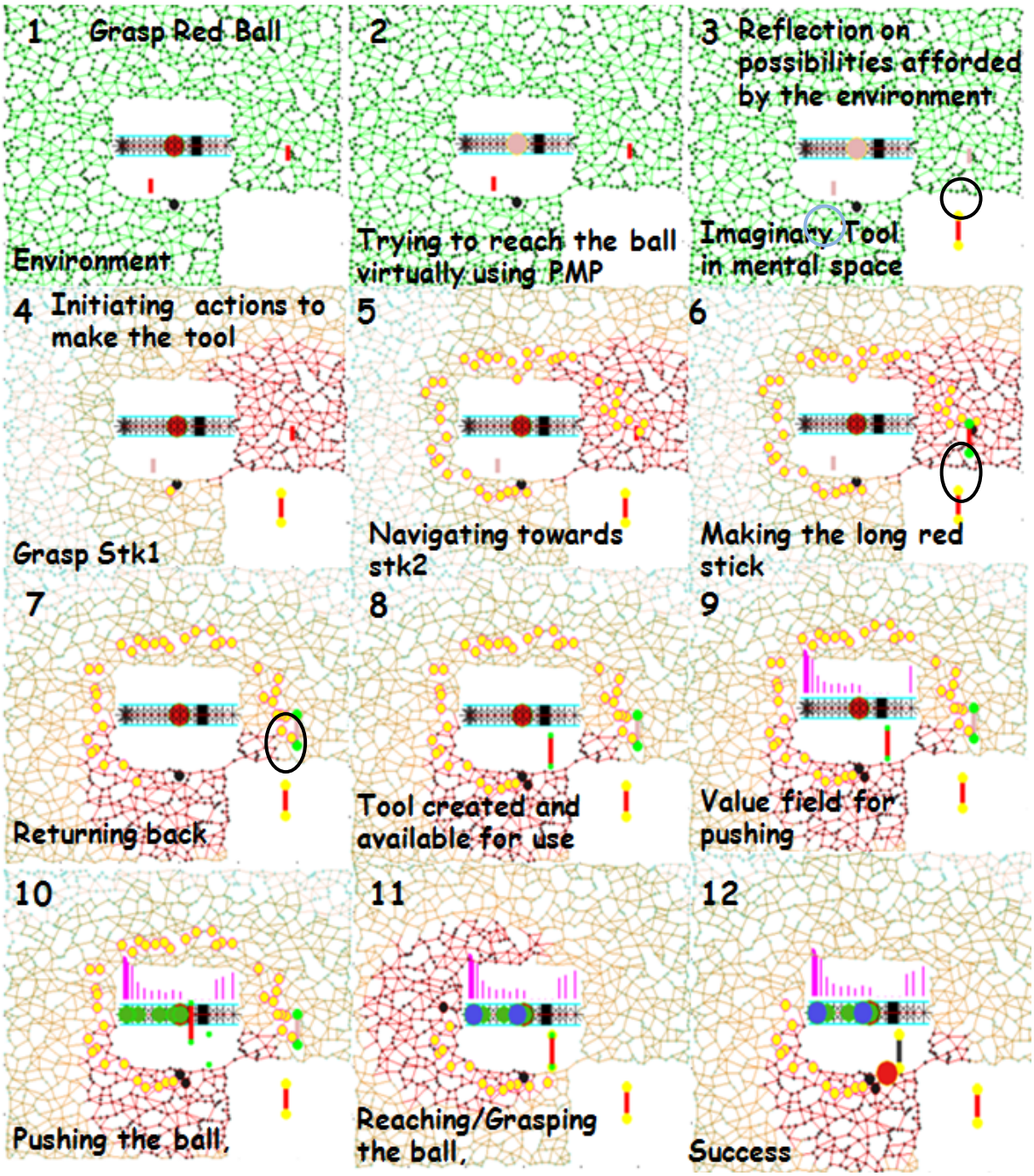


Figure 18. The goal of the robot is to grasp the red ball in the given environment. The robot evaluates the fact that it may not be able to get the goal directly using its end effector or by using the small red stick alone, but if it had two small red sticks then it can make a long enough tool (as suggested by its past experience), and this new tool subsequently available in the place map can be useful in reaching the goal. Now the robot has to initiate few sequences of actions to physically create a new tool (Panel 6) and use it to push the ball appropriately and then move one again to eventually grasp the ball sucessfully.

Simulating what it could do in case it had two small red sticks requires projecting the activity in the ASMS to the motor space using the intermap connections to find the set of executed actions in the past from this sensory situation. Now the system detects one more possible action (Adhesion) that could be executed from this new situation (and which was not possible with only one stick). This also confirms the fact that there were two red sticks in the environment in the past (even though only one is in front of the robot, Note that the information in the place map (which shows another small red stick in some spatial position) cannot always be relied upon because red sticks are represented the same way and look visually the same, there is no red stick 1 , red stick 2 written on the objects). The consequence of executing adhesion can be seen by moving from the action space to ASMS through motor modulated lateral connections. Now the robot has an imaginary object at its disposal and enters into the place map. Will that object be useful in the context of the active goal can be evaluated by trying to simulate a reaching action on the ball with the imaginary tool held in the gripper. Since this is successful, now the robot is left with the task of acquiring the long red stick, and it already knows the action that it needs to execute in order to have the long red stick. A sub goal related to the adhesion action sequence (which already exist as a primitive) is created now in the goal space. The first stick is grasped successfully after a simulation to reach it using PMP. Since only the spatial location of the other red stick is known, the robot has to initiate spatial navigation to reach the second stick, and make the long red stick. The spatial trajectory of the robot and the making of the new tool is shown in panel 5 and 6 respectively. We can now observe that the place map has changed appropriately, the two small sticks disappear and a long red stick is visible at the position where the tool was made (shown enclosed in a black circle). The temporary utility of this new object is disabled for other goals in the place map and the robot now plans its way back (Panel 7). Note the change in spatial value fields in the spatial map, also note that the long red stick is now displayed in a shaded color which indicates it is right now under consideration by the robot.

In panel 7, we see a new internal world in front of the robot, in which it has a new tool at it's disposal to realize the user goal. Simulating the consequence of pushing (Panel 8) now predicts the new location of the ball, initiating one more spatial movement due to rewards valued induced by the new displaced location of the ball (Panel 9, note change in value field again) predicts the simulated location of the body, from where triggering passive motion paradigm to reach the ball results in success. Now the complete plan of the actions rehearsed in the mental space in order to realize the goal enters into the plan descriptor and the robot is now ready to initiate physical actions in the world to hopefully realize the user goal and get its promised reward (Panel 12).

5.4.3 Nonexistent object: 'Grasping a Long Red Stick'

We now come to the third environmental scenario that looks quite similar to the scenario shown in figure 18 but with a small twist in the sense that the robot is now asked to grasp a long red stick and not a green ball. As usual the user goal is initialized in the goal space and the robot begins its process of chalking out the plan to realize the goal. However the place map that usually keeps track of the objects in the world does not have any object descriptor that identifies with a long red stick, triggering a local visual search also fails in this case (unlike previous cases where the goal object was available, and it was the forward inverse models that reported the simulated inconsistency in reaching the goal). Any inconsistency resulting in any action leads to a reflection of the possibilities that the environment affords in relation to realizing the current goal. Since the robot has two small sticks available to it and can simulate the fact that initiating the adhesion action on the small red sticks will result in a new object in the place map (the long red stick) which is exactly the object that it was asked to grasp.

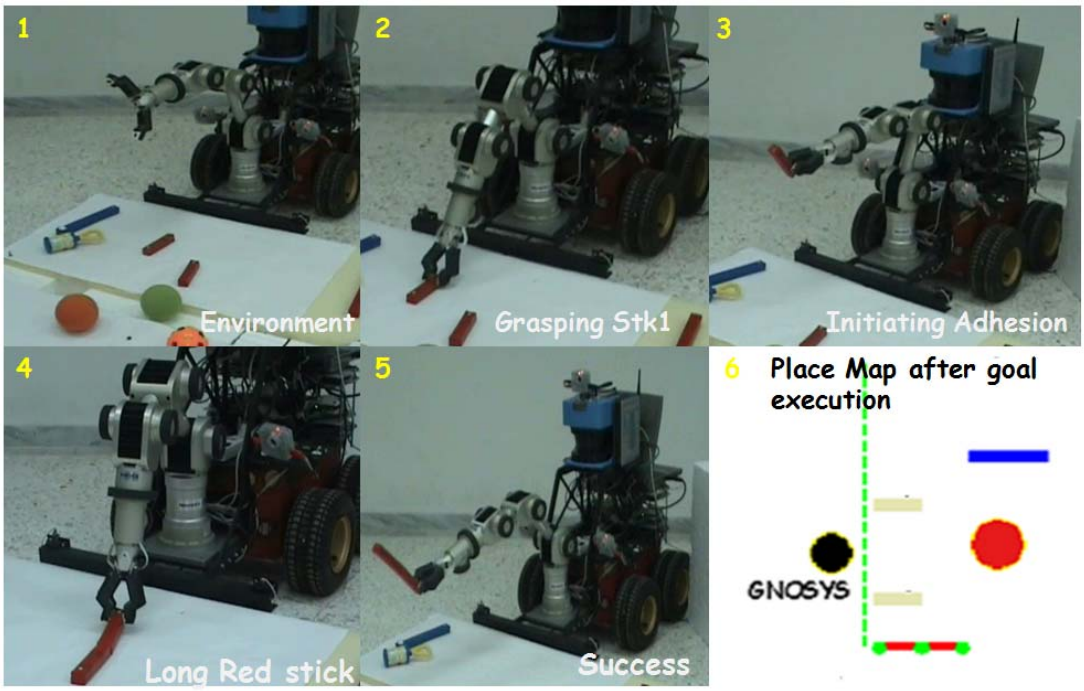


Figure 19. 'Grasping a Nonexistent object' : Any inconsistency in the execution (real/virtual) of any action, and anywhere in the reasoning-action generation architecture, triggers a higher level of reasoning either to exploit what the environment affords or go into exploration to learn more. Further, complexity at different levels are finely abstracted in the sense that even though the task of coordinating the coupling of two sticks is extremely complicated at the level of motor control, the abstract reasoning system does not have to worry about planning in the level of any sensorimotor variables involved in coordination of reaching, moving , pushing, adhesion etc.

Figure 19 shows the sequence of action initiated by the robot to realize this goal. Panel 6 shows the resulting state of the world after this goal was successfully solved. If immediately

the robot was now asked to grasp a red ball, , the situation will be exactly as the case we saw in figure 17, the only difference being that the tool used will be the long red stick which was created while realizing the previous goal. In case one of the red stick was not immediately present in front of the robot, the sequences of reasoning would have been similar to the situation in figure 18. In case the other red stick was located somewhere in space and is known approximately to the robot (through the place map) the goal would still have been realized however with a larger sequence of actions. In case no small red sticks were present in the place map, it would have led to low motivation and quitting. IN case one small blue stick and one small red stick was placed in front of the robot, the robot would have still explored the possibility of making a long red stick, by initially grasping the red stick and then going on a spatial exploration to find the other red stick (since it can simulate the fact that even it does adhesion to a red and the blue stick the world/place map is still going to remain the same). We can this observe how even making very small changes in the environment also causes a significant change in the required strategy, reasoning and actions needed to successfully realize a user goal.

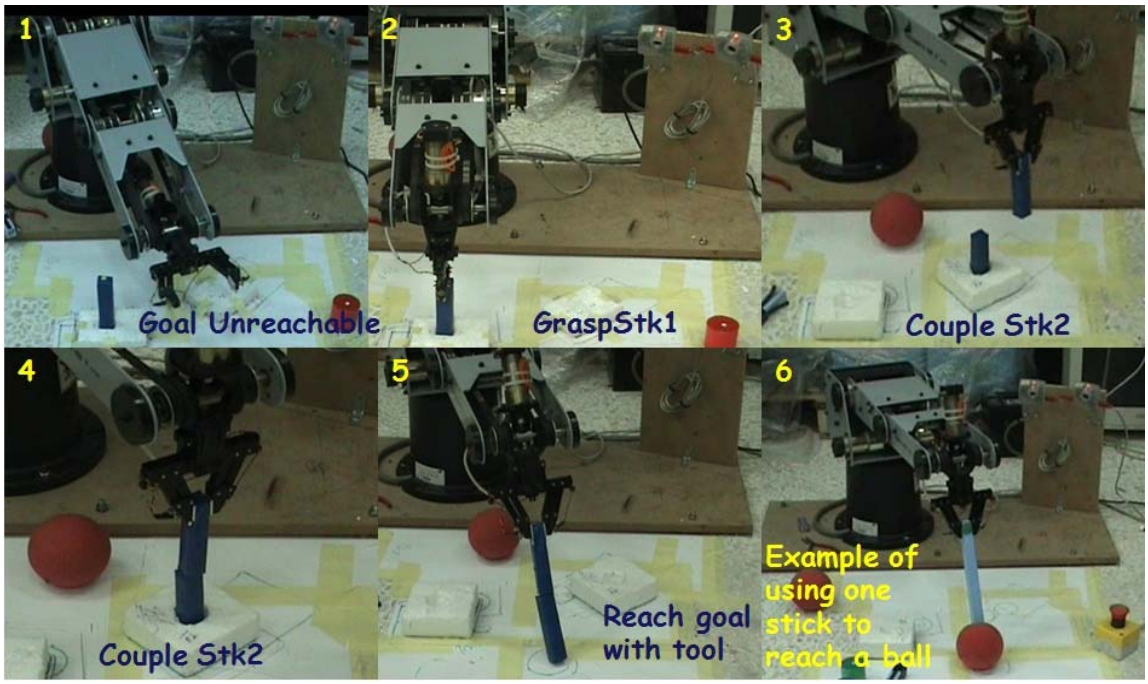


Figure 20. Before porting the architecture on the GNOSYS robot, we initially tried out the reasoning–action generation system on a simplified robotic platform consisting of a scorbot arm and 2 cameras. Figure shows some trial snapshots of similar experiments on this simplified (non mobile) robot.

5.4.4 On Quitting and Having a reason to quit

When traps were initially introduced in the tapping groove after the phase of learning the pushing SMS and the associated reward structure in the absence of traps, every trial the robot had an experience, an experience of contradiction because of the trap, an experience

of exploration which characterized its attempt to nullify the effect of the trap so as to realize the goal and an experience of being rewarded by the user/self in case of success (as shown in figure 4.19). We also saw how these experiences of pushing the ball in the presence of single traps in the trapping groove were represented in terms of fields and were superimposed in a task relevant fashion to give rise to a net resultant field that coordinated the dynamics of the pushing subsystem. Whenever a new pushing goal was presented to the pushing SMS, we could also observe goal/trap specific changes in the value field and the changes in pushing direction as a function of both the relative position of the hole, and the starting position of the reward/ball (figure 4.20).

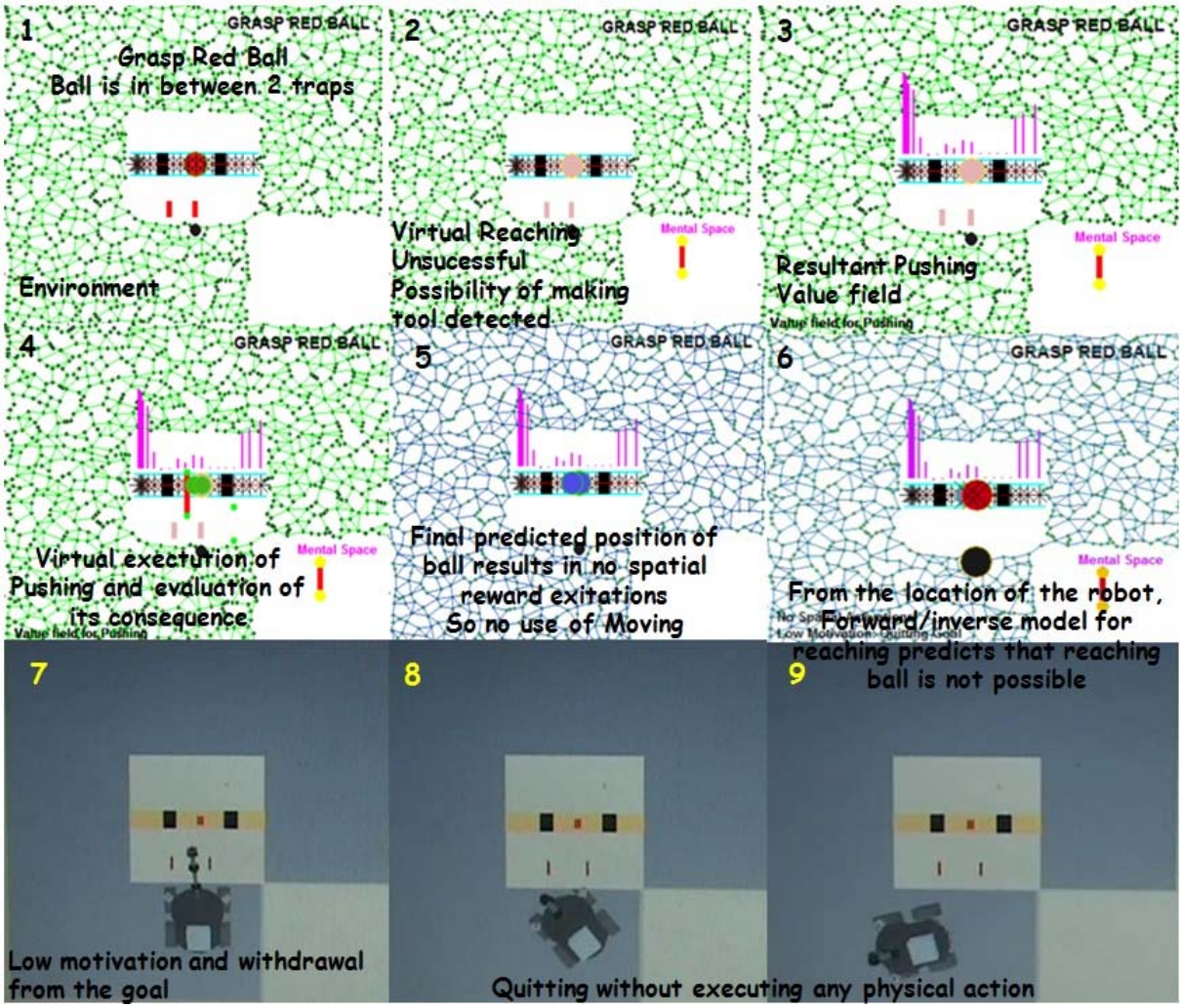


Figure 21. The user goal is to grasp the red ball placed in between the two traps. Even though the robot had past experience of learning to push objects in the trapping groove taking traps into account (chapter 4), they were all experiences with single traps placed at different locations along the groove. Can it generalize the previous experiences with single trap to the novel case of two traps, with the goal in between? What will be the resultant value field ? Looking at the value field what more can we predict about how the robot will behave in some other environmental scenarios, for example if the ball was on the other side of the either traps (i.e. not in between the two of them)? Of course humans can mentally evaluate that it's an impossible goal. Will the robot quit without executing any physical action at all? and will there be a reason to quit?

After presenting the robot with several reasoning scenarios in which goals, environmental objects were presented in different combinations (as described in the previous sections) along with single traps in different places, we were motivated to play a bit further with the traps in the trapping groove and observe the behavior of the robot. This led to the scenario shown in figure 21 (Panel 1).

The major change in this scenario as compared to the previously described ones is the introduction of multiple traps in the trapping groove. In the environment shown in figure 20, more over the goal (ball) is placed in between the two traps, and is hence impossible to reach or grasp or push it. Humans and even children may very easily figure it out, we do not know what would be the behavior of a capuchin or a chimpanzee in a similar environment. We can however analyze the behavior of the GNOSYS robot, and also peep inside its Gbrain to figure out what exactly made it behave the way it did (which is also an important feedback to the designer in order make appropriate changes in the computational architecture itself). We can also observe that the robot has the two small red sticks at its disposal when it is issued the user goal to grasp the red ball.

As usual the goal is initialized and the forward/inverse model for reaching predict that the ball is unreachable directly using its end effector. This leads to a reflection of possible opportunities afforded by the environment, and just like the previous example the robot successfully figures out that it can make a long red stick out of the two small ones by magnetically coupling them (hence we can see a long red stick in the internal place map). Now the robot is ready to simulate the consequences of pushing the ball with the imaginary tool and triggers the pushing dynamics. The resulting value field for pushing (which is a superposition of the default reward field and previous 'single trap' experiences based on their relevance to the current environmental configuration) is shown in panel C.Since pushing is done in the direction in which the resultant value field (reward) increases, it is easy to visualize the consequence of pushing under the influence of the value field shown in panel 3. In simple terms, just by looking at the resultant field structure we can see that if the ball is in between the two traps, it will end up being approximately in between them when the pushing dynamics becomes stationary. On the contrary if the ball was placed anywhere other than in between the two traps, it would have ended up being displaced to one of the corners of the table as a consequence of the simulation of the pushing action.

Panel 5 shows the final predicted location of the ball as a consequence of the mental simulation of the pushing action in the current environmental scenario. Moreover, this predicted position of the ball induces no spatial reward excitations on the internal spatial map as shown in panel 5 (since the ball is inaccessible from any spatial location in the environment). Hence the robot is able to evaluate the fact that there is no use of moving its body, unlike the previous cases where the virtual movement of the ball induced reward excitations in space, hence resulting in body movements that lead the robot close enough to

the goal such that it was directly accessible with the end effector. From the position the robot is currently located, it had already evaluated the possibility of reaching the ball directly using the forward/inverse models for reaching. It has also mentally evaluated what it could possibly do with the affordances provided by the environment (make a long stick and try to push the goal out of the groove).

Hence by 'moving in the mental space' the robot was able to evaluate the fact that 'there was no need to move in the physical space'! No wonder, it quit the goal, and also had a valid reason to Quit!

Atomic Cognitive Agents

> To myself I am only a child playing on the beach, while vast oceans of truth lie undiscovered before me…..
>
> ISSAC NEWTON

As concisely stated by the acclaimed neuroscientist V.S Ramachandran, with people from several disciplines tinkering around with different open problems, cognitive science research is right now entering into a 'Faraday era' in terms of discovering the general principles related to the structure and function of the 'three pound jelly', that on the first place makes all these 'tinkering and discovering' possible. Playing with a moderately complex robot situated in a moderately complex (and changing) world (inspired from animal reasoning experiments), in the last five chapters we attempted to develop and better understand the fundamental computational principles necessary to drive an artificial agent/robot to exhibit atleast some level of resourcefulness, purposefulness, flexibility and adaptability in its 'reasons and actions' while inhabiting unstructured worlds. This effort may just be equivalent to playing on the beach and we are fully aware of the fact that vast oceans of truth still lie undiscovered in the quest to better understand the 'forces and the causes' that shape our 'reasons and actions' and make us as explorative, intuitive, cognitive, expressive, emotional, irrational, unpredictable and conscious we really are. In this concluding chapter we will initially review what we have proposed so far and what we believe are the major contributions of this thesis. Finally, we will outline some crucial design principles like abstraction, structural self similarity, scalability, circularity, recursivity, goal directedness and domain agnostic nature that form the backbone of the proposed reasoning-action generation architecture and make it extremely powerful in terms of future up gradation and effortless portability on other robotic embodiments and humanoids.

6.1 The Journey so far

World is a mixture of structure and chaos. There are regularities, which can be exploited in order to counteract limitations (of perceptions, actions and movements) imposed by one's embodied physical structure. Species natural or artificial, which do this best have the greatest chances of survival. Humans and even a wide circle of animals (as shown by the

emerging results from animal cognition) during the course of their lifetimes gradually master this ability of exploiting environmental affordances by coherently integrating different streams of information: a) Bottom-up (sensory); b) Top-down (internal drives, goals, reward expectancy); c) Memory (a set of contextually relevant past experiences); d) Virtual (a set of simulated experiences). Further, for an embodied cognitive agent (in contrast to software programs and pen and paper based exercises) the problem of exhibiting intelligent behaviour in changing environmental scenarios is not merely about logic, induction, probability, generalization, propositions etc but about *action*. It is about the effects of intervention and is hence intimately tied to perceptions and actions that take place in physical worlds, imagined worlds and counterfactual worlds. In this thesis, we developed and investigated a conceptual architecture for reasoning and action generation (real and mental) with an aim to empower an artificial robot with some preliminary ability to virtually manipulate neural activity in its mental space in order to exhibit flexible goal directed behavior in the physical space. In the previous five chapters we gradually extended the discussion starting from the basic robotic platform, the design of the environmental set up (playground), creation of the different internal models for action generation, an architecture for reasoning about actions and some interesting simulations and experiments we conducted on the robot.

For an embodied cognitive agent (in contrast to software programs and pen and paper based exercises), the problem of exhibiting intelligent behaviour in changing environmental scenarios is not merely about logic, induction, probability, generalization, propositions etc It's about *action*. It is about the effects of intervention and is hence intimately tied to perceptions and actions that take place in physical worlds, imagined worlds and counterfactual worlds.

Some of the highlights and major contributions of this thesis are briefly summarized below.

- *Creating biologically inspired habitats of varying levels of complexity and subtlety for robotic artifacts (for gaining sensorimotor experience and validating the cognitive architecture).*

  Even though creating an environment consisting of two tables, few sticks, cylinders and ball may seem absolutely trivial, it was in fact the environment (and the robot) that drove the development of different computational models presented in this thesis. It is not the content of the environment, but the underlying purpose behind its creation, the thinking about range of possibilities that it affords for a robot to learn, the thinking about range of experiments that it affords for an experimenter to test with the robot (in fact the many scenarios (like the quitting task) were never planned and were just a gradual evolution of the research seeing the behavior of the

robot and seeing the shortcomings in the computational models that is driving the generation of that behavior), the need to tap the most suitable (and computationally challenging) experiments from animal cognition, the need to balance complexity of the task with limitations of technology (sensing, energy, manipulation etc for example if the trapping groove was a tube like the real experiments from animal reasoning it would have been very complex to visually recognize the ball inside it) are some of the factors that make this task absolutely nontrivial. The several advantages that could be extracted from such environments for embodied cognitive science research has already been dealt with in chapter 2 and we will not reiterate them here. One final comment we would like to make is that the environment can serve as a learning experience not only for the robot but for the experimenters themselves. In fact the fully stable and final architecture presented in this thesis were either revisions or completely modified versions of the computational models that could not scale up to the complexity imposed by different environmental scenarios ().

- *Development of biomimetic, force-field based forward/inverse models for reaching (with arm/arm+tool, subsequently implemented GNOSYS robot) and for complete upper body coordination in Humanoids (subsequently implemented on the baby humanoid iCub).*

Exploiting the solutions offered by the physics of passive motion (Passive Motion Paradigm, Mussa Ivaldi et al 1988) and the concept of terminal attractors, we presented a simple, distributed computational framework for representing and solving a range of ill-posed coordination problems arising in redundant robotic platforms like the GNOSYS robot and the baby humanoid 'iCub'. A key feature of the architecture is that the same computational model is used to support mental simulations related to reaching movements (and using tools) employed by a higher level reasoning process and the actual delivery of motor commands during movement execution. The difference is that, in the latter case, the motor outflow interacts with the peripheral circuitry of the motor system (spinal circuitry and mechanical properties of the neuromuscular apparatus in the biological domain or its simplified equivalent in the robotic domain). In this sense, we not only

We not only address the problem of planning the motion of highly redundant body, but also address the problem of allowing reasoning/cognitive layers to access/modulate efficiently such computation.

The system does not "search" for a solution. On the contrary, it has access to a whole family of solutions stored in the form of a holographic memory form which it "discovers" the one that is closest at hand.

address the problem of planning the motion of highly redundant body, but also address the problem of allowing reasoning/cognitive layers to access/modulate efficiently such computation. The computational process of relaxation in the attractor landscape of the PMP model is similar to coordinating the movements of a puppet by means of attached strings, the strings in our case being the virtual force fields generated by the intended/attended goal and the other task dependent combinations of constraints involved in the execution of the task. Further, the force fields we are considering do not really describe the biomechanical forces at play during the execution of movements, but are just computational metaphors that describe the dynamics of the internal computational engine. We then described how the computational framework of PMP can be extended to create composite forward inverse models (with /without branching nodes) in order to coordinate real/virtual body movements under the presence of a range of different task specific constraints and coordinating the motion of two arms (with/without external objects coupled with them). Computational properties like robustness, run time optimization, fast task adaptation, interference avoidance and local to global computation that makes the proposed Forward/Inverse models both biologically plausible and extremely useful in the control of complex robotic bodies were discussed in detail in chapter 3. As a final note, there are no predefined cost functions/optimization constraints in the model and the system does not "search" for a solution. On the contrary, it has access to a whole family of solutions stored in the form of a holographic memory form which it "discovers" the one that is closest at hand.

- *Learning a mental map of the GNOSYS playground, realizing goal directed planning in the learnt spatial sensorimotor space, learning when to optimize what while navigating in the play ground and switching between exploration and normal dynamics in response to changes in the world.*

A past experience influences the value field dynamics based on its relevance with respect to the current goal for which the field structure is being computed.

An internal representation of the GNOSYS playground was learnt using a growing neural gas algorithm (Fritzke et al), suitably extended to take into account issues related to motor coupling (based on the idea of Sensorimotor maps of Toussaint et al 2006, 2004). This learnt neural gas layer (Spatial SMS) not only self organizes sequences of sensorimotor data generated by the robot through random motor explorations (through the loop of real experience) but also sub symbolically represents the forward inverse functions of various sensorimotor dependencies (that is encoded in the connectivity structure where ever and

when ever learning takes place). A novel computational scheme to organize the dynamics of the Spatial SMS and its bidirectional interactions with the action space was proposed in 4.3. The bifurcation parameter introduced in the activation dynamics automatically triggers a transition of system behavior from normal dynamics to explorative dynamics based on a measure of coherence between the bottom up (sensory) and top down (simulated) information.

In addition to the activation dynamics that moves neural activity back and forth between the SMS and action space, an additional dynamic process which can be thought as an attractor in the sensorimotor space performs the function of organizing goal oriented behavior. The novel feature introduced in the second dynamic process (i.e. the goal induced quasi stationary value fields) is the possibility of continuously incorporating new (good/bad) experiences encountered by the robot in the value field dynamics. A past experience influences the value field dynamics based on its relevance with respect to the current goal for which the field structure is being computed. Section 4.6 presents an example of how allowing relevant past experiences to shape the field structure can implicitly cause a switch in constraints that need to be optimized based on the goal at hand. A novel 3 point heuristic to distribute rewards in continuous, high dimensional state spaces was proposed and subsequently demonstrated in the experimental scenario considered in section 4.6. Finally, experimental results from the robot's spatial behavior that touch several important (and controversial) issues like representation, goals, sub goals, optimality, contradictions and reinforcements are presented in the ensuing discussions.

- *Pushing and Pushing Intelligently*

The pushing sensorimotor space and reward structure were learnt by repeated trials of random explorative pushing of the goal in different directions along the groove, followed by an attempt to grasp the goal (by moving and reaching). As previously mentioned in chapter 4, all dynamics, learning procedures, mechanisms to deal with constraints were same in all the internal models, the only difference being the sensorimotor variables on which they operate. In the pushing sub system, we also showed how energetic issues can have their effects in the learnt reward structure if there are multiple solutions to get the reward successfully through pushing. Once the pushing SMS and reward structures were learnt, traps were introduced in different locations along the trapping groove. In section 4.9 we showed how an experience of contradiction because of the trap, an experience of exploration which characterizes the attempt of the robot to nullify the effect of the trap and an experience of being rewarded by the user/self in case of success, is used by the robot to learn trap specific adaptations in the value field. Just after four experiences of playing with traps, we could observe both trap specific changes in the value field (that drives the

dynamics of the pushing SMS) and changes in the pushing direction as a function of both the relative position of the trap, and the starting position of the reward/ball. In the concluding section of chapter 4, we showed how the internal models for reaching, spatial navigation and pushing form a closely connected network, predictions of one slowly driving the other or providing enough information to make the other mental simulation possible. Starting from a mentally simulated body (coming from spatial sensorimotor map), the robot could now mentally simulate a reaching action directed towards a mentally simulated position of the goal target (coming from the pushing sensorimotor space), using the forward inverse model for reaching (passive motion paradigm). In other words, GNOSYS now had the seamless capability to mentally simulate sequences of actions (in different sensorimotor spaces) and evaluate their resulting perceptual consequences: **"…** since there is a trap there, it is advantageous to push in that direction; if I push in that direction, the ball may eventually go to that side of the table; in case I move my body closer to that edge, I may be in a position to grasp the ball **…".**

- *Reasoning and most importantly dealing with an ever-changing world and the information that it keeps generating*

  GNOSYS now had the computational architecture needed to simulate sequences of 'actions and perceptions' in multiple sensory motor state spaces in order to realize a high level goal. What was missing was a higher level, more abstract internal model that learns to orchestrate these virtual experiments (actions and their simulated consequences) in accordance with internal goals, especially when inhabiting unstructured and continuously changing environments. The computational complexity in the problem of realizing an user goal like 'Reaching a Red Ball' in a dynamic and changing environment results from the fact that before reaching the red ball itself with end effector, there may be several intermediate sequences of real/virtual 'Reaching', 'Grasping', 'Pushing', and 'Moving' etc directed at 'potentially useful ' environmental objects, information regarding which is specified by the root goal itself (which was just 'reach the red ball'). So before realizing the root goal, the robot has to 'track down' and 'realize' a set of useful subgoals that 'transform' the world in ways that would then make the successful execution of the root goal possible. This necessity motivated the natural transition of the computational architecture from reasoning at the level of 'force flows' to the level of 'situation plans' both systems being loosely coupled with each other and with the world in which they operate. New connectivity structures needed to move from objects to actions (intermap connections, since there could exist learnt experiences of object action that initially don't fetch any value at all), to move from situations to situations

(conceptual lateral connections, that cause shifts in activity in the sensorimotor without the mediation of any motor action), growing reward and quality matrices (to compute goal specific value fields) and their associated dynamics, interactions were formulated. New memory structures like the monitor process, the place map and the goal space, their associated interactions with the reasoning system (at a conceptual level and at a software level in terms of message composition, decomposition, communication) in order to bring stability inside the computational architecture irrespective of a continuously changing internal and external world were defined and implemented. Ways to loosely couple reasoning and exploration under the same computational/informational flow were devised and implemented. In many ways, the beauty, flexibility and stability of the architecture emerges from the way it manages the massive amount of information that is generated continuously as a result of virtual execution of actions, real execution of actions and the consequent changes in they cause in a complicated environmental set up. In very general terms, the mental simulations initiated by the abstract reasoning process is directed towards clustering the available world into objects which are potentially useful, and those which do not afford any utility value in the context of the currently active goal. Useful objects become subjects of sub goals and enter the goal space from where they become subjects of different actions, successful realization of which was expected to change the internal world (mental space) in ways that aid the realization of the user goal. Four different reasoning tasks (a case of direct tool use, an artificial reconstruction of Betty's Novel tool making task, the trap tube paradigm and a case of quitting without initiating any physical action) that we believed reflect different aspects of performance of the reasoning system on the robot were presented. We finally note that the number of possible environmental scenarios that can be created and tested on the reasoning system driving the behaviour of GNOSYS robot is limited only by the reasoning capabilities of the experimenter himself.

6.2 'On Dance and Chance' - When Actions become Affordance

Perceiving the order and invariance inherent in the infinite expressions of nature has always been the fundamental quest for any kind of science. While skill corresponds to mastering its inherent complexity, creativity corresponds to mastering its inherent simplicity.

The computational architecture that drives the reasoning and action generation system of the GNOSYS robot is undoubtedly a complex system composed of several subsystems each having their own levels of complexity and interacting with each other several complex ways. The goal behind development of this architecture was not just to make the GNOSYS robot reason and act intelligently in its little world but also to look for general computational principles that can scale up with the complexity of newer worlds and goals, allow effortless portability of the architecture on more complex robotic platforms, allow quick upgrading/

relearning the existing internal models when ported on other robots, allow learning of new internal models without seriously affecting the already existing interactions, allow hassle-free interfaces with computational models (developed by other groups) that support the reasoning architecture (mainly visual perception) etc. In this section, we will look for the order and simplicity inherent in the computational architecture that we believe makes all these sub goals (future goals) achievable. We will also exploit this opportunity to highlight general design principles like abstraction, approximate self similarity over scales, circularity, scalability, recursivity, goal directedness and domain agnostic nature that form the backbone of the proposed architecture and endow it with seamless flexibility and adaptability needed to face the real world.

6.2.1. Thinking Abstraction

Beginning from the innermost core, despite the diverse functional roles performed by the various computational models for action generation presented in the previous chapters, we can still extract some general principles regarding their internal structure and the mechanisms they employ to realize their respective local goals. In simple terms, the atomic structure of every computational model was composed of the following four elements:

a) a set of sensorimotor variables (that characterizes the sensory and motoric scope of the model)

b) a set of connectivity structures (that mediates the interactions between the sensorimotor variables)

c) a set of value fields (that organize goal directed behaviour in the respective sensorimotor spaces taking into account task relevant constraints represented in terms of superimposed fields)

d) a set of trajectories (in the respective sensory and motor spaces, that characterizes the real/mental behaviour of the system in response to the currently active goal)

All the models were further capable of continuously learning/ incorporating newer subtleties discovered in the environment concerned with the respective sensorimotor spaces they represent and seek to control, using very much the same principles for acquisition, self-organization, transformation and use of information. By loosely demarcating the internal dynamics, connectivity (associated neural networks), associated reward structures of every computational model local to itself , we were beginning to abstract away the complexity at the local level , when it comes to dealing with complexity at a higher, more global level. For example, a reasoning system that wants to evaluate the possibility of using a stick as a tool or the consequence of pushing a ball with the grasped stick, need not be concerned about the complexity at the level of end effector trajectories, necessary wrist orientation to grasp the stick, the joint angles needed to coordinate the body or the direction of pushing based

on the location of traps. All it needs to learn is to find the right person (or the variable) to do the job.

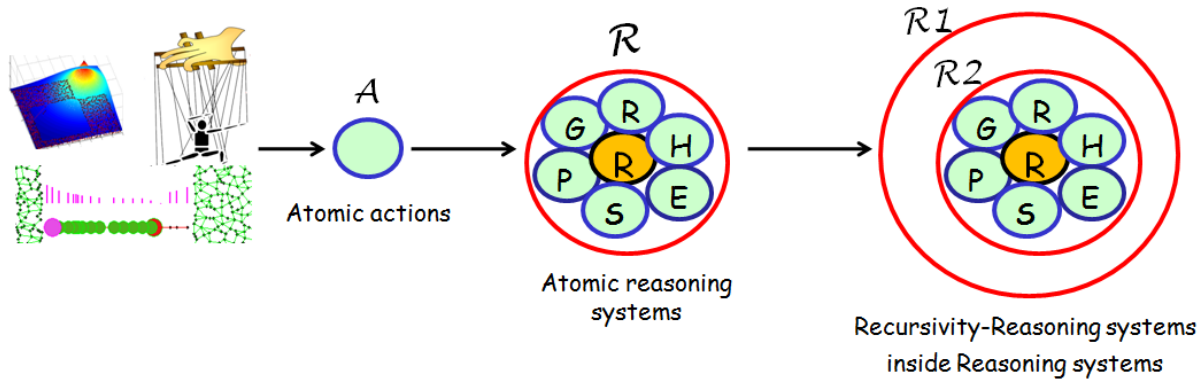


Figure 1. All dynamics, connectivity, reward structures influencing behavior at one scale are finely abstracted away from the complexity emerging at the higher scales. Yet approximate self similarity across scales is a fundamental feature inherent in the architecture. Be it the trajectory of abstract atomic actions generated through reasoning, be it a trajectory of the end effector or a the trajectory of the stick held by the end effector or a trajectory of the ball being pushed by the stick, or a trajectory of the body attracted by the ball, it the same computational principles which are used again and again in different contexts.

The internal models themselves now look at the snapshot of the locally available world (in the place map or goal space) to acquire more detailed information about the issued their goals (spatial position, constraints) construct their composite task specific forward/inverse models, simulate their dynamics and send only two messages, one to the higher cognitive layer that issued them their local goal summarizing the outcome and one to the place map to update the world as a result of their intervention (in case it was mental). Before extrapolating the architecture to higher scales, figure 1 shows a simple cartoon representing where we have reached in the journey so far.

> Along with abstraction and recursivity, circularity is a principal feature inherent in the reasoning-action generation architecture. The loop between perception and action is closed at all levels of hierarchy, inconsistencies at one level automatically becoming an affordance to act/reason/explore at some other level.

At this micro scale we can also observe the fact that outcome of the performance of one atomic action automatically creates an affordance for some the other action to take place (a visual search makes reaching possible, reaching makes grasping possible, pushing makes moving possible and so on at the bottommost level of abstraction in the architecture). In addition to abstraction and recursivity, the architecture at all levels of implementation is also inherently circular in nature (pictorially depicted in figure 2). In other

words, the loop between perception and action is closed at all levels, inconsistency at some level automatically forming the basis of reasoning/acting/exploring at some other level.

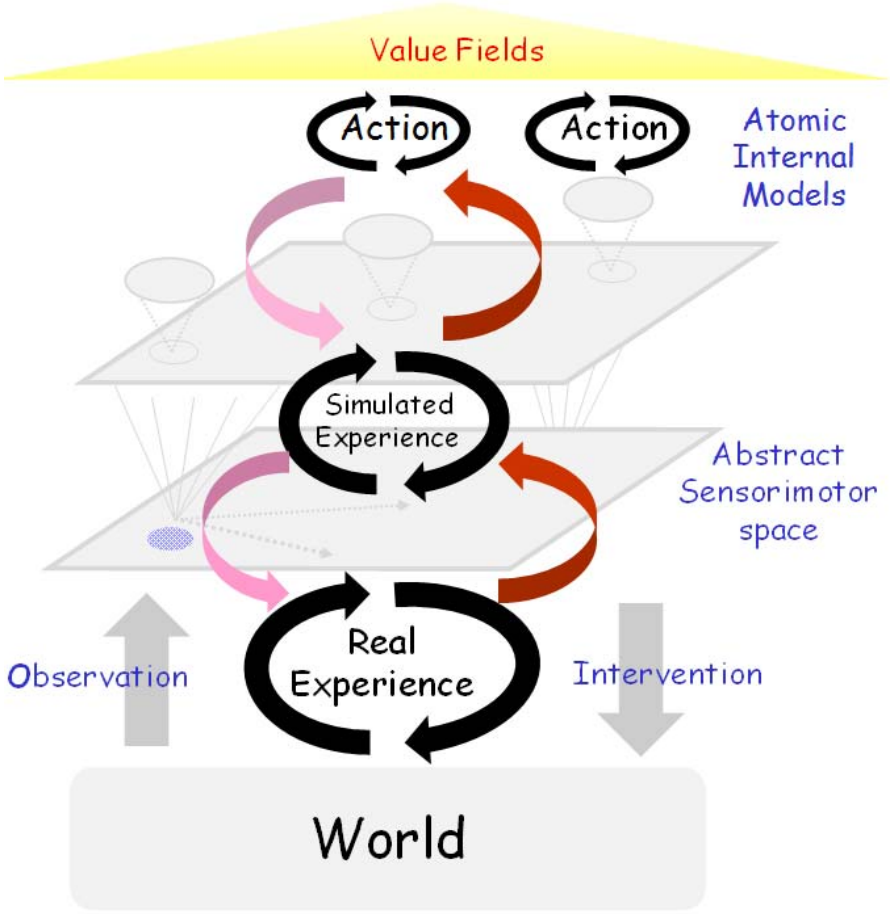


Figure 2.Circularity is a principal feature inherent in the reasoning-action generation architecture. The loop between perception and action is closed at all levels of hierarchy, inconsistencies at one level automatically becoming affordances to act/reason/explore at some other level.

6.2.2 Thinking Portability

A side effect of having high level of abstraction, well defined interfaces (computational, software, hardware) and minimum interference between different modules is the fact that it makes the task of porting the reasoning-action generation architecture on different robotic platforms (of varying complexities) almost effortless. Further, all computational modules can be locally improved/trained without disturbing the global interfaces and behavior. Figure 3 shows some snapshots of the architecture 'in action' on the three different robots we had the opportunity to work with during the course of the development of this work.

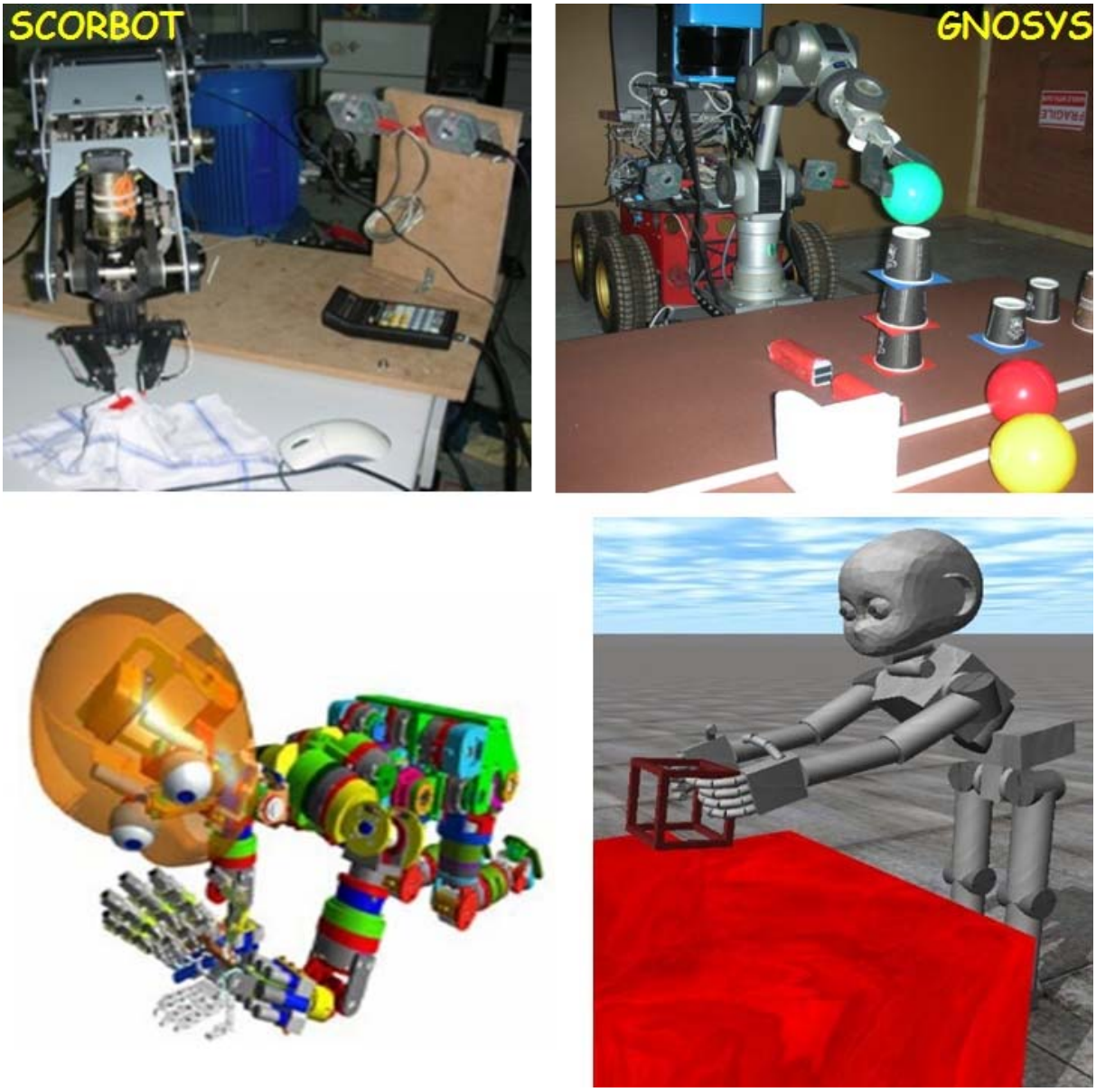


Figure 3. Modularity, abstraction, well defined interactions and minimum interference between different computational models facilitate almost effortless portability of the reasoning-action generation architecture to different robotic platforms with arbitrary levels of structural complexity.

6.2.3. When goals compete for the body

Let us now abstract the architecture a bit further, now thinking at the level of the robot itself as a cognitive agent with multiple goals (and rewarding opportunities). It must be quite clear by now what the next level of abstraction should be. Just like actions acted as variables for the reasoning system, reasoning systems themselves will now act as variables at the level of a cognitive agent with multiple competing (and cooperating) goals.

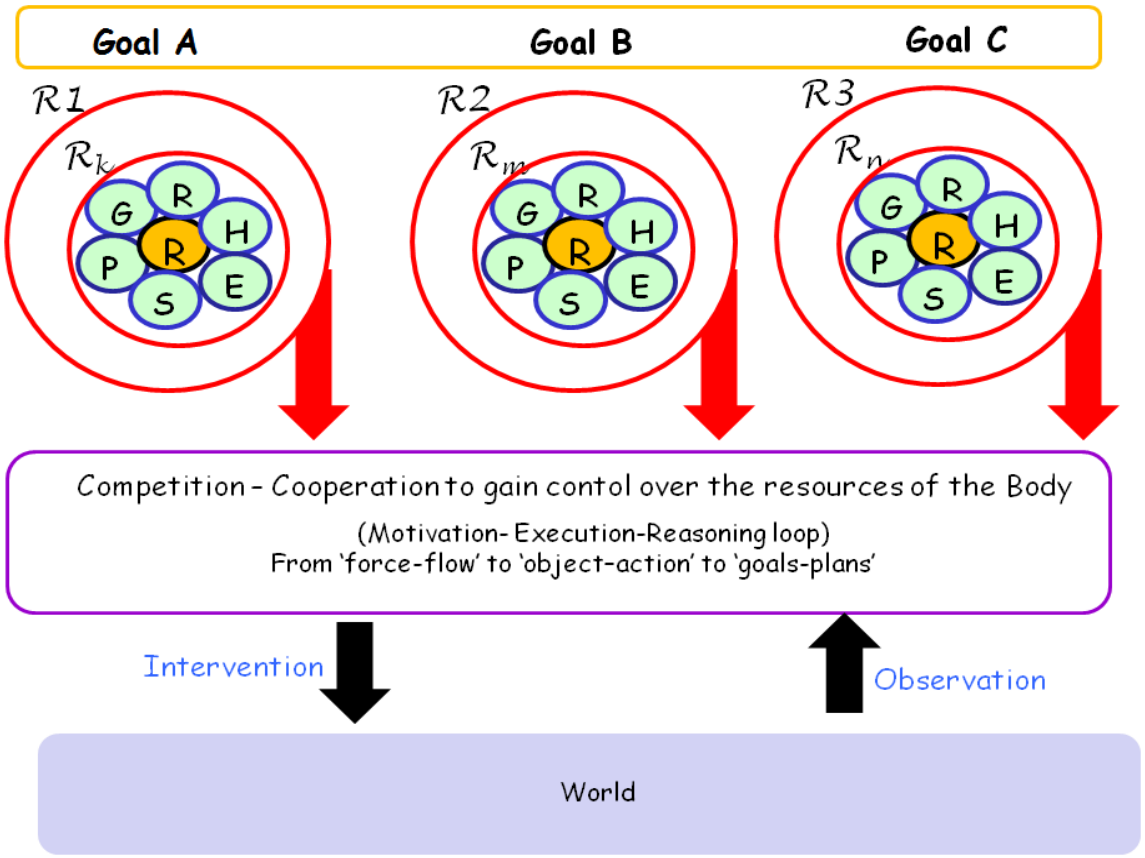


Figure 4. Abstraction Yet Again: At the level of the Robot, multiple goals compete/cooperate for the resources needed to make their plans a reality. Every root goal instantiates its own reasoning 'object' R_n that autonomously connects to the locally available world and the currently existing knowledge to propose actions necessary to realize it. At this level it becomes possible to exploit the seamless power of mental simulation a very high level, to sequence plans proposed by different reasoning systems in ways such that they can cooperate effectively (an opportunistic *out of turn* execution of the request/action of creating a long red stick can modify the world and afford a tool to successfully realize a previously requested user goal of grasping a red ball). No wonder, mental space is a natural site for running such mental simulations. Hence, by moving in the mental space (which is inexpensive in terms of resources) it often becomes possible to move intelligently and efficiently in the physical space (which often carries a cost in terms of use of environmental and body resources and most importantly energy).

At the level of software the reasoning system is just one more class and any number of instantiations of the class i.e. reasoning objects operating with their local user goals (just like actions operated with their local goals issued by reasoning) can be created on the fly. The basic idea can be appreciated without entering into implementation level details using several analogies. As an example, let us consider that we had a Dog class that defines what it is to be a Dog object, and all the "dog-related" messages a Dog object can act upon like bark, fetch whatever. Once we have such an interface, it is possible to make as many dog objects like Rover, Fido etc and begin visualizing the orchestra at a much higher level. Figure 4 shows a pictorial representation of the abstraction in the architecture seen at this level. Every goal now instantiates its own reasoning object that begins its life as an autonomous execution

entity in the brain of the robot. We must note that at this level there is competition at the level of goals to express the actions proposed by their reasoning processes in the physical space using the resources of the body. More over actions executed by the robot in order to realize some goal can retard the progress towards realization of some other goal (For example, if a person has a goal1 to become very rich and also has a goal2 to spend flamboyantly, assuming that the resources are limited the actions executed to realize one goal will automatically reduce probability of realization of the other, assuming that there are no unexpected changes in the world). We can visualize this even in the simple environment of sticks and cylinders that we created for our robot. If the robot had a goal1 to grasp a small red ball (in a scenario like betty's task) and goal 2 to transport a small red stick to some spatial location, in case actions for goal 1 is executed first (making a long tool), it is impossible to realize goal2 (since small red sticks no longer exist in the world). There can also be many situations, where opportunistic out of turn executions of actions needed to realize some goal, automatically created more affordances to successfully realize some other goals. Mental space being inexpensive in terms of resources is the natural site to perform such experiments. Since any change occurring in the world as a result of virtual simulation of different actions is reflected in the place map, the robot can see what changes the actions requested by different goals cause to the world (and how they may be useful/harmful in realization of other goals. Based on these virtual experiments the robot can dynamically alter priorities for execution of different actions so as to gain maximum rewards.

> Every goal now instantiates its own reasoning object that begins its life as an autonomous execution entity in the brain of the robot.
>
> By moving in the mental space (which is inexpensive in terms of resources) it often becomes possible to move intelligently and efficiently in the physical space (which often carries a cost in terms of use of environmental and body resources and most importantly energy).

6.2.4. A 'WITS' enabled World

Being a little bit speculative again, considering the fact that the reasoning and action generation system is already ported on different robotic bodies, can we abstract the architecture one step again, now at the level of the world (at least in the simulator)? Assuming that we couple the reasoning action generation system with the necessary perceptual systems (based on the nature of the embodiment), if we abstract once again now at the level of the world, the complete cognitive architecture becomes just one more variable G, suitably embedded in some robotic body B.

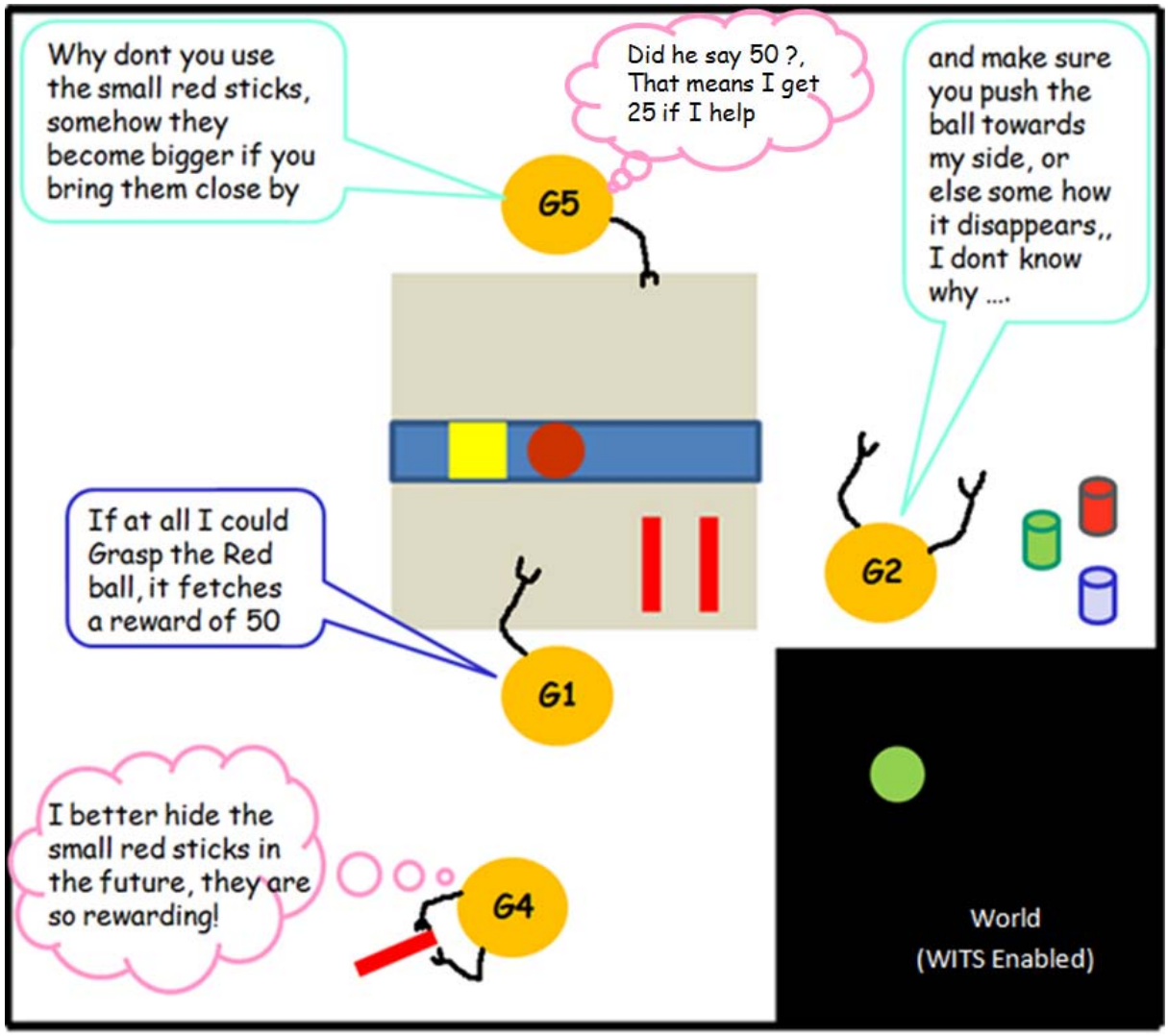


Figure 5. At the level of the world, agents compete; at the level of the agent, goals compete; at the level of goals, reasoning processes compete; at the level of reasoning, actions compete; at the level of actions, fields compete.

Several such atomic cognitive agents (G1, G2..G_N) can now exist in a virtual play ground, each making pseudorandom explorations to develop their own global knowledge space (beginning from learning the primitive actions, to learning what is possible with the different objects in the more complex playground). Assuming that the playground was a bit more diverse, with more objects, more affordances, appropriately distributed in space such that different agents learn different things (some of which being complementary to what others need) and further assuming that there was some primitive wireless information transmission service (WITS) using which the agents could communicate (at least their root goals), we can create situations that encourage social behavior, cooperation between agents (and hence

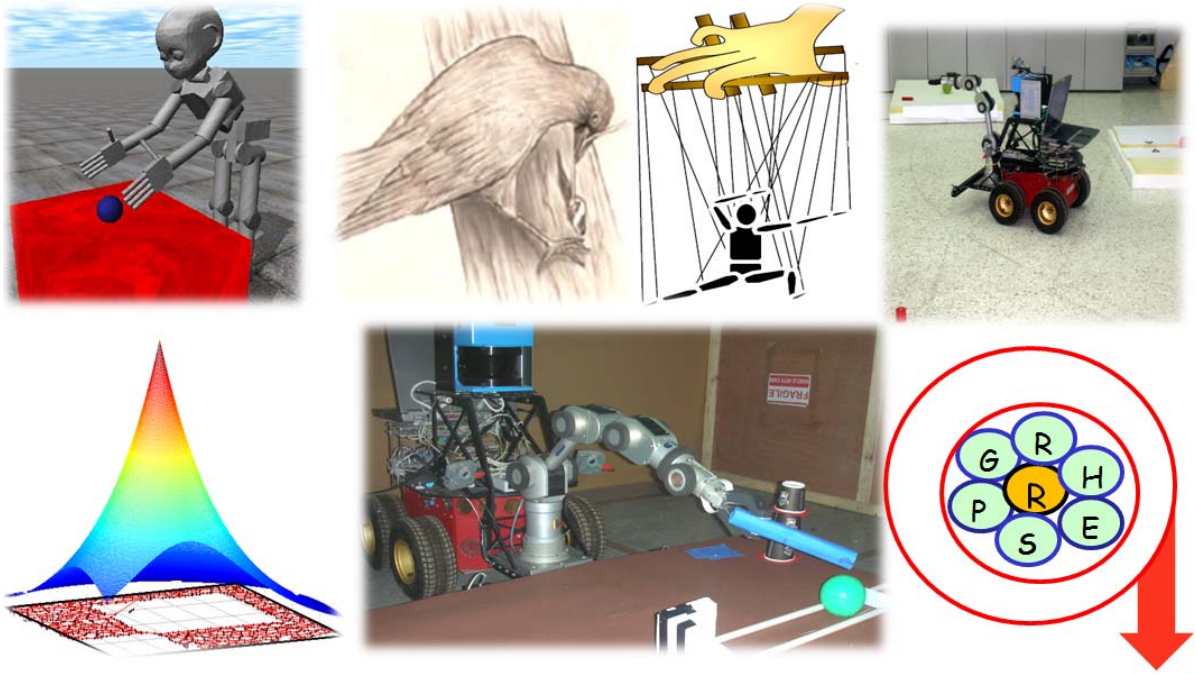
being able to share each other's rewards, reinforcements given by the user) and their collective co evolution in the world 'W'.

A very simple scenario could be like G5 has had experience with magnetized sticks and knows about exploiting their additional affordance, G1 has a goal to grasp the red ball (Betty's scenario again) and G2 has some experience with pushing in the presence of traps. In case G5 had no seriously rewarding goal (of itself) to act upon it could simulate the goal of G1 and either communicate the strategy through WITS or physically act on the world and make the long stick hence creating a new affordance for G1. In this case we can enter into issues like reward/penalty sharing among agents, knowledge transfer between sensorimotor spaces of different agents, in many ways a totally new world of complexity. Further, since resources in the virtual world will be limited, there will be competition among agents to gain control over resources available in the world, like in the cartoon G4 is reasoning about the value of the red sticks and actions it should initiate to gain control of that object forever. Agents who's interests are complimentary could now begin to from small groups (realizing that being together is more valuable to maximize rewards given by the user and minimize expenditure of one's own resources like energy).

At the level of the world, agents compete; at the level of the agent, goals compete; at the level of goals, reasoning processes compete; at the level of reasoning, actions compete; at the level of actions, fields compete. The dance of one dynamical system becomes a chance for some other dynamical system to evolve in space-time.

Once again we see familiar principles of competition, cooperation, neighborhood, reinforcements, associations and value fields at play, now at the level of the world. The outcome of the performance of one atomic action automatically creates an affordance for some the other action to take place. Actions executed by one arm, becomes an affordance for the other arm, actions executed by one robot becomes an affordance for some other robot. In short, the dance of one dynamical system becomes a chance for some other dynamical system to evolve. At the level of the world, agents compete; at the level of the agent, goals compete; at the level of goals, reasoning processes compete; at the level of reasoning, actions compete; at the level of actions, fields compete. It was the same principles operating at different levels of abstraction. And finally, the wireless information transmission service (WITS) that facilitates one atomic agent to control/manipulate the mental space (thoughts, actions and the body) of some other agent points towards nothing but the ultimate form of abstraction and the supreme faculty of the human race 'Language'. Running the complete orchestra is something we definitely look forward to in the future and this thesis was just a small brick in the grand scheme of things.

Finally, we take this opportunity to thank the interested reader for 'being onboard' this journey and depart with a picture that summarizes the previously written 200 pages of words.



A pictorial conclusion

Bibliography

Abbot,L., Sejnowski,T.J. (1999). Neural Codes and Distributed representations. MIT Press.

Abend, W., Bizzi, E., Morasso, P. (1982). Human arm trajectory formation. Brain, 105, 331-348.

Amari, S. (1977). Dynamics of patterns formation in lateral-inhibition type neural fields. *Biological Cybernetics, 27*, 77–87.

Anderson, M.L. (2003). Embodied cognition: A field guide. Artificial Intelligence 149(1):pp. 91-130.

Atkeson, C.G., Hale, J.G., Pollick, F. (2000). Using Humanoid Robots to Study Human Behavior, IEEE Intelligent Systems, 15, 46-56.

Baillieul, J. (1985). Kinematic Programming Alternatives for Redundant Manipulators. In IEEE International Conference on Robotics and Automation, 722–728.

Baltzakis,H. (2004). A hybrid framework for mobile robot navigation: Modellign with switching state space networks. PhD Thesis, University of Crete.

Boysen, S.T, Himes, G.T. (1999). Current Issues and Emerging Theories in Animal Cognition. Annual Reviews of sychology 50:683-705.

Bizzi, E., Mussa Ivaldi, F.A., Giszter, S. (1991). Computations underlying the execution of movement: a biological perspective. Science, 19, 253, 287-91.

Blair, H.T., Cho, J., Sharp ,P.E. (1998). Role of the lateral mammillary nucleus in the rat head direction circuit: a combined single unit recording and lesion study. *Neuron* 21: 1387–1397.

Bluff, L.A., Weir, A.A.S., Rutz, C., Wimpenny, J.H. & Kacelnik, A. (2007). Tool-related cognition in New Caledonian crows. Comparative Cognition & Behavior Reviews 2: 1-25.

Brooks, R.A. (1997). The Cog Project. Journal of the Robotics Society of Japan, 15, 968-970.

Byrne, R. W. (2007). Primate intelligence. In Henke, W, Rothe, H & Tattersall, I (Eds) Handbook of paleoanthropology. Vol.2. Primate evolution and human origin. *Springer-Verlag: Heidelberg*

Byrne, R. W., Russon, A. (1998). Learning by imitation: a hierarchical approach. Behavioral and Brain Sciences, 21, 667-721.

Bullock, D., Grossberg, S. (1988). Neural dynamics of planned arm movements: emergent invariants and speed-accuracy properties. Psychol Rev, 95, 49–90.

Behrmann, M. (2000). The mind's eye mapped onto the brain's matter. Current Psychological Science, 9, 2, 50-54.

Brooks, R.A. (1986). A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation, 2(1), 14-23.

Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., & Rosen, D. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks, 5*, 698–713.

Chappell, J., Kacelnik, J. (2002). Selection of tool diameter by New Caledonian crows Corvus moneduloides Animal Cognition 7:121-127.

Clark, A. (1997). Being There: Putting Brain, Body and World Together Again, Cambridge, Massachusetts: MIT Press.

Churchland, P.S. (1986). Neurophilosophy: An unified science of Mind Brain, Cambridge MA: MIT Press.

Churchland, P.S. (2002). Self-representation in nervous systems. Science; 296: 308-310.

Cohen, M.S. (1996). Changes in cortical activity during mental rotation. A mapping study using functional MRI. *Brain* 119, 89–100.

Chater, N. (1996). Reconciling simplicity and likelihood principles in perceptual organization. Psychol. Rev. 103, 566–581.

Damasio, A.R. (2000). The feeling of what happens: Body, emotion and the making of consciousness. NY: Vintage.

Dayan, P., Abbott, L. (2001). *Theoretical neuroscience*. Cambridge, MA:MIT Press.

Desmurget, M., Grafton, S. (2000). Forward modelling allows feedback control for fast reaching movements. Trends in Cog Sci 4:423-431.

Diekamp, B., Kalt, T., Gunturkun, O. (2002). Working Memory Neurons in Pigeons. Journal of Neuroscience 22(RC210) 1-5.

Dracopoulos, D.C. (1997). Evolutionary Learning Algorithms for Neural Adaptive Control. London: Springer.

Edelman, G.M. (2006). Second Nature: Brain Science and Human Knowledge. Yale University Press.

Edelman, G.M., Tononi,.G. (2001). A universe of consciousness: How matter becomes imagination. Basic Books.

Emery, N.J. & Clayton, N.S. (2004). The Mentality of Crows: Convergent Evolution of Intelligence in Corvids and Apes. Science 306: 1903-1907.

Erlhagen,W., & Sch¨oner, G. (2002). Dynamic field theory of movement preparation. *Psychological Review, 109*, 545–572.

Feldman, J., Narayanan, S. (2004). Embodied meaning in a neural theory of language. Brain and Language, 89, 385–392.

Feldman.J.(2006). From molecule to metaphor: A neural theory of language, MIT Press.

Festinger, L. (1957). A theory of cognitive dissonance. Evanston, IL: Row, Peterson.

Flash, T., Hogan, N. (1985). The coordination of arm movements: an experimentally confirmed mathematical model. J Neurosci, 5, 688-703.

French, R. M. (1995). The Subtlety of Sameness, Cambridge, MA: The MIT Press.

French, R.M (2006). The dynamics of the computational modelling of analogy-making, CRC Handbook of Dynamic Systems Modelling, Paul Fishwick (ed.), Boca Raton, FL: CRC Press LLC.

Fritzke, B. (1995). A growing neural gas network learns topologies. In G. Tesauro, D. Touretzky, & T. Leen (Eds.), *Advances in neural information processing systems, 7*(pp. 625–632). Cambridge, MA: MIT Press.

Gersho, A., & Gray, R. (1991). *Vector quantization and signal compression*. Boston:Kluwer.

Fuster, J.M. (2003). Cortex and Mind: Unifying Cognition. Oxford university press.

Fragaszy, D.M., (1998). How non-human primates use their hands, in *The Psychobiology of the Hand*, ed. K.J. Connolly. Cambridge: Mac Keith Press, 77–96.

Gallese,V. Lakoff,G. (2005). The brain's concepts: The role of the sensory-motor system in reason and language, Cognitive Neuropsychology 22, pp. 455–479

Geffner,H. (1992). Default Reasoning: Causal and Conditional Theories. MIT Press.

Georges-Francois, P., Rolls, E.T., Robertson, R.G. (1999). Spatial view cells in the primate hippocampus: allocentric view not head direction or eye position or place. *Cerebral Cortex* Volume 9(3) pages 197-212.

Georgeff, M. P., Lnsky, A. L. (1987). Reactive reasoning and planning. In Proceedings of the 6th National Conference on Artificial Intelligence (AAAI-87), Seattle, WA, pp.677-682.

Georgeff, M. P. (1999). The belief-desire-intention model of agency. In Intelligent Agents, V (eds J. P. Müller, M. P. Smith and A. S. Rao) LNAI Volume 1555, pp.1-10. Springer, Berlin.

Gibson, J.J. (1979). The Ecological Approach to Visual Perception, Houghton Mifflin, Boston.

Glenberg, A.M. (1997). What memory is for. Behaviour and Brain Sciences, 20, 1–19.

GNOSYS project documentation: www.ics.forth.gr/gnosys

Gribble, P. L , Ostry, D. J. , Sanguineti, V. and Laboissière, R. (1998). Are Complex Computations Required for the Control of Arm Movement? Journal of Neurophysiology , 79(3):1409-1424.

Grush, R. (1995). Emulation and Cognition, Doctoral Dissertation, University of California, San Diego.

Grush, R. (2004). The emulation theory of representation: Motor control, imagery, and perception. Behavioral and Brain Sciences, 27, 377–396.

Gunturkun, O. (2005). Avian and Mammals Prefrontal Cortices. Brain Research Bulletin 66:311-316.

Hafting. T., Fyhn, M., Molden, S., Moser, M.B., Moser, E.I. (2005). "Microstructure of a spatial map in the entorhinal cortex. *Nature* Volume 436(7052), pages 801-806

Hatze,H. (1988). High-precision 3D photogrammetric calibration and object space reconstruction using a modified DLT-approach. *J. Biomech* 21, 533-53.

Hesslow, G. (2002). Conscious thought as a simulation of Behavior and Perception, Trends in Cognitive Sciences, Vol 6.

Hesslow, G., Jirenhed, D.A, (2007). The Inner World of a Simple Robot. *Journal of Consciousness Studies*, 14:85-96

Hersch, M., Billard, A.G. (2008). Reaching with multi-referential dynamical systems. Auton Robot 25, 71–83.

Hirose, M., Ogawa, K. (2007). Honda humanoid robots development. Philos Transact A Math Phys Eng Sci, 365,11-9.

Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci U S A, 79, 2554-8.

Hoffmann, H., Moller, R. (2004). Action selection and mental transformation based on a chain of forward models. In: Proceedings of the eighth international conference on the simulation of behavior (pp. 213–222). SAB 04.

Hofstadter, D. R. (1984). The Copycat project: An experiment in nondeterminism and creative Reasoning in Intelligent Systems. Morgan Kaufmann, 1988.

Hofstadter, D.R. (2007). I am a strange loop. NY: Basic Books.

Hofstadter, D.R. (1979).Gödel,Escher,Bach: an Eternal Golden Braid, NY: Basic Books.

Hofstadter, D. R. & the Fluid Analogies Research Group (1995). Fluid Concepts and Creative Analogies. New York: Basic Books.

Holland, O., Goodman,R. (2003). Robots with internal models: a route to machine consciousness? Journal of Consciousness Studies, Special Issue on Machine Consciousness, vol 10, No 4.

Hunt, G.R.,Gray, R.D. (2002). Species-wide manufacture of stick-type tools by New Caledonian Crows. *Emu* 102: 349-353.

Imamizu, N. (2000). Human cereballar activity reflecting an acquired internal model of a new tool. Nature 403:192-196.

James, W. (1890). The Principles of Psychology, 2 vols. Dover Publications 1950, vol. 1: ISBN 0-486-20381-6, vol. 2: ISBN 0-486-20382-4.

Jordan, M., Jacobs, R. (1992). Hierarchies of adaptive experts. In J. Moody, S. Hanson, & R. Lippmann (Eds.), *Advances in neural information processing systems, 4* (pp. 985–993). San Mateo, CA:Morgan Kaufmann.

Jordan, M.L., Rumelhart, D.E. (1992). Forward Models: Supervised learning with a Distal teacher, Cognitive Science, 16, 307-354.

Jordan M.I. (1986). Attractor dynamics and parallelism in a connectionist sequential machine, in Proc. Eighth Ann Conf Cognitive Science Society, 531-546, Hillsdale, NJ: Erlbaum.

Jordan M.L., Wolpert, D.M. (1999). Computational motor control, In: M. Gazzaniga, (Ed.). The Cognitive Neurosciences. Cambridge, MA: MIT Press.

Kasderidis, S.,Taylor, J. G. (2004). Attentional Agents and Robot Control, *International Journal of Knowledge-based and Intelligent Systems* 8, 69-89.

Kasderidis,S. (2006). A Computational Model of Multiple Goals. Lecture Notes in Computer Science (LNCS), Proceedings of 2006 International Conference on Artificial Neural Networks (ICANN), Athens, Greece.

Kawato, M.(1999). Internal models for motor control and trajectory planning. Current Opinion in Biology 9:718-727.

Kenward, B., Weir, A. A. S., Rutz, C., Kacelnik, A. (2005). Tool manufacture by naive juvenile crows. *Nature* 433: 121.

Kohonen, T. (1995). *Self-organizing maps*. Berlin: Springer.

Kokinov, B.N., Petrov, A. (2001). Integration of Memory and Reasoning in Analogy-Making: The AMBR Model, The Analogical Mind: Perspectives from Cognitive Science, Cambridge, MA: MIT Press.

Kosslyn, S.M. (1993). Visual mental imagery activates topographically organized visual cortex: pet investigations. *J. Cogn. Neurosci.* 5, 263–287.

Kosslyn, S. M., Thompson, W. L., Ganis, G. (2006). *The case for mental imagery*. New York, NY: Oxford University Press.

Kanwisher, N. (1997). The fusiform face area: A module in human extrastriate cortex specialized for face perception. *J. Neurosci.*17, 4302–4311.

Krauss, S. (1999). Simplifying Bayesian inference: the general case. In Model-based Reasoning in Scientific Discovery (Magnani, L. et al., eds), pp. 165–179, Kluwer Academic/Plenum Press.

Krichmar, J. L., Nitz, D. A., Gally, J. A., and Edelman, G. M. (2005). Characterizing functional hippocampal pathways in a brain-based device as it solves a spatial memory task. Proceedings of the National Academy of Science USA 102, 2111-2116.

Krichmar, J. L., Seth, A. K., Nitz, D. A., Fleischer, J. G., and Edelman, G. M. (2005). Spatial navigation and causal analysis in a brain-based device modeling cortical-hippocampal interactions. Neuroinformatics 3, 197-222.

Lakoff,G., Núñez,R. (2000). Where Mathematics Comes From: How the Embodied Mind Brings Mathematics into Being. New York: Basic Books.

Layard, E.L., & Layard, E.L.C. (1882). Notes on the avifauna of New Caledonia. *Ibis* 6: 520-522.

Limongelli, L., Boysen, S.T., Visalberghi, E. (1995). Comprehension of cause-effect relations in a tool-using task by chimpanzees (*Pan troglodytes*). *Journal of Comparative Psychology* 109: 18-26.

Lowe, D.G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60: pages 91-110.

Lungarella,M., Metta,G., Pfeifer,R., Sandini,G. (2003). Developmental Robotics: A Survey. Connection Science. 15(4), pp. 151-190.

Le Bihan, D. (1993). Activation of human primary visual cortex during visual recall: A magnetic resonance imaging study. *Proc. Natl. Acad. Sci. U. S. A.* 90, 11802–11805.

Liegeosis, A. (1977). Automatic Supervisory Control of the Configuration and Behavior ofMultibodyMechanisms. IEEE Trans Systems, Man, and Cybernetics, 7, 868–871.

Loncomilla,P., Ruiz del Solar,J. (2005). "Improving SIFT-Based Object Recognition for Robot Applications." In ICIAP, pages 1084-1092.

McGrew, W.C. (1992). Fanzee Material Culture. Cambridge: Cambridge University Press.

Metzinger, T. , Gallese, V. (2003). Motor Ontology: The representational reality of Goals, actions and selves, Philosophical Psychology,Vol 16.

Metta, G. (2000). Babybot a study on sensorimotor development. PhD thesis, University of Genoa, Genoa.

Metta, G., Fitzpatrick, P., Natale, L. (2006). YARP: Yet Another Robot Platform. International Journal of Advanced Robotics Systems, 3, 43-48.

Mikolajczyk, K., Schmid, C. (2005). A Performance Evaluation of Local Descriptors." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10): pages 1615-1630.

Miall, R.C., Wolpert, D.M. (1996). Forward Models for Physiological Control. Neural Networks 9:1265-1270.

Minsky, M., Singh, P. (2004). An Architecture for Cognitive Diversity, in Visions of Mind, Darryl Davis Ed.

Minsky, M. (2006). The Emotion Machine, Simon & Schuster Adult Publishing Group.

Morasso, P. (1981). Spatial control of arm movements. Experimental Brain Research, 42, 223-227.

Morasso, P., V. Sanguineti, editors (1997). Self-Organization, Computational Maps and Motor Control. Elsevier, Amsterdam.

Morasso, P. (2006c). Consciousness as the emergent property of the interaction between brain body and environment: the crucial role of haptic perception, Artificial Consciousness, Imprint Academic.

Mussa Ivaldi, F.A, Morasso, P., Zaccaria, R. (1988). Kinematic Networks. A Distributed Model for Representing and Regularizing Motor Redundancy. Biological Cybernetics, 60, 1-16.

Mohan, V., Morasso, P. (2006). A forward/inverse motor controller for Cognitive Robotics, Artificial Neural Networks - ICANN 2006, Lecture Notes in Computer Science, vol 4131/2006, p. 602-611, Springer, Berlin/Heidelberg.

Mohan. V, Morasso, P. (2007a). 'Neural Network of a Cognitive Crow': An Interacting map based architecture, Proceedings of IEEE International Conference on Self Organizing and Self Adaptive Systems, MIT Boston, Mass., USA.

Mohan. V., Morasso, P. (2007b). Towards reasoning and coordinating action in the mental space. International Journal of Neural Systems, 17, 4,1-13.

Mohan, V., Morasso, P., Taylor, J., Kasderidis, S. (2007c). Reasoning system. GNOSYS Deliverable 15, http://www.ics.forth.gr/gnosys/.

Mohan, V., Morasso, P., Taylor, J., Kasderidis, S. (2007d). Action generation system. GNOSYS Deliverable 16, , http://www.ics.forth.gr/gnosys/.

Mohan, V., Morasso, P. (2008). 'Reaching Extended': Unified computational substrate for mental simulation and action execution in Cognitive Robots. Proceedings of third International conference of Cognitive science, CogSci 2008, Moscow, Russia (June 2008).

Natale, L., Orabona, F., Metta, G., Sandini, G. (2007). Sensorimotor coordination in a "baby" robot: learning about objects through grasping. Prog Brain Res, 164,403-24.

Newell, A., Simon, H. (1976). Computer Science as Empirical Enquiry: Symbols and Search, Communications of ACM, 19, 113-126.

Nishiwaki, K., Kuffner, J., Kagami, S., Inaba, M., Inoue, H. (2007). The experimental humanoid robot H7: a research platform for autonomous behaviour. Philos Transact A Math Phys Eng Sci, 365,79-107.

Op de Beeck, H., Haushofer, J., Kanwisher, N. (2008). Interpreting fMRI data: maps, modules, and dimensions. Nature Reviews Neuroscience.

Oppenheim,A., Schafer,R., Buck,J.R.(1999). Discrete time signal processing. ISBN: 9780137549207.

O'Craven, K.M., Kanwisher, N. (2000). Mental imagery of faces and places activates corresponding stimulus-specific brain regions.*J. Cogn. Neurosci.* 12, 1013–1023.

O'Keefe, J., Dostrovsky. J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain Research* Volume 34, pages 171-175.

OReilly, R.C. (2006). Biologically Based Computational Models of High-Level Cognition. Science, 314, 91-94.

Oztop, E., Wolpert, D., Kawato, M. (2004). Mental state inference using visual control parameters. Cog Brain Research.

Parsons L.M., Sergent, J., Hodges, D. A. and Fox, P. T. (2005). Cerebrally-lateralized mental representations of hand shape and movement. . *Journal of Neuroscience 18*, 6539 - 6548.

Pearl, J. (1988). Probabilistic analogies. AI Memo No. 755, Massachusetts Institute of Technology, Cambridge, MA.

Pearl, J. (1998). Graphs, Causality, and Structural Equation Models, UCLA Cognitive Systems Laboratory, Technical Report R-253.

Phillips, C.L., Harbor, R.D. (2000). Feedback Control Systems. Upper Saddle River, NJ: Prentice Hall.

Pulvermuller, F. (2003). *The Neuroscience Of Language: On Brain Circuits Of Words and Serial Order*, Cambridge University Press: Cambridge

Ridderinkohf, K.R., Ullsperger, M., Crone, E.A., Sander, N. (2004). The Role of the Medial Frontal Cortex in Cognitive Control , Science, Vol 306.

Rizzolatti, G., Fogassi, L., Gallese, V. (2001). Neurophysiological mechanisms underlying the understanding and imitation of action. Nature Reviews Neuroscience, 2: 661-670, 2001.

Rougier, N.P., Noelle, D.C., Braver, T.S., Cohen, J.D., O'Reilly, R.C. (2005). Prefrontal cortex and the flexibility of cognitive control: Rules without symbols. Proceedings of the National Academy of Sciences USA, 102, 7338-7343.

Savage-Rumbaugh, E.S., Rumbaugh, D.M., Fields, W.M. (2006). Language as a window on rationality. Chapter 23, pp 513-552 in Hurley S & Nudds M (eds) (2006) Rationality in Animals? Oxford: Oxford University Press.

Sejnowski, T.J., Koch, C., Churchland, P.S. (1988). Computational neuroscience. Science, 9, 2411299-306.

Serre,T., Wolf, L., Poggio, T. (2005). "Object Recognition with Features Inspired by Visual Cortex." In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*,2, pages 994-1000.

Šoch, M., Lórencz, R. (2005). Solving Inverse Kinematics – A New Approach to the Extended Jacobian Technique. Acta Polytechnica, 45, 21-26.

Shadmehr, R. (1999). Evidence for a forward dynamic model : human adaptive motor cotnrol. NIPS:11 :3-9.

Shadmehr, R., Mussa-Ivaldi, F.A. (1994). Adaptive representation of dynamics during learning of a motor task. J Neurosci, 14,3208–3224.

Shadmehr, R., Krakauer, J.W. (2008). A computational neuroanatomy for motor control, Experimental Brain Research, 185, 359–381.

Shapiro, R.(1978). Direct linear transformation method for three-dimensional cinematography. *Res. Quart.* 49, 197-205.

Shanahan, M. (2006). A cognitive architecture that combines internal simulation with a global workspace, Consciousness and Cognition Vol.15, 433–449.

Shanahan, M. P. (2005). Perception as abduction: Turning sensor data into meaningful representation. Cognitive Science, 29, 109–140.

Sch¨oner, G., & Dose, M. (1992). A dynamical system approach to task level system integration used to plan and control autonomous vehicle motion. *Robotics and Autonomous Systems, 10*, 253–267.

Sch¨oner, G., Dose,M., & Engels, C. (1995). Dynamics of behavior: Theory and applications For autonomous robot architectures. *Robotics and Autonomous Systems, 16*, 213–245.

Sloman, A. (2000). Architectural requirements for human-like agents both natural and artificial. In Dautenhahn, K., editor, Human Cognition And Social Agent Technology, Advances in Consciousness Research, pages 163–195. John Benjamins, Amsterdam.

Smolensky, P., Legendre, G. (2006). The Harmonic Mind: From Neural Computation to Optimality-Theoretic Grammar, Vol. 1 and II. MIT Press.

Sun, R. (2000). Symbol grounding: A new look at an old idea Philosophical Psychology, Vol.13, No.2, pp.149-172.

Sun, R. (2002). Duality of the Mind. Lawrence Erlbaum Associates.

Sun, R. (2004). Desiderata for cognitive architectures Philosophical Psychology, Vol 17, No.3.

Sun, R. (2007). The importance of cognitive architectures: An analysis based on CLARION. Journal of Experimental and Theoretical Artificial Intelligence, Vol.19, No.2, pp.159-193.

Sutton, R., Barto, A. (1998). Reinforcement learning. Cambridge, MA: MIT Press.

Taylor, J. G. (2000). Attentional Movement: the control basis for consciousness. Neuroscience Abstracts 26 (Part 2) 839.3, 2231.

Taylor, J.G., Kasderidis, S., Trahanias,P., Hartley,M. (2006). A basis for Cognitive machines. Proceedings of International Conference on Artificial Neural Networks, Athens, Greece.

Taylor, .J.G (2003). The CODAM Model and Deficits of Consciousness (2003) Lecture Notes in Computer Science, 2774/2003, Springer Berlin / Heidelberg.

Tcheang, L., Bays, P.M., Ingram, J.N., Wolpert, D.M. (2007). Simultaneous bimanual dynamics are learned without interference. Experimental Brain Research, 183,17-25.

Tikhanoff, V., Cangelosi, A., Fitzpatrick, P., Metta, G., Natale, L., Nori, F. (2008). *An Open-Source Simulator for Cognitive Robotics Research. Cogprints, article 6238.*

Tomasello, M., Call, J. (2006). Do chimpanzees know what others see - or only what they are looking at? Chapter 17, pp 371-384 in Hurley S & Nudds M (eds) Rationality in Animals? Oxford: Oxford University Press.

Tani, J., Nolfi, S. (1999). Learning to perceive the world as articulated: an approach for hierarchical learning in sensory-motor systems", in From animals to animates 5, R. Pfeifer, B. Blumberg, J. Meyer, and S. Wilson, Eds. Cambridge, MA: MIT Press., 1998, later published in Neural Networks, vol12, pp1131.

Tani, J., Yokoya,R., Ogata,K..,Komatani, Okuno,H.G. (2007). "Experience-based imitation using RNNPB", Advanced Robotics, Vol.21, No.12, pp.1351-1367.

Toussaint, M. (2004). Learning a world model and planning with a self-organizing dynamic neural system . Advances in Neural Information Processing Systems 16 (NIPS 2003), 929-936, MIT Press, Cambridge.

Toussaint, M. (2006). A sensorimotor map: Modulating Lateral connections for Anticipation and Planning, Neural Computation 18, 1132-1155.

Tsuji, T., Morasso, P., Shigehashi, K., Kaneko, M. (1995). Motion Planning for Manipulators using Artificial Potential Field Approach that can Adjust Convergence Time of Generated Arm Trajectory. Journal Robotics Society of Japan, 13, 285-290.

Tulving, E. (2000). Introduction to Memory. In M.S. Gazzaniga (Ed.), The New Cognitive Neurosciences, 2nd Ed. (pp. 727-732). Cambridge, MA: MIT Press.

Umiltà, M.A., Escola, L., Intskirveli, I., Grammont, F., Rochat, M., Caruana, F., Jezzini, A., Gallese, V., Rizzolatti, G. (2008). When pliers become fingers in the monkey motor system. Proc Natl Acad Sci U S A.105(6):2209-13.

Uno, Y., Kawato, M., Suzuki, R. (1989). Formation and control of optimal trajectory in human multijoint arm movement: minimum torque-change model. Biol Cybern, 61, 89-101.

Varela, F. J. , Maturana, H.R. & Uribe, R. (1974). Autopoiesis: the organization of living systems, its characterization and a model. Biosystems 5 187–196.

Visalberghi, E., Fragaszy, D. (2006). What is challenging about tool use? The capuchin's perspective. In Wasserman, E.A. & Zentall, T.R. (Eds.), *Comparative Cognition: Experimental Explorations of Animal Intelligence* (pp. 529-552). New York: Oxford University Press

Visalberghi, E. (1993). Capuchin monkeys: a window into tool use activities by apes and humans. In K. Gibson & T. Ingold (Eds.), *Tool, Language and Cognition in Human Evolution* (pp. 138-150). Cambridge: Cambridge University Press.

Visalberghi, E., Limongelli, L. (1996). Action and understanding: tool use revisited through the mind of capuchin monkeys. In A. Russon, K. Bard, & Parker, S. (Eds.), *Reaching into thought. The minds of the great apes* (pp. 57-79). Cambridge: Cambridge University Press.

Visalberghi, E. (2000). Tool use behaviour and the understanding of causality in primates. In E. Thommen & H. Kilcher (Eds.), *Comparer ou prédire: examples de recherches en psychologie comparative aujourd'hui* (pp. 17-35). Fribourg (Switzerland): Les Editions Universitaires.

Visalberghi, E., Tomasello, M. (1997). Primate causal understanding in the physical and in the social domains. *Behavioral Processes, 42,* 189-203.

von der Malsburg, C. (1973). Self-organization of orientation-sensitive cells in the striate cortex. *Kybernetik, 15*, 85–100.

Wiemer, J. (2003). The time-organized map algorithm: Extending the self-organizing map to spatiotemporal signals. *Neural Computation, 15*, 1143–1171.

Willshaw, D. J., Von der Malsburg, C. (1976). How patterned neural connections can be set up by self organization. *Proceedings of the Royal Society of London, B194*, 431–445.

Watkins, C. (1989). Learning through delayed rewards, PhD Thesis, Cambridge University, Cambridge, UK.

Weir, A.A.S., Chappell, J., Kacelnik, A. (2002). Shaping of Hooks in New Caledonian Crows. Science 297:981-3.

Kacelnik, A., Chappell, J., Weir, A. A. S., Kenward, B. (2004). Tool use and manufacture in birds. In: *Encyclopedia of animal behavior* (ed. Bekoff, M.), pp. 1067-1069, Volume 3. Greenwood Publishing Group, Westport, CT, US.

White, D.A., Sofge, D.A (1992). Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches. New York: Van Nostrand.

Whitney, D. E. (1969). Resolved Motion Rate Control of Manipulators and Human Prosthesis. IEEE Transactions on Man-Machine Systems, MMS-10, 47–53.

Williams, J.D., Rippon, G. (1995). Psychophysiological correlates of dynamic imagery. *Br. J. Psychol.* 86, 283–300.

Wiener,N. (1948). Cybernetics: Or the Control and Communication in the Animal and the Machine. MIT Press.

WP9 Work Programme 9 (Draft Version released by the European commission on 01-07-08). Theme for research and development under the specific programme “Cooperation” implementing the Seventh Framework Programme (2007-2013) of the European Community for research, technological development and demonstration activities.

Wolpert, D.M., Ghahrmani, Z.,Jordan, M.I. (1994). An Internal Model for Integration. Science 269:1880-1882.

Wolpert, D. M. & Kawato, M. (1998). Multiple paired forward and inverse models for motor control. Neural Networks 11, 1317–1329.

Wolpert, D.M., Ghahramani, Z. (2000). Nature Neuroscience , Computational Principles of movement neuroscience, 1212–1217.

Yuille, A., Carter, N., Tenenbaum, J.B. (2006). Probabilistic models of cognition: Conceptual foundations, Trends in Cognitive Sciences Vol.10 No.7.

Zak, M. (1988). Terminal attractors for addressable memory in neural networks. Phys. Lett. A, 133, 218–222.

Zatorre, R. J., Halpern, A. R. (1993). Effect of unilateral temporal lobe excision on perception and imagery of songs. *Neuropsychologia* 31, 221–232.

Zatorre, R.J., Chen, J.L., Penhune, V.B. (2007). When the brain plays music. Auditory-motor interactions in music perception and production. Nature Reviews Neuroscience, 8, 547-558

Table of Illustrations

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6