Teaching iCub to draw ‘Shape’

Vishwanathan Mohan$^1$, Jacopo Zenzeri, Pietro Morasso$^1$, Giorgio Metta$^1$, VS Chakravarthy$^2$, and Giulio Sandini$^1$

$^1$Robotics, Brain and cognitive science department, 
Italian Institute of technology 
Via Morego 30, Genova, 
ITALY
$^2$Department of Biotechnology, 
Indian Institute of technology, Chennai

(vishwanathan.mohan, pietro.morasso, giorgio.metta, giulio.sandini) @ iit.it
(schakra@iitm.ernet.in)
Abstract: The core cognitive ability to perceive and synthesize ‘shape’ underlies all our basic interactions with the world, be it shaping one’s fingers to grasp a ball or shaping one’s body while imitating a dance. In this article, we describe our attempts to understand this multidimensional problem by creating a primitive shape perception/synthesis system for the baby humanoid iCub. We specifically deal with the scenario of iCub gradually learning to draw or scribble shapes on a drawing board after observing a demonstration by a teacher and aided by a series of self evaluations of its performance. Learning to imitate a demonstrated human movement (specifically, visually observed end effector trajectories of a teacher) can be considered as a special case of the proposed computational machinery. Concealed behind the playful scenario of teaching iCub to draw, solutions to the complex set of mechanisms that transform a teachers demonstration to the motor commands of the student are addressed, mainly a) Characterizing the ‘shape’ of a demonstrated trajectory using a set of critical points derived using Catastrophe theory (CT, Thom ); b) Reconstructing the AVP into a concrete motor goal (CMG) in iCub’s egocentric space; c) learning to synthesize a continuous virtual trajectory similar to the demonstrated shape using the discrete set of critical points defined in CMG; d) coupling the synthesized virtual trajectory to the appropriate internal body model of iCub to generate the motor action i.e. the virtual trajectory acts as an attractor to the body generating the motor action; e) analysis of the self generated movement (forward model output) using CT to form an abstract motor program (AMP); f) comparison of abstract visual and motor information (AVP and AMP) to generate a score of performance and hence closing the learning loop. Issues related to modularity, compositionality and generalization while drawing/imitating more complex trajectories, end effector independent action representations, motor learning, motor imagery, the perception and synthesis of ‘Shape’ in general are discussed in the context of the proposed architecture.

Keywords: Shape, Shaping, Catastrophe theory, Passive Motion Paradigm, Terminal attractors, iCub
1. Introduction

A monkey trying to cling to a branch of a tree, a couple dancing, a woman embracing her baby or a baby humanoid trying to grasp a ball are all essentially attempting to shape their bodies to conform to the shape of the worlds with which they are interacting. Simply stating, behind all our incessant perception-actions underlies the core cognitive faculty of ‘perceiving and synthesizing’ shape. Perceiving affordances of objects in the environment for example a cylinder, a ball, an amoeba etc, or performing movements ourselves like reaching, moving, pointing in specific trajectories, coordinating ones fingers while manipulating objects, reading, writing, drawing or imitating are some examples. Surprisingly, it is not easy to give a precise mathematical or quantitative definition of ‘shape’ or even express it in mensurational quantities like length, angles or topology. Vaguely, shape is the core information in any object/action that survives the effects of changes in location, scale, orientation, end effectors/bodies used in its creation, noise, and even minor structural injury. It is this informational invariance that makes ‘shape’ the seed for any high level sensorimotor interaction. Hence, an unified treatment to the dual operations of shape perception and synthesis is critical both from the intrinsic viewpoint of better understanding our own perceptions and actions, to creating autonomous robots that can flexibly aid us in our needs and in the environments we inhabit and create.

In this article, we describe our attempts to understand this problem by creating a primitive shape perception/synthesis system for the baby humanoid iCub. The background scenario using which we develop our results is of iCub gradually learning to write or scribble shapes on a drawing board after observing a demonstration by a teacher and aided by a series of self evaluations of its performance. Learning to imitate a demonstrated human movement can be seen as a special case of this system, writing task of course imposing additional constraints of making smooth planar trajectories on the drawing board with a manipulated tool (in this case the paint brush) coupled to the arm. It is easy to visualize that the scenario of iCub observing a shape demonstrated by a teacher and then learning to draw that shape on a drawing board encompasses the complete perception-action loop. Sub problems like visual perception of shape, 3D reconstruction of the observed demonstration to one’s own egocentric space, creation of motor goals, trajectory formation, inverse kinematics and resolution of redundant degrees of freedom that are unconstrained by observation (especially in 53 DoF iCub), learning through self evaluations, generalization and compositionality of acquired knowledge, creation of compact sensory-motor representations, internal models, modularity and motor equivalence, tool use, are active fields of research in themselves. Even attempting to untangle a problem that brings all of them together is undeniably daunting. Hence, throughout this article, we employ a divide and rule strategy at the same time trying to maintain a fine balance between local behavior of specialized modules and the global behavior that emerges out of their mutual interactions. Central ideas are presented both intuitively and using formal analysis, supported by experiments on iCub.
1.1 Describing ‘Shape’

Several schools of psychology have endeavored to understand mechanisms behind visual perception, specifically shape and form description. Seminal works of Hebb (Hebb, 1948), Gibson (Gibson, 1960) and Marr (Marr, 1982) have specifically been influential in the field. Marr described vision as proceeding from a two-dimensional visual array to a three-dimensional description of the world in different stages, mainly a preprocessing and feature extraction (including edges, regions etc) phase, followed by 2.5d sketch where textures, shades are acknowledged and finally a continuous 3D map. Building on his work several approaches to characterize shape like shape from shading (Horn, 1990), shape from contour (Ulupinar et al, 1990), shape from stereo (Hoff et al, 1989) and shape from fractal geometry (Chen et al, 1990) were proposed. In general, shape analysis methods can be classified based on those which parse the shape boundary (external) as opposed to those which use the interior (Global feature based methods). Examples of the former class are algorithms which parse the shape boundary (Duda et al, 1973) and various Fourier transforms of the boundary can be found in the works of Wallace (1980). Examples of global methods include the medial axis transform (Blum, 1967), moment based approaches (Belkasim et al, 1991), and methods of shape decomposition into other primitive shapes (Bruckstein et al 1992). Research in 3D object recognition studied extensively by the computer vision community has also resulted in several approaches to characterize shape of 3D volumes based on aspect graphs (Koenderink et al, 1979), extended Gaussian images (Horn et al, 1984), superquadrics (Solina et al, 1990), spin images (Johnson et al, 1999), and geometric hashing (Lamdam et al, 1988) among others. Interested reader may refer to (Iyer et al, 2005, Loncaric, 1998) for a comprehensive survey of the pros and cons of the various shape analysis approaches. The search for points that mark significant shape related events on a curve also led to the invention of a variety of special points like anchorage points (Anquetil and Lorette, 1997), salient points (Fischler and Wolf, 1994), dominant points (The and Chin, 1989), singular points (Rocha and Pavlidis,1994) etc. Such special points have been used for curve partitioning, 2D and 3D object representation and recognition, for scene segmentation and compression, object motion tracking etc. Handwritten character recognition research employs features like corners (Mehrotra et al., 1990), line-crossings (X), T-junctions, high curvature points and other shape primitives (Fischler and Wolf, 1994; Li and Yeung, 1997; Sanfeliu and Fu, 1983; Teh and Chin, 1989) to describe shape. But often the choice of these features is ad hoc and domain/script dependent. And one is never certain if a given set of features is minimal or redundant. Further, in addition to just detecting shape, our objective to also teach iCub to draw the shape after observing a demonstration required us to look for a fixed set of script independent (possibly universal) primitives that both gives rise to a concise higher level representation and be exploited to drive action generation in iCub. However, for such a program to succeed one must obtain a fundamental insight into the shape of line diagrams in general, of which demonstrated movements by a teacher, hand written characters are all significant instances.
A systematic treatment of the problem of shape can be found in a branch of mathematics known as Catastrophe theory (CT) originally proposed during the late 1960’s by the French mathematician Rene Thom to formally explain the origin of forms and shapes in nature, further developed by Zeeman, Gilmore among others and applied to a range of problems in engineering and physics. According to CT, the global shape of a smooth function \( f(x) \) can be fully characterized by a set of special local features (like peaks, valleys etc) called critical points (CP), where first and probably some higher order derivatives vanish. The set of critical points, small and finite in nature can be formally classified using four measures 1) Stability 2) Codimension 3) Valency and 4) Compositionality. When all the CP’s and their type is known, we know the global shape and also its complexity and stability. Thom believed that the stability of form in biological systems (and many other phenomena as well) can be accounted for in terms of the catastrophe model. The profound results of CT have then been used to explain a range of phenomenon in nature like the breaking of a swelling wave, the streaks of light seen in a coffee cup (known as ‘light caustics’), the splash of a drop on a liquid surface (Thom, 1975; Gilmore, 1981), classification of shapes of 3D volumes (Koenderink and van Doorn, 1986), distinguish phantom edges from real ones in a scale-based approach to edge extraction (Clark, 1988) and a variety of other problems in engineering and physics (Poston and Stewart, 1978). In a previous work, (Chakravarthy et al, 2005) extended CT and systematically derived a small set of 11 shape features to characterize the shape of any line diagram in general. These 11 shape features are found in several of the world’s scripts and may in fact be universal in nature. More complex shapes as they have shown generally breakdown into weighted combinations of the 11 primitives. In this paper we exploit this framework in two different ways: 1) to form an abstract representation of the shape iCub observes being demonstrated by the teacher, that it has to then learn to create either by reproducing the shape by pointing with its index finger or drawing the shape on the drawing board; 2) to form an abstract representation of self generated movement during its attempt to imitate the demonstration. The former representation is called the abstract visual program (AVP) and the later abstract motor program (AMP), for the obvious reason that the former is observed visually and the latter is a self generated motor action. As we will see later, abstract visual and motor information (AVP and AMP) both created through a CT analysis can be directly compared to close the learning loop that helps iCub self evaluate its performance.

1.2 Creating ‘Shape’

Moving ahead, the inverse operation of shape perception i.e shape synthesis or trajectory formation is one of the basic functions of the neuromotor controller. In particular, reaching, pointing, avoiding, controlling impacts (hitting), generating scribbles, drawing, gesturing, pursuing moving targets, dancing, imitating are different motion paradigms that result in formation of spatial trajectories of different degrees of complexity. Modeling the way in which humans generate these motor actions in daily life is an important scientific topic from many points of view, such as medical, psychological,
kinesiological, and cybernetic. A significant theoretical framework in the field of neural control of movement was inspired by the idea that during movements a huge amount of energy can be stored passively in the biomechanics of the muscle system and controlling the flow of such energy can improve/simplify the motor control task for the brain. This approach called the equilibrium point hypothesis (EPH) was pioneered by Feldman (1966) and Bizzi et al (1976) and later investigated by a large number of research groups. In the context of the motor system an EP (Equilibrium point) is a configuration at which, for each joint, agonist and antagonist torques cancel out. And the problem of trajectory formation can be formulated as the consequence of shifting the EP of the system. A plausible generalization of the EPH concept (in a purely computational sense) is that the goal and multiple constraints (structural, task specific etc) that characterize any given task can be implemented as superimposed force fields that collectively shape an energy function, whose equilibrium point drives the trajectory formation process of the different body parts participating to a motor task. This was the central idea behind the formulation of the Passive Motion Paradigm (PMP, Mussa Ivaldi et al,1988).

Intuitively, 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 force fields). In the recent years, the basic PMP framework has seen a series of significant evolutions. The key advances are: a) Integration with terminal attractor dynamics (Zak, 1988) for controlling the ‘timing’ of the PMP relaxation and to achieve synchronization between motion of different body parts like in bimanual coordination (Tsuji et al 2005, Mohan et al 2009); b) Formulation of branching nodes, for structuring PMP networks in agreement with the body that needs to be coordinated (like humanoids, wheeled platforms etc) and the kinematic constraints of a specific task when external objects/tools are coupled to the body (Mohan et al 2009); c) Development of Forward/Inverse internal model for action generation using PMP and subsequent integration with higher level cognitive layers like reasoning and mental simulation of action (Mohan and Morasso 2007, 2010); d) combining postural and focal synergies during whole body reaching tasks (Morasso et al 2009). In this article we apply the PMP framework in two different ways. The first application is the unconventional one: to create the Virtual Trajectory Generation System (VTGS). We show how the same PMP framework can be used to transform a discrete set of reconstructed critical points (Abstract visual program) into a collection of continuous virtual trajectories (shapes) by pseudorandom titillations in the space of impedances (virtual stiffness that connect the various CP’s to the end effector/tool) and the timing of the different time base generators (i.e Terminal attractors). The second application is the conventional (yet nontrivial) one i.e to coordinate the upper body of iCub (Left arm-waist- Right arm chain) along with the paint brush to derive motor commands for drawing the shapes generated by the VTGS.
1.3 Imitation

Imitation requires a complex set of mechanisms that map an observed movement of a teacher to one's own movement apparatus. The cognitive, social and cultural implications of imitation are well documented today (Rizzolatti and Arbib, 1998, Schaal et al 1999). It is quite evident that scenario of iCub learning to draw a shape after observing a teachers demonstration essentially embeds the core loop of imitation i.e transformation from the visual perception of a teacher to motor commands of a student. In the recent years, a number of interesting computational approaches have been proposed to tackle parts of the imitation learning problem (see Schaal et al 2003 for a detailed review). Direct policy learning methods employ supervised learning to acquire the control policy directly using imitation specific criteria (and without any need to know the task goal of the teacher). Imitation learning is highly simplified in these techniques but often plagued with problems of stability, difficulty in reuse even in slightly modified scenarios and almost no scope of self improvement. In model based learning approaches (Schaal and Atkeson, ), a predictive model of task dynamics is approximated from the demonstration (and not a policy). The idea was that, with the knowledge of the task goal, a task-level policy can be computed with reinforcement learning procedures based on the learnt model. For example, Schaal and Atkeson (Atkeson & Schaal 1997a; Atkeson & Schaal 1997b; Schaal 1997) showed how the model-based approach allowed an anthropomorphic robot arm to learn the task of pole-balancing in just a single trial, and the task of a "pendulum swing-up" in only three to four trials. Another prominent approach to imitation is to learn a policy from demonstrated trajectories. This approach is based on the fact that usually a teacher's demonstration provides a rather limited amount of data, best described as “sample trajectories”. Various projects investigated how a stable policy can be instantiated from such small amount of information. The major advancement in these schemes was that the demonstration is used just as a starting point to further learn the task by self improvement. In most cases, demonstrations were usually recorded using a marker based optical recording equipment and then either spline based techniques or dynamical systems were used to approximate the trajectories. The basic idea was to learn attractor landscapes in phase space for canonical dynamical systems with well defined point attractor properties (Ijspreet et al 2001). The approach has been very effective for movement imitation, and has been successfully applied in different imitation scenarios like learning the kendama game, tennis stokes, drumming, generating movement sequences with an anthropomorphic robot (Billard et al, 2002). Compared to spline based techniques, the dynamical systems based approach have the advantage of being temporally invariant (because splines are explicitly parameterized in time) and naturally resistant to perturbations. The approach proposed in this paper is also based on nonlinear attractor dynamics and has the flavour of self improvement, temporal invariance (through terminal attractor dynamics) and resistance to novel task specific constraints. However, we go beyond these approaches in the sense that what iCub learns to imitate is the ‘Shape’ a rather high level invariant representation extracted from the observed demonstration. It
is independent of scale, location, orientation, time and also the end effector/body chain that creates it (For example, we may draw a circle on a piece of paper or run a circle in a football field). No extra sensory optical marker equipments recording all joint angles of the teacher is employed. Since eyes of the iCub are the only source of gathering formation, a crucial problem of transferring a visually observed motion in teachers ego centric space to one’s own ego centric space is encountered by us. In any case, very use of joint information for motion approximation/generation makes it difficult to generalize the learnt action to a different body chain. Further, as we will see later PMP also provides a well defined mechanism to arrive at a posture similar to the teacher if that is an ‘additional’ goal in the imitation task. Finally, since a small set of primitives shapes can be derived from catastrophe theory, once the primitive actions needed to generate these shapes are learnt, this knowledge can be exploited systematically to imitate/compose more complex actions/shapes. An interesting reference in the context of composing complex shapes from shape primitives comes from a rather distant field of architecture where concepts of shape grammar (Stiny, 2006) and algorithmic aesthetics (Stiny and Gips, 1978) are quite well known.

1.4 Teaching iCub to draw ‘Shape’ – Building blocks, High level information flows

Teaching iCub to write ‘iCub’ is a complex task and requires solutions to several sub problems. In this section we present a concise overview of the core building blocks in the proposed architecture, the problems they solve and the global information flow. A high level block diagram of the proposed architecture is shown in figure 1. We break the scenario starting from the demonstration of a shape to iCub to iCub learning to create it into following sub tasks:

a) **From demonstration to Abstract visual program (AVP):** This task deals with the problem of describing the global shape of a demonstrated trajectory using a set of critical points/features derived using catastrophe theory (CT). In this stage we basically transform a video of the demonstration captured by left and right cameras of iCub, to a concise high level description of its ‘shape’. For example, the essence of a demonstrated shape like ‘C’ in figure 1 is the presence of maxima (or a Bump ‘B’) between two end points (hence shown as E-B-E in the graph). In other words, this stage extracts the essence of the observed demonstration as seen from the image planes of the two cameras (or eyes). We call this information as the Abstract visual program (AVP).

b) **From Abstract visual program to Concrete motor goal (CMG):** AVP basically contains information of the shape critical points (their location and type) as seen by the cameras (i.e in the 2D image plane). The first problem in this task is to transform information in AVP to iCub’s ego centric space counterpart though a process of 3D reconstruction. The problem of transforming visually observed information of the motion in the teachers egocentric space to ones’ own egocentric space is nontrivial and is specially encountered when observation is made only using two eyes (and without use of any external marker based optical recording instruments connected to the teacher to record the demonstration as in almost all computational models in imitation learning today).
this phase is to impose other task specific constraints like the body chain involved (writing with a paint brush, or creating the shape by pointing with index finger), scaling of the shape (if necessary) among others. In sum, this critical phase imposes ‘context’ into the context independent AVP and converts it into a concrete motor goal (CMG), the starting point for iCub to learn to imitate the demonstration.

c) From Concrete motor goal to Virtual trajectories: In the previous two stages we have transformed visual information of a teacher’s movement into a motor goal for the student (iCub). The next task is of the student learning to generate motor actions necessary for imitating the teacher reliably (or achieving the motor goal). This subtask basically deals with the problem of learning to synthesize a continuous virtual trajectory similar to the demonstrated shape using the discrete set of reconstructed critical points in the CMG. Further, by exploring the space of virtual stiffness’ (connecting the different CP’s to the end effector/tool) and the overlap between different time base generators (timing, terminal attractors), different virtual trajectories through the CP’s can be synthesized (figure 1 shows some examples). We finally note that virtual trajectories synthesized in this fashion are not really shapes. Rather, they have a higher cognitive meaning in the sense that they act as an attractor to the internal
body (body+tool) model involved in action generation and play a significant role in deriving the motor commands needed to create the shape itself.

d) *From Virtual trajectories to Motor Action:* In this phase, the synthesized virtual trajectory is coupled to the appropriate internal body (body+tool) model of iCub to derive the motor action i.e trajectories of joint angles to be sent to the actuators to make iCub draw the shape demonstrated by the teacher. The framework of Passive motion paradigm is used in the action generation phase. Intuitively, the computational idea of PMP is analogous to coordination of the movements of a puppet by means of attached strings. Simply, 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. When motor commands obtained by this process of virtual relaxation is actively fed to the robot, the robot will reproduce the same motion. Another key feature in this scheme is that PMP networks naturally form forward/inverse models (). Hence, while the inverse model derives the shape being created as a consequence of the motor commands fed to the actuators. The forward model output is a crucial piece of information that can be directly used by the student to self evaluate his performance in imitating the teacher. No additional visual processing is necessary. The system that generates movement is also used for its recognition. As we will see later, this concept is compatible with the mirror neuron hypothesis in neurobiology. The use of predictive forward models is also confirmed by recent functional imaging experiments during imitation (for example, specular vs anatomical imitation tasks) where the authors showed that the increased activity in superior temporal sulcus (STS) during imitation is due to efferent copies of motor commands that are originating from fronto-parietal mirror areas and are sent back to STS for monitoring purposes.

e) *From motor action to abstract motor program (AMP), closing the learning loop:* The phase deals with the final two questions: What is the matching criteria and in which coordinate frame does it take place? The elegance in our approach is that the system that creates action also recognizes it. In other words, the forward model output created during action generation phase is employed for the monitoring purposes. However there is a twist in the story at this point. Since the observed demonstration of the teacher is available to iCub in its more abstract version (AVP), we must convert the generated action also into a more abstract version, in order for the comparison to be possible (formally). The solution is simple. A CT analysis is done on the forward model output to derive the essence of the generated shape, we call this information as the Abstract Motor Program. The criteria for success is that AVP and AMP should be equivalent or contain the same set of critical points. Otherwise, the relative difference between AVP and AMP can be used to give a score of performance to guide the forthcoming attempts (mental or physical) to draw the shape correctly.

The rest of the paper is organized as follows: in section 2 we describe the application of catastrophe theory to transform a teacher’s demonstration into an abstract visual program (AVP). Section 3 covers a range of topics that finally result in the emergence of shape through action. In section 3.1 we
describe how AVP is transformed into a concrete motor goal (CMG). 3.2 describes the virtual trajectory synthesis system (VTGS). Use of Passive Motion Paradigm (PMP) for action generation (scribbling on the drawing board) presented in section 3.4. Monitoring process by comparing abstract visual and motor information is summarized in 3.5. Generalization in the proposed architecture while drawing more complex shapes and progression of iCub towards scribbling a set of characters (for example, its name ‘iCub’) is presented in 3.6. A discussion concludes.

2. Perceiving the Shape of smooth functions: The catastrophe theory

The starting point is the demonstration of a shape by the teacher to iCub. In all the demonstrations, the teacher creates the shape with a marker (always a green pen). The teacher’s demonstration is usually composed of a sequence of strokes, each stroke tracing a finite, continuous line segment inside the visual workspace of both cameras. The video of the demonstration is captured runtime by the left and right cameras of the iCub and undergoes a pre-processing stage where the location of the pen in each image frame (320 x 240 pixels) is computed using simple colour detection module. If there are N frames in the captured demonstration, the information at the end of the pre-processing phase is organized in the form of a Nx4 matrix, \( \begin{bmatrix} U_{left} & V_{left} & U_{right} & V_{right} \end{bmatrix} \) in the left and right cameras during the N\(^{th}\) frame. Hence, the output of the pre-processing phase is the representation of the demonstrated trajectory in the image planes of the left and right cameras and can be described as:

\[
\begin{align*}
U_{Cam}(t) &= U_{Cam}(t) \\
V_{Cam}(t) &= V_{Cam}(t) \\
t \in [t_0, t_1]
\end{align*}
\]  

where \( U(t) \) is the horizontal coordinate and \( V(t) \) is the vertical coordinate as shown in figure 2, subscript Cam stands for the camera from which recording is made i.e. right or left. Like extracting honey from a honeycomb, we now start the process of extracting the essence from the bulky pre-processed information of the teacher’s demonstration: i.e characterizing its ‘Shape’. The framework of Catastrophe theory (CT) was used for this purpose. According to CT the overall shape of a smooth function, \( f(x) \), is determined by special local features like “peaks”, “valleys” etc. Mathematically, these features are characterized by the points where the first, and probably some higher, derivatives vanish. Such points are known as the critical points (CP) for which there CT provides a formal classification. When all the CPs of a function and their type is known, we know the overall shape of the function.

Further developing the framework of CT, Chakravarthy et al derived a set of 11 critical points (Atoms of shape) and showed that this minimal set is sufficient to characterize the shape of any line diagram in general. The framework also allows to formally speak about the complexity and structural stability of a shape under small perturbations (specially in handwritten scripts, since one of our goals is to teach iCub to write). Since this section is about shapes, from now we let the shapes speak for
themselves. Figure 3 shows four shape features from the set of 11 shape atoms of Chakravarthy et al that we will also use to explain key ideas in CT.

Figure 3 illustrates the definition of Codimension and valence. If a critical point is enclosed by a circle of infinitesimally small radius, the number of lines that intersect it is the valence. As we can visualize, interior, bump and wiggle have valence of 2. End point has valence of 1. As we can also visualize, I, E and B survive small perturbations, however the wiggle is does not. As seen, if the perturbation \( \epsilon \) is negative, wiggle breaks down into two simple critical points (Bump). For a positive perturbation the CP vanishes. Codimension near a shape CP is the number of parameters necessary to bring back the function from its perturbed version to original state. As we can make out, Codimension for I, E and B is zero and W has a codimension of one (i.e \( \epsilon \)).

**Interior Point (I):** This is not really a shape, but as we will see later, it is extremely useful for defining more complex CP’s. ‘I’ is simply any interior point in a stroke. It is stable, has codimension of 0 and valence of 2.

**End Point (E):** This is a terminating point of a stroke, stable with Codimension 0 and Valence of 1.

**Bump (B):** A Bump is an interior point where the derivative of either \( U(t) \) or \( V(t) \) is zero. It is defined as

\[
dU/dt \neq 0, \quad d^2U/dt^2 = 0.
\]
A bump simply is the minimum or a maximum of a one-dimensional smooth function. Near a ‘B’, a tangent drawn to the stoke will be either horizontal or vertical. Bumps may occur in 4 different ways: positive / negative horizontal bumps and right/ left vertical bumps. It is stable, with codimension 0 and valence of 2.

**Wiggle (W):** Wiggle is a complex CP. As seen in figure 3, at a wiggle both the first and second derivative along U or V dimensions vanish. It is defined as

\[
\frac{dU}{dt} = \frac{d^2 U}{dt^2} = 0; \quad \frac{dV}{dt} = \frac{d^2 V}{dt^2} = 0; \quad \text{at } u=0,0, v=0,0
\]  

Wiggle is unstable under perturbation and either breaks up into two bumps or vanishes completely. Hence it has a codimension of 1 and valence of 2.

Till now we have just described four shape CP’s derived using CT. It may be striking to observe that these simple CP’s are vital components in the creation of almost 50% of the English alphabets and 70% of the English numerals (for example, C,D,O,P, S,Q,U,R,G,2,3,5,6,8,9,0). To be accurate, 73% of English alphabets and 82% of the numerals have a codimension of zero (Chakravarthy et al 2003). Inversely, from the motor point of view since most English alphabets and numerals are ‘synthesized’ with very few CP’s of high codimension, the script is very stable and robust. We now proceed to more complex shape CP’s and during the discussion illustrate some more ideas using which iCub creates a high level visual description of the teachers demonstration. Figure 4 shows the rest of the primitive shape CP’s, all having some uniquely distinguishable features in them and commonly found in several of the worlds scripts. Reader can easily visualize the valence of these CP’s and also guess their codimension by looking at some of their perturbed forms (4b).

**Figure 4.** 4a shows the remaining eight shape critical points derived by Chakravarthy et al using Catastrophe theory. 4b shows some perturbed versions of the CP’s. 4C shows how more complex CP’s can be created using simpler CP’s. For example, a ‘T’ is a combination of an end point ‘E’ and internal point ‘I’.
The document discusses various critical points (CPs) in the context of line diagrams. Here is a summary of the key points:

1. **Dot 'D'**: A stroke of zero length. It breaks down into two E points under perturbation. It has a codimension of 1 and valence of 0.

2. **Cross 'X'**: A unification of two internal points. It remains a cross under small perturbations, hence its codimension is 0 and valence 4.

3. **Cusp 'C'**: Spiky appearance and formal definition as a critical point where both U and V derivatives vanish simultaneously. It is unstable and breaks down into a simple bump or a self-intersecting loop. It has a codimension of 1 and valence of 2.

4. **T point 'T'**: The unification of an interior point and an end point. It can be restored by moving the end point, so it has a codimension of 1 and valence of 3.

5. **Star 'S'**: A unification of three interior points. It is unstable and has a valence of 6.

6. **Contact 'Co'**: When two strokes meet at a single point with the same slope, it is a contact point. Unlike a cross 'X', it is unstable with a codimension of 1 and valence of 4.

7. **Angle 'A'**: Occurs when a stroke begins where another ends, making it an unification of two end points. It has a codimension of 2 and valence of 2.

8. **Peck 'P'**: A peck is obtained by unifying a cusp with an interior point. To restore the peck, we need to move the cusp until it meets the interior point of the vertical segment, giving an appearance of a bird pecking at the bark of a tree. It has a codimension of 2 and valence of 4.

Despite its simplicity, English scripts are primarily composed of 12 critical points. The teachers' demonstration consisted of English alphabets, numerals, and random shapes. After preprocessing, the CPs can be easily computed at runtime using simple mathematical operations (computing first and second derivatives, testing valence, and testing for overlapping points). The information about the shape, described by CPs, is the abstract visual program (AVP). CT analysis is performed on the trajectories extracted from both cameras. Since both cameras capture the same demonstration at the same time, the AVPs from both cameras are consistent.
will be same. For example, CT analysis of a shape ‘U’ will show the presence of a minima (Bump) for preprocessed trajectories of demonstration recorded by both right and left cameras. However the location of the CP in the respective image planes \((U_{\text{left}}, V_{\text{left}}, U_{\text{right}}, V_{\text{right}})\) will be different. This a crucial information for 3D reconstruction of AVP to iCub’s egocentric space i.e the first step in action generation. 11 example shapes and their resulting AVP are shown in figure 5. To summarize, till now, by applying concepts of catastrophe theory we have managed to transform the teachers demonstration seen by iCub’s eyes into a concise high level representation i.e the AVP. AVP just contains the essence of the teachers demonstration to iCub i.e information about its shape. In the action generation section that follows we will see how iCub learns to draw the trajectories shown by the teacher with AVP as the starting point.

![Diagram](image)

Figure 5. 11 example shapes and the results of their critical points analysis (i.e. the resulting AVP) are shown. ‘*’ denotes the starting point. Connectivity just indicates the temporal order of the identification of critical points.

3. Emergence of Shape through Action

3.1 3D reconstruction using babbling movements

Actions are enacted in space. AVP computed in the previous section may be thought as a high level visual goal created in iCub’s brain after perceiving the teachers demonstration. To facilitate any action generation to take place, this visual goal must be transformed into an appropriate motor goal in iCub’s egocentric space. Firstly, we have to transform the shape critical points computed in the image planes of the two cameras into corresponding points in the iCub’s egocentric space by a process of 3D reconstruction. Of course the ‘type’ of the CP is conserved i.e a bump is still a bump, a cross is still a
cross. Figure 6a gives a pictorial description of the problem. In simple terms, we need to convert sets of four numbers \((U_{left}, V_{left}, U_{right}, V_{right})_{CP}\) into corresponding set of three numbers \((x, y, z)_{CP}\). To achieve this, we created a Direct Linear Transform (Shapiro, 1978) based stereo camera calibration and 3D reconstruction system for iCub. The approach is simple, quick and fairly accurate. Moreover, babbling movements by iCub itself are used to create this system.

Conventionally, 3D coordinates of a generic point in space \((x, y, z)\) can be mapped into the corresponding coordinates on the plane of the camera \((u, v)\) using equation 5. This equation is nonlinear with respect to both the transformation of coordinates and 7 unknown parameters: Camera position: \([x_0, y_0, z_0]\), Camera orientation: \([r_i]\), and the principal distance: \(d\).
The trick of the DLT (Direct Linear Transform) is that it expresses eq. 5 (non linear in 7 independent parameters) into a form like eq.6, linear in 11 parameters \((L_1 - L_{11})\) that can be experimentally computed.

\[
\begin{align*}
u - v_o &= -d \frac{r_{11}(x-x_o) + r_{12}(y-y_o) + r_{13}(z-z_o)}{r_{15}(x-x_o) + r_{12}(y-y_o) + r_{13}(z-z_o)} \\
v - v_o &= -d \frac{r_{21}(x-x_o) + r_{22}(y-y_o) + r_{23}(z-z_o)}{r_{25}(x-x_o) + r_{22}(y-y_o) + r_{23}(z-z_o)}
\end{align*}
\]

From equation 6 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. The procedure summarized below shows how data generated by iCub through a process of motor babbling is sufficient to estimate the calibration matrix and thereby achieve 3D reconstruction.

Let us consider that \(W\) is the workspace of the iCub, i.e. the set of points reachable by both arms and identified by the vision system through a visual marker (a green band tied to the two index fingers). The first step is to generate with the iCub 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 both arms 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-effectors; d) tracking the object gripped by the end-effector with the visual system and obtaining the camera plane coordinates 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 7 and 8.

\[
\begin{bmatrix}
u_1 \\
v_2 \\
\vdots \\
u_N \\
v_N
\end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\
0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\
0 & 0 & 0 & 0 & x_N & y_N & z_N & 1 & -v_Nx_N & -v_Ny_N & -v_Nz_N
\end{bmatrix} \begin{bmatrix} L_1 \\
L_2 \\
\vdots \\
L_{11}
\end{bmatrix}
\]

\(U = A \cdot L\)

\[(5)\]
As seen from equation 7, for each iteration ‘i’ of a babbling movement, we get two equations \((u_i, v_i, \text{and so on})\) 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 an initial set of 30 iterations of random bimanual movements with iCub (i.e total of 60 training points, 30 form each arm). An extended set of 30 iterations were additionally performed with a stick (of 20 cms) held in both arms, to achieve calibration in the extended tool space. From eq.7, the calibration matrix can then be estimated easily using equation 8.

\[
L = \left[ A^T A \right]^{-1} A^T \cdot U
\]  

(8)

Table 1 shows the estimated values of the calibration matrix of left and right cameras.

<table>
<thead>
<tr>
<th>Camera</th>
<th>-0.08</th>
<th>0.45</th>
<th>0.01</th>
<th>160.5</th>
<th>0.008</th>
<th>0.448</th>
<th>-0.109</th>
<th>176.11</th>
<th>0.001</th>
<th>0.0028</th>
<th>0.0005</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right Camera</td>
<td>0.03</td>
<td>0.6565</td>
<td>0.0980</td>
<td>184.88</td>
<td>0.023</td>
<td>0.434</td>
<td>0.058</td>
<td>123.15</td>
<td>0.001</td>
<td>0.0035</td>
<td>0.0005</td>
</tr>
</tbody>
</table>

After having estimated the calibration matrix \(L\) for both cameras, the reconstruction algorithm of the 3D coordinates of any target point \((u_{iR}, v_{iR}, u_{iL}, v_{iL})\) can be determined by 9 and 10 (we ignore optical distortions for the moment).

\[
\begin{bmatrix}
    u^{C1} - L^{C1}_4 \\
    v^{C1} - L^{C1}_4 \\
    u^{C2} - L^{C2}_4 \\
    v^{C2} - L^{C2}_4 \\
\end{bmatrix}
= \begin{bmatrix}
    (L^{C1}_1 - u^{C1}L^{C1}_y) & (L^{C1}_2 - u^{C1}L^{C1}_y) & (L^{C1}_3 - u^{C1}L^{C1}_y) \\
    (L^{C1}_y - v^{C1}L^{C1}_y) & (L^{C1}_y - v^{C1}L^{C1}_y) & (L^{C1}_y - v^{C1}L^{C1}_y) \\
    (L^{C2}_1 - u^{C2}L^{C2}_y) & (L^{C2}_2 - u^{C2}L^{C2}_y) & (L^{C2}_3 - u^{C2}L^{C2}_y) \\
    (L^{C2}_y - v^{C2}L^{C2}_y) & (L^{C2}_y - v^{C2}L^{C2}_y) & (L^{C2}_y - v^{C2}L^{C2}_y) \\
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]  

(9)

Hence,

\[
X = A^T \cdot Y
\]  

(10)

where \(L^{C1}\) and \(L^{C2}\) are the calibration matrixes obtained for left and right cameras respectively. Finally, the entire process involving iterations of random bimanual movements, 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 60 such iterations estimating the calibration matrix by solving equation 8 was compressed as an automated software object that executes when ever commanded by the user/or the executive system of iCub.

The performance of the proposed 3D reconstruction system in transforming image plane coordinates of salient points in the visual scene into corresponding iCub egocentric space coordinates was tested thoroughly in a series of experiments like bimanually reaching a visually observed green cylinder continuously being moved to a different location by the teacher. Other experiments like stacking, reaching with a stick an otherwise unreachable object were performed. Figure 7 shows some
of the results of the experiments testing performance of this system. Of course we will be using this system to transform the shape critical points computed in the image planes of the two cameras into corresponding points in the iCub's egocentric space. For the example in figure 6a., the primary motor goal for iCub is to learn to draw a continuous trajectory passing through three reconstructed CP’s i.e. starting from ‘E’ and having a minima (or bump) at point ‘B’ and finally terminating at end point ‘E’. Addition task specific constraints (if any) can also be added to the motor goal at this point. The body chain involved in generating the action must be defined. This is an important component of motor goal in our system because, as we will see in the next section, learnt motor actions are represented in a way that they are independent of the body chain/end effector involved in their generation. Other additional information to be specified in the motor goal are the geometric parameters of the tool (like the paint brush), joint/torque limits, application of scale to the shape if necessary, among others. In sum, we have moved one more step ahead and have transformed the abstract visual program into a concrete motor goal for iCub. Formation of CMG is the first step in the action generation process.

3.2 Shape Synthesis: From Critical points to Virtual trajectory

Given a set of two points in space, an infinite number of trajectories can be shaped passing through them. How can iCub learn to synthesize a continuous trajectory similar to the demonstrated shape using a discrete set of critical points in the CMG? How is the motor skill acquired in creating some shape represented and reused while composing more complex/novel shapes? Can iCub extract more general insights into shape synthesis itself through its various explorative sensorimotor experiences in...
creating shapes? In the following three sections we will seek answers to some of these questions. Maintaining modularity, shape synthesis is in our architecture is achieved in three closely coupled phases: Action planning, Action generation and Action learning. This section focuses on the planning phase or the synthesis of a virtual trajectory between the shape critical points in the CMG. Synthesized virtual trajectories do not really exist in space and must not be confused with the actual shapes drawn by iCub. Instead, they play a significant role in the generation of the action that creates the shape and secondly they symbolize a ‘motor equivalent’ version of the learnt action. One may visualize the virtual trajectory as an aqueduct that allows ‘energy efficient’ flow of water from one point to another, water metaphorically symbolizing the concerned end effector and its flow the creation of the shape. The virtual trajectory basically acts as an attractor landscape to body chain creating the shape. Figure 8a shows the network for virtual trajectory synthesis through two critical points. The underlying idea behind the virtual trajectory synthesis system is motivated by the seminal works on regulation of movement by Nikolai Bernstein () and impedance control by Neville Hogan (). Bernstein proposed that any movement is uniquely adapted to the ‘context’ by a hierarchical set of control systems, each exerting input into the motion generation process in sequence as well as in parallel, in such a way to reduce significantly the number of control variables. As seen in 8a, in our VTGS, the action of force fields generated by different critical points is the ‘parallel’ component and the resulting string of equilibrium points is the ‘sequential’ component. Further, the entire resulting dynamics (in all motor spaces: shape CP space, end effector space, joint space, tool space) is the result of interactions between three basic elements: generalized forces, generalized flows (position) and impedances that link them. Let \( X_{\text{ini}}(x,y,z) \) be the initial condition i.e. the point in space from where the creation of shape is expected to commence. As seen in 8b, we may visualize \( X_{\text{ini}} \) is connected to all the shape CP’s in the CMG by means of virtual springs and hence being attracted by the force fields generated by them \( F_{\text{CP}}=K_{\text{CP}}(X_{\text{CP}}-X_{\text{ini}}) \). The strength of these attractive force fields depends on: 1) the virtual stiffness ‘\( K \)’ of the springs and 2) time varying modulatory signals generated by their respective TBG’s, that basically weigh the influence of different CP’s through time. In this context, the virtual trajectory is the set of equilibrium points created during the motion of \( X_{\text{ini}} \) through time, under the influence of the net attractive field. If there are \( n \) CP’s in the CMG, the spatiotemporal evolution of virtual trajectory \((x,y,z,t)\) is equivalent to integrating non-linear differential equation that takes the following form:

\[
\frac{d}{dt} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = K_{\text{CP1}} \gamma_1(t) \begin{bmatrix} x_{\text{CP1}} - x \\ y_{\text{CP1}} - y \\ z_{\text{CP1}} - z \end{bmatrix} + \cdots + K_{\text{CPn}} \gamma_n(t) \begin{bmatrix} x_{\text{CPn}} - x \\ y_{\text{CPn}} - y \\ z_{\text{CPn}} - z \end{bmatrix}
\]

Where,

\( X \in (x, y, z) \) : state vector

\( \text{F}(X) = -\frac{\partial V}{\partial X} = K(X_{\text{targ}} - X) \) : Convergent force field to target
$K_{CP1}, K_{CPn}$ are virtual stiffness matrices of the form:

$$K = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix}$$

where, $k_{xx} = \frac{\partial F_x}{\partial x}$, $k_{xy} = \frac{\partial F_y}{\partial y}$.

Considering that the behaviour of neuromuscular system is predominantly spring like, we will consider only symmetric stiffness matrix $K$, with all non-diagonal elements zero (in other words, the resulting vector fields have zero curl). Since force fields generated are conservative ($\text{Curl}=0$), an incremental displacement from an equilibrium position $dX$ due to a force $dF$ determines a change of potential energy $\frac{1}{2}(dX^T K dX)$. The scheme of equation 11 can be rewritten as:

\begin{equation}
\dot{X} = \gamma(t) F(X), \quad \text{Finite time relaxation}
\end{equation}

\begin{equation}
\gamma(t) = \frac{\dot{\xi}}{(1 - \xi)}
\end{equation}

\begin{equation}
\dot{\xi} = 6\left(\frac{t}{T}\right)^5 - 15\left(\frac{t}{T}\right)^4 + 10\left(\frac{t}{T}\right)^3
\end{equation}
where $\xi(t)$ is a time-base generator (TBG): a scalar function that smoothly evolves from 0 to 1 with a prescribed duration $T$ and with a symmetric bell-shaped speed profile (figure 9 bottom panel). A simple choice for the TBG is the minimum jerk polynomial function of equation 13, but other types of TBGs are also applicable without any loss of generality. Terminal attractors is a non linear dynamics based approach to achieve synchronization and control over timing (without using a clock). The technique was originally proposed by Zak (1988) for speeding up the access to content addressable memories and later applied to control ‘time and timing’ in number of problems in motor control like reaching (), focal and postural synergy formation (), multi tasking (), bimanual coordination in iCub () among others. We will using terminal attractors to control timing both in the planning phase (virtual trajectory synthesis) and in action generation phase (PMP) we describe in the next section. In the VTGS, $\gamma(t)$ acts as a gating signal switching on/off and scaling the force fields generated by different CP’s and allowing relaxation to a CP in finite time $T$. In appendix A1, we elaborate in detail how terminal attractor dynamics can be applied to control the timing of the relaxation of a dynamical system to equilibrium. The central idea is that simulating the dynamics of equation 11 with different sets of stiffness $(K)$ and TBG durations results in different sets of virtual trajectories. Figure 9 shows three different trajectories synthesized by iCub, through the same CP’s and starting from same initial condition, for 3 different sets of $K$ and TBG. These shapes were drawn on a drawing board placed 32cms away from iCub along the Y-axis (see figure 6). Hence only X-Z plane (in cms) is shown. Since at all times $Y_{CP}=Y=32cm$, Y component of the force field is always zero independent of the value of $K_y$ ($K_y=1$), so only values of $K_{xx}$ and $K_{zz}$ are indicated. Bottom panel shows the timing signals $\gamma_1(t)$ and $\gamma_2(t)$, controlling the timing of relaxation to the two CP’s. Horizontal axis represents iterations in time and vertical axis represents the scalar value of the TBG signal at different time instances. Bell shaped speed profiles are also shown. With a range of virtual trajectories can be synthesized by simulating the dynamics of equation 11 with different values of virtual stiffness $K$ and timing of the TBG’s. Inversely, the goal of learning is to derive the ‘correct’ set of values for $K$ and TBG such that the resulting virtual trajectory has the same shape as the teachers demonstration.

![Figure 9](image-url)
32cms away from iCub along the Y-axis (see figure 6). Hence only X-Z plane is shown in figure 9. Bottom panel shows the timing signals $\gamma_1(t)$ and $\gamma_2(t)$, controlling the timing of relaxation to the two CP’s. During the time window when both timing signals overlap, the force fields generated by both CP’s influence the dynamics. A range of virtual trajectories can be synthesized by titillating the space of K and TBG (duration and overlap). The goal of learning is to arrive at the ‘correct’ set of values for K and TBG such that the resulting virtual trajectory has the same shape as the demonstration. For the example of synthesizing a shape with two CP’s, the goal is to learn the values of $K_{xx}, K_{zz}$ and $T$ for CP$_1$ and CP$_2$.

3.3 Coupling the body/internal body model to the Virtual trajectory

Through a series of transformations described so far, we have gradually moved from the teachers demonstration to AVP, from AVP to CMG and then from CMG to the synthesis of the virtual trajectory as described in the previous section. The next step is to derive the motor commands for the designated body (body+ tool) chain, which will ultimately transform the virtual trajectory into a real trajectory created by iCub. The passive motion paradigm based forward/inverse model for iCub upper body coordination (Mohan et al, 2009) is used in this phase of action generation. This computational framework has already been successfully applied for a range of tasks like reaching, bimanual coordination and tool use (reaching, pushing, bimanual transportation) using iCub. The scenario in which we apply the framework here is not mere ‘reaching’, but ‘shaping’ i.e. reaching in a very specific manner along the path specified by the synthesized virtual trajectory. Hence at the very onset, we clarify that virtual trajectory synthesis and action generation (motor command synthesis) do not sequentially follow each other, but occur concurrently and co evolve together.

Intuitively, one may visualize the scenario of a puppeteer artistically pulling the designated end effector of a puppet along some virtual trajectory that he has in his mind. From the puppets point of view, its body is elastically reconfiguring itself to new postures continuously so as to allow its end effector to comply to the externally induced pull. Hence, as its end effector tracks the evolving virtual trajectory, a trajectory motor commands needed to allow the end effector to do so are also concurrently being synthesized at the other end (i.e. the puppets joint space). Let us now consider that the puppet was in fact iCub’s internal body model, and the whole process of relaxation a mental simulation of action in reaction to the goal induced pull (of the VT, here goal is the puppeteer). If the trajectory of motor commands derived through this process of virtual relaxation is actively sent to the actuators, iCub will enact the same trajectory. This is the central function realized by the network of figure 11. Further, even if the motor commands are not sent to the actuators, the forward model actively predicts its consequences and the resulting end effector trajectory can be visualized in the mind. Hence while learning to imitate the teachers demonstration, iCub can explore a series of solutions in its mental space before performing the best possible imitative action in its physical space. So a key feature of PMP, is that the same model is used to support mental simulation of action and the actual delivery of motor commands during movement execution.
Let \( x \in (X,Y,Z) \) be the vector that identifies the pose of the end-effector of iCub in the extrinsic workspace and \( q \) the vector that identifies the configuration of the iCub in the intrinsic joint space. \( x = f(q) \) is the kinematic transformation that can be expressed, for each time instant, as follows:

\[
\dot{x} = J(q) \dot{q}
\]

where \( J(q) \) is the Jacobian matrix of the transformation. If \( x_T \) is the vector describing the evolving virtual trajectory (i.e., the goal), the motor planner/controller, which expresses in computational terms the passive motion paradigm, is defined by the following steps:

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

\[
F = K_e (x_T - x)
\]

where \( K_e \) is the virtual impedance matrix in the extrinsic space of left arm. The intensity this force field decreases monotonically as the end-effector approaches the target. Note that \( K_e \) is the virtual stiffness connecting the end-effector/tool to the evolving virtual trajectory and is different from virtual stiffness...
2) Map the force field into an equivalent torque field in the intrinsic space, according to the principle of virtual works:

\[ T = J^T F \]  

(15)

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

3) Relax the arm configuration in the applied field:

\[ \dot{q} = A_{\text{int}} \cdot T \]

(16)

where \( A_{\text{int}} \) 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:

\[ \dot{x} = J \cdot \dot{q} \]

(17)

The basic PMP network (expressed by equations 14-17) is an asymptotically stable dynamical system with a ‘moving’ point attractor that brings the end-effector to the target if the target is indeed reachable. However, asymptotic 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. It is like a mechanism of cortical interaction without subcortical control, with some analogy to the pathological behavior of Parkinson patients. Timing of the PMP relaxation can be controlled exactly in the same way as in VTGS by using the concept of terminal attractor dynamics.

This simply requires that equation 17 is substituted by the following one:

\[ \dot{x} = \Gamma(t) \cdot J \cdot \dot{q} \]

(17a)

\[ \Gamma(t) = \frac{\xi}{(1-\xi)} \]

\[ \xi(t) = 6 \cdot (t/\tau)^5 - 15(t/\tau)^4 + 10(t/\tau)^3 \]

(18)

Here \( \tau \) is the planned duration of the movement and \( \xi(t) \) is a function generator that follows a minimum-jerk profile from 0 to 1.

5) Integrate over time until equilibrium

\[ x(t) = \int_{t_0}^{t} J \dot{q} d\tau \]

(19)

Integrating equation 17a over time we obtain a trajectory in the extrinsic space, that basically tracks the goal, in our case the evolving virtual trajectory synthesized by the VTGS (see figure 12, panel D). An indirect consequence of the interaction between the internal model and the goal is that the trajectory of motor commands is synthesized concurrently at the intrinsic space (each point in this trajectory is 10 dimensional: 7 DoF for the left arm and 3 DoF waist: figure 12 panel F). Hence what
we have is a simple non-linear dynamics based system that at runtime transforms the synthesized virtual trajectory into motor commands for iCub’s body. When the obtained trajectory of motor commands are buffered to the actuators, iCub will generate the action that transforms the virtual trajectory into a real trajectory (in space, on a drawing board etc according to the task specification).

We refer the interested reader to an article in a previous issue of this journal \(\) for a more intricate analysis of PMP in relation motion planning and coordination of humanoids, mainly iCub. For a demonstrated trajectory like in figure 2, an analysis of the coupled interaction between virtual trajectory synthesis system and the action generation system is shown in figure 12.

![Figure 12](image)

Figure 12. These graphs were obtained when iCub was given a goal to draw a shape (as seen in figure 2) on a drawing board placed -320mm in front of it (along the Y axis, for coordinate system see figure 6a) using a paint brush (12cm in length). Panels A-C show the evolution of iCub’s end effector trajectory along X, Y and Z axes as a function of time. Note that the motion of the end effector along Y-axis is almost negligible (about 5 mm). Hence, the additional constraint of making a planar trajectory while moving in a 3D space is successfully satisfied (i.e. the paint brush is always in contact with the drawing board, slight inaccuracies in few mm range are resisted by the compliance of the brush itself). Since motion along Y-Axis is negligible, panel D shows the resulting trajectory in the X-Z plane. The synthesized virtual trajectory (i.e the goal) that triggered the PMP relaxation is also shown. As seen in panel D, the end effector basically tracks the evolving virtual trajectory synthesized by the VTGS (analogous to the water flowing in an aqueduct). In other words, the virtual trajectory acts as an attractor to the internal body model that is dynamically coupled to it based on the task specification. An indirect consequence of this relaxation process is that a trajectory of motor commands is being synthesized concurrently at the intrinsic space as shown in panel F (analogous to the puppet metaphor). Each point in this trajectory is 10 dimensional: 7 DoF for the left arm (J3-J9) and 3 DoF waist (J0-J2). The CAD diagram of panel E clarifies the intrinsic degrees of freedom. The time base generator functions are also shown in panel F along with the temporal evolution of each intrinsic DoF. As seen in panel F, finally a complete motor command set i.e a matrix of 3000 rows (iterations in time) each having 10 columns (values of the 10 joint angles at each time instance) is derived.
3.4 Learning to Shape

We saw in the previous section how the virtual trajectory can be transformed into a real trajectory (drawn now by iCub) if the synthesized set of motor commands (Panel F) are buffered to the actuators for execution. With this, we have finally traversed the loop beginning from the visual perception of a teacher’s demonstration to the motor commands of the student. The final step is to ensure that the actions generated by the student closely matches that of the teacher. This implies that the causal effect of the action i.e the shape created by the student must closely match the shape demonstrated by the teacher. There are two additional coupled mechanisms we need to create in iCub to achieve this objective.

1) A way to self evaluate and score its performance in comparison with the observed demonstration

2) A way explore and generate a range of solutions/shapes and adapting them systematically based on the received self evaluations

In the following two sub sections we will attack these problems, once again aided by results of experiments conducted with iCub.

3.4.1 Comparing abstract Motor and Visual representations to evaluate performance

It is easy to observe that the PMP network of figure 11 naturally forms a forward/inverse model chain. During the process of relaxation to a goal, while the inverse model computes the necessary motor commands, the forward model predicts the consequence of the motor actions (i.e. the resulting trajectory generated by the end effector, panel D). Hence, the shape of the trajectory drawn by iCub can be easily computed by performing a CT analysis of the end effector trajectory predicted by the forward model. This information, being the result of the CT analysis of iCub’s own motor output, we call it as the Abstract Motor Program (AMP). Further, information in CMG and AMP are in the same reference system (iCub’s egocentric frame), one describing the shape demonstrated by the teacher and the other describing the shape created by iCub. Hence a direct comparison is feasible. The greater the equivalence between AMP and CMG, the greater is the score of performance.

How do we compute the equivalence between ‘shapes’ of two different trajectories? In other words, can we compare two shapes (in general) and arrive at a scalar number that measures the relative equivalence between them? We will answer these questions with the help of an example of comparing a demonstrated shape ‘C’ (i.e. CMG) with a set of five different attempts made to draw it (AMP1-5, figure 13). In general, let us consider that we have two shapes S1 and S2 we wish to compare. Let n and m be the number of critical points detected in the CT analysis of S1 and S2 respectively. In the ideal case n=m, but there can be pathological cases (for example in figure 13, AMP5 has 4CP’s while CMG has 2).

So the first step is to establish a correspondence between CP’s in S1 and S2. Of course this problem is not as complex as the correspondence problem in vision, because we are just comparing two shapes and if the two shapes match then the locations of the CP’s in both of them must ideally coincide. Hence
correspondence between CP’s in two shapes can be solved by assigning a region of influence for each CP in the goal shape (S1). We took a circle of radius 30mm around a CP as the region of influence (shown by yellow circle around CP in CMG). If a CP in S2 falls inside this region, then the two CP’s are matched. If there are multiple CP’s in S2 that fall in the region of influence of a particular CP in S1 (these cases are quite rare), the CP of same type is matched, the rest remain unmatched. If there are two CP’s of same type, then the one that is closer is matched, the other remains unmatched. Thus the ‘m’ CP’s in S2 can be divided into ‘p’ CP’s that correspond to S1 and ‘q’ CP’s that remain unmatched. At this stage, we are not bothered about the type of the CP. All we are finding out is which CP in S2 that falls in the region of influence of a CP in S1. As seen in figure 13, the CMG has 2 CP’s (a Bump and an End point, we will omit the starting point E* considering that it is just the initial condition). The solutions AMP\(_{1,4}\) also have 2 CP’s, each lying inside the region of influence of a CP in CMG (So p=2,q=0). AMP\(_5\) is a much worse solution having 2 CP’s (Bumps) that don’t correspond to any CP in CMG. Hence for AMP\(_5\), m=3 (B,B,and E), p=1(End point that corresponds to end point of CMG) and q=3(two Bumps of AMP\(_5\) and one bump in CMG that are not matched).

Figure 13. Shows comparison of the goal shape (CMG) with five different attempts to create it (AMP\(_{1,5}\)). AMP\(_1\) and CMG have same number of CP’s and located in the same places. However, the type of CP is different i.e there is a cusp at the place where a bump is present in the goal. Since in AMP\(_2\) and AMP\(_3\) the CP’s (B and E respectively) were prematurely created their score is lesser that the best solution AMP\(_4\) that more closely matches the CMG. Premature creation of a CP or premature termination of a shape can result due to wrong timing signals generated by the TBG. Still, AMP\(_2\) and AMP\(_3\) score greater than AMP\(_1\) because they nevertheless share the same essence of CMG. AMP\(_5\) is a pathological case where there is a mismatch in the number of CP’s in the goal shape and the created shape. Neither of the two Bumps in AMP\(_5\) fall in the region of influence of the bump CP in CMG. So there are 3 mismatched CP’s. The greater the mismatch in the number, type and proximity of CP’s between the perceived and self generated shapes, the lesser is the score. In this case AMP\(_4\) is the best solution with score of 1. It is followed by AMP\(_2\) and AMP\(_3\) that come close to the goal shape at least in terms of its essence. AMP\(_1\) and AMP\(_5\) come last since they share very less similarity with the goal shape.

Now, if there are p matched CP’s and q excess (unmatched) CP’s between shape S1 and S2, the score of how closely S1 resembles S2 can be computed as follows:
\[ S = \frac{1}{p} \left( \sum_{CP_{S1} \neq CP_{S2}}^{p} \Psi(\text{CP}_{S1}, \text{CP}_{S2}) \cdot \text{dist}((\text{CP}_{S1}, \text{CP}_{S2})) - q \right) \]

Where

\[ \Psi(\text{CP}_{S1}, \text{CP}_{S2}) = 1 \text{ if CP}_{S1} \text{ and CP}_{S2} \text{ are of the same type} \] (21)

Otherwise, \( \Psi(\text{CP}_{S1}, \text{CP}_{S2}) = 0 \)

\[ \text{dist}(\text{CP}_{S1}, \text{CP}_{S2}) = e^{-\frac{(x_{S1}-x_{S2})^2}{25}} \] (22)

It is very easy to understand the effect of equation 20. The function \( \psi \) imposes the condition that the two CP's must be of same type (a bump in S1 should correspond to bump in S2). Hence as we see in AMP_{b}, even though the different CP's are matched perfectly (q=0) and also are located in the same position, the score is very less because a cusp was found in the place where bump was expected. The function 'dist' imposes the condition of physical proximity i.e. the corresponding CP's must ideally coincide if the shapes match perfectly. Since in AMP_{2} and AMP_{3} the CP's (B and E respectively) were prematurely created their score is lesser that the best solution AMP_{4} that more closely matches the CMG. However, AMP_{2} and AMP_{3} score greater than AMP1 because they nevertheless share the same essence of CMG. The term '1/p' is a normalizing element. The final term '-q' basically penalizes presence of excess/unmatched CP's. The more there are unmatched CP's, the more the two shapes differ and hence lesser is the score. The scheme is extremely simple and robust and can be applied to compare shape of any two line diagrams in general. Further it doesn't give a digital answer. It basically says how much a shape resembles some other shape in a continuous analog way.

3.4.2 Creating a diversity of ‘Shapes’

Since the end effector basically tracks the virtual trajectory (figure 12, Panel D), the problem of learning to shape can be formulated as a problem of learning to synthesize the appropriate virtual trajectory such that the shape of the resulting end effector trajectory predicted by the forward model correlates with the shape of the trajectory demonstrated by the teacher (i.e. represented by AVP or CMG). It implies learning to synthesize the correct attractor landscape that triggers behavior which fetches maximum rewards. Once the correct virtual trajectory is obtained it can be easily transformed into motor commands using PMP. We already saw in section 3.2 that range of virtual trajectories can be synthesized by simulating the dynamics of equation 11 with different values of virtual stiffness K and timing of the TBG's. Combining the process of exploration with a mechanism of evaluation presented in the previous section, the overall learning loop consists of the following steps (shown schematically in figure 18c):

1) Creation of an initial spread or an inaccurate discretized map of the space of parameters (K and \( \chi \))

2) Synthesis of the virtual trajectory using these parameters
3) Coupling the internal model to the virtual trajectory to generate the motor action and its consequence (resulting end effector trajectory). This is a mental simulation of action and is very fast.

4) Transforming forward model output into an AMP and evaluating the score of performance

5) Choosing an initial range of parameters that give a reasonable score (greater than 0.65)

6) Fine tuning: Creating a new, more accurate discretized map of parameters in this new range of optimal values

7) Return to step 2, now till the performance raised to a score greater than 0.86 (or greater)

The above loop includes all the subsystems we described after the creation of CMG (which is done once looking at the teacher). We now ponder in detail about four questions that arise specific to our architecture and to any learning process in general:

1) How long should exploration be conducted and will the system converge to a solution?

2) What is the time taken (in a computational sense) to run an explorative learning for one shape?

3) Can we arrive at a well defined range of values inside which if exploration is conducted, it would be possible to synthesize at least a large assortment (if not all) shapes?

4) Are there any general underlying principles as to why a specific set of ‘virtual stiffness and timing’ results in a specific shape? Understanding this significantly reduces the effort in exploration because iCub can then use this knowledge to anticipate the parameters that should correspond to the synthesis of a specific virtual trajectory, hence driving the synthesis of a specific shape.

We will answer question 2 first. Since the loop of exploration, action and evaluation takes place in the mental space, the time for running these simulations is very less. iCub can run about 60 explorative mental simulations of drawing a moderately complex shape like ‘S’ in five minutes. Later it can physically execute the best possible solution, since the trajectory of motor commands are always synthesized at the other end during all mental simulations. The answer to questions 1, 3 and 4 are related in the sense that if the answers for questions 3 and 4 are in the affirmative (which we shall show), then the system will converge to a solution in reasonable number of trials. Further, the number of explorative trials will decrease significantly as more insight into synthesizing specific shapes is gained. In later stages only few simulations for fine tuning will be needed. From a very general perspective, we also note that exploration must never be looked at as a separate process. There is always counter balancing force called as ‘motivation’, that basically prevents the exploration form going on perpetually. In such cases, either a different strategy is attempted or the goal is dropped (Mohan et al). While learning to draw, we assume that iCub is highly motivated baby humanoid.

We start with the basics, teaching iCub to draw the primitive shape features that we derived in section 2 using catastrophe theory. Since more complex shapes can be ‘decomposed’ into combinations of these primitives, inversely the actions needed to synthesize them can ‘composed’ using combinations of the corresponding ‘learnt’ primitive actions. Hence maximum effort and exploration on the part of iCub is required while learning the basics (drawing the primitives). If this
knowledge can be systematically bootstrapped while drawing more complex line diagrams, only small amount of fine tuning will be needed in the later stages. From the action generation perspective, once iCub learns to synthesize straight lines, bumps and cusps, the rest of the primitives can be easily composed by combining/sequencing them appropriately.

The stiffness matrices $K$ described in equation 11 are basically open parameters with the only constraint that they be positive definite. In general, while reaching between two points, we may even renounce the knowledge of their precise value (or set $K$ as an Identity matrix), since displacement to a new position is always obtained by a relaxation paradigm that aims force $dF$ at a target and keeps cycling through the network until the target is reached. This process, called as force driving (Mussa-Ivaldi et al, 1988), is used in a virtual sense in the VTGS. However, if we actively modulate the values of $K$, the attractive force fields generated by the critical points (according to equation 11) can be isotropic or anisotropic according to the fact that the eigenvalues of virtual stiffness matrix $K$ are equal or unequal. The flowlines in the former case are straight lines and are curved in the latter case. Once the correct values of $K$ is estimated for creating specific shape features (Cusp, Bump etc), they can be stored as a heteroassociative memory and used in future. To begin with, let us consider that we want to generate different virtual trajectories between two points starting from $X_{ini}$ and terminating at $X_T$.

Since there is only one target CP (i.e. $X_T$), equation 11 can be simplified as:

$$\frac{d}{dt} \begin{bmatrix} x \\ z \end{bmatrix} = \gamma(t) \begin{bmatrix} k_{xx} & 0 \\ 0 & k_{zz} \end{bmatrix} \begin{bmatrix} x_t - x_{ini} \\ z_t - z_{ini} \end{bmatrix}$$

The timing of the relaxation is set to 1.5 time units (2000 iterations, exactly similar to $v_1(t)$ of figure 12F). Figure 18a shows a set of virtual trajectories synthesized between the two points when the virtual stiffness matrix elements ($K_{xx}$ and $K_{zz}$) are varied from 1 to 10 respectively (in increments of 2). As we can see, a rich set of trajectories of varying curvature can be created within this range itself. We also observe that when the $x$ and $z$ components of the stiffness are alike, we get straight line trajectories. As the discrepancy increases (for example, $K_{xx}=9$ and $K_{zz}=1$), more and more curved virtual trajectories are obtained. As seen in figure 18b, The curvature as a function of virtual stiffness is independent of the initial and final condition ($X_{ini}$ and $X_T$). Figure 18b shows the same process of pseudo random titillations in the space of virtual stiffness while synthesizing trajectories form $X_{ini}$ to three different final positions. As we will see later, it is also independent of scale. Hence with the same values of virtual stiffness, same curvature in different scale can be synthesized.

The simplest possible shape is just a straight line. In the AVP (and equivalently in the CMG), it is just described using two CP's (both of type 'E', representing the start and end of the trajectory). Since there is no additional shape CP involved in the description, this is a default case. From the explorative trials in figure 18, iCub easily learns the appropriate parameters needed to create a straight line between two points (simply, $K_{xx}=K_{zz}$). Even though simple it may be, straight lines are the most common shapes we create during movements, and further are essential components in many of the worlds scripts. It is
known from almost three decades (Morasso, 1981) that human reaching movements show straight line trajectories with a symmetric bell shaped velocity profiles.

Let us now involve bit of volitional control and consider that iCub has the goal to create specific trajectories (other than straight lines). These trajectories belong to the set of shapes that are described using three CP’s in the AVP (two end points ‘E’ and one CP that gives more information about the specifics). Bumps and Cusps (oriented in different directions) come in this category. Once iCub learns to draw them, all the remaining shape CP’s derived in section 2 can be created by combining straight lines, bumps and cusps.

![Graph showing various end effectors trajectories predicted by the forward model in response to the motor commands synthesized while tracking the attractor landscapes created by the corresponding virtual trajectories. Since all the shapes created during the learning phase cannot be shown due to limitations of space, we have arranged the graphs in a way such that more intuitive understanding can be gained. All shapes in figure 19 have 3 CP’s (indicated with a star according to their description in the CMG). Hence there are two sets of virtual stiffness matrices and two TBG’s. Note that the indicated stiffness belong to the Virtual trajectory synthesis system. When the synthesized virtual trajectory is coupled to the PMP relaxation, the motor

Figure 18. Panel A shows a set of virtual trajectories synthesized between $X_{ini}$ and $X_T$ when the virtual stiffness matrix elements ($K_{xx}$ and $K_{zz}$) are pseudorandomly titillated between 1to 10 respectively (in increments of 2). As seen, a gamut of virtual trajectories of varying curvature can be synthesized. Also note that when the X and Z components of virtual stiffness (hence the resulting force field components) are similar, straighter trajectories are synthesized. As they begin to diverge, more curved trajectories are obtained. As shown in panel B, this behaviour is independent of the initial/final conditions and also the scale. Panel C schematically describes the learning loop involving pseudorandom exploration in the space of virtual stiffness and timing, coupled with self evaluations of the resulting performance (in mental space).
commands and corresponding forward model prediction of the end effector trajectory shown in figure 19 is obtained. Shapes in every column have been drawn by iCub with same values of the virtual stiffness (indicated at the bottom). Shapes in the first two rows (i.e panels 1-6) share the same timing and similarly shapes in the second two rows (panels 7-12) share the same timing. The timing graph is shown along the sides. We can observe that all the shapes at least share the essence of being either cusps or bumps, of course varying in quality. Coming to the first column, when the x and z components of stiffness are proportional (or equal), we see straighter trajectories are generated. For example, the shape in panel 10 resembles the English alphabet ‘V’. Even though some shapes in panels 1-6 resemble cusps and bumps, we see that they are rather stunted and some suffer from overshoots too. This effect is can be attributed to the result of improper timing, since the shapes in panels 7-12 (which also share similar stiffness’) do not show these pathologies when created with a different timing signal. Stunted shapes can also be seen if the values of stiffness’ are low. We many intuitively imagine that the puppeteer is not pulling the puppet with enough force (in the case of suboptimal virtual stiffness’) or not pulling long enough (in the case of suboptimal timing).

Let’s now focus on panels 8,9,11and 12, all driven by same timing. The Bump in shape 8 and cusp in shape 11 are created using the same values of the virtual stiffness. Even though the values of the stiffness matrices were arrived at by random exploration (in range of 1-10), it is possible to observe some underlying order. During the first phase of trajectory synthesis (when the goal of moving from $X_{ini}$ to CP$_1$ is more dominant) the x-component of the virtual stiffness is 10 times the y-component. Hence as we observe in both panel 8 and 11, there is an initial evolution of the trajectory more along the horizontal direction as compared to the vertical direction. As the time varying gain $\gamma_1$ increases, the temporal force to reach CP$_1$ also increases. So in the later stages we see a smooth culmination of the trajectory at CP$_1$. At the same time CP$_2$ is also beginning to exert attractive pull with the triggering of $\gamma_2$. During the second phase when force field generated by CP$_2$ is more dominant, we see the scenario in terms of the virtual stiffness is reversed i.e. $K_{xx}=10K_{zz}$. So there is an initial exaggerated evolution of the trajectory along the vertical direction, and in the later stages along the horizontal direction before finally terminating at CP$_2$. Based on the spatial arrangement of the CP’s, it is easy to visualize that in the global presence of such time varying goal induced force fields, we should get a bump in panel 8 and cusp in panel 9. We are sure that the reader can easily visualize why with the virtual stiffness’ configuration in column 3, iCub manages to create a Bump in panel 9 and cusp in panel 12.

Figure 20 presents scanned images of the first drawings created by iCub using the architecture presented in this article. Figure 21 shows few snapshots of iCub during the process of drawing some shapes shown in figure 20. As mentioned earlier, the drawing board was placed approximately 32cms along the Y axis. On the drawing board, a 3cm thick layer of sponge (black color) was attached. Drawings were created by iCub on a white sheets of paper pinned to this layer. We also created a special paint brush (12cms in length), thick and soft enough to be grasped by iCub’s fingers. The grasp
Figure 19. Different end effector trajectories of iCub predicted by the forward model in response to the motor commands synthesized while learning to draw bumps and cusps are shown. For reasons of space, only an eclectic set of shapes generated during the learning phase are shown. The graphs are arranged in a way such that an intuitive understanding can be gained of why specific parameters drive creation of specific shapes. All shapes have 3 CP’s (indicated with a star according to their description in the CMG). Hence there are two sets of virtual stiffness matrices and two TBG’s (Note: These parameters belong to the VTGS). Shapes drawn by iCub with same set of the virtual stiffness are arranged in a column. The values of the corresponding virtual stiffness’ are indicated at the bottom of each column. Shapes in the first two rows (i.e panels 1-6) share the same timing and similarly shapes in the second two rows (panels 7-12) share the same timing. The timing graph is shown along the sides.

We can observe that all the shapes at least share the essence of being either cusps or bumps, of course varying in quality. In the first column, where the $K_{xx}=K_{zz}$, we see straighter trajectories. For example, the shape in panel 10 resembles the English alphabet ‘V’. Even though some shapes in panels 1-6 resemble cusps or bumps, we see that they are rather stunted. Some suffer from overshoots too. This effect is can be attributed to the result of improper timing, since the shapes in panels 7-12 (despite having similar stiffness’) do not show these pathologies when created with a different timing signal. Stunted shapes can also be seen if the values of stiffness’ are low. We many intuitively imagine that the puppeteer is not pulling the puppet with enough force (in the case of suboptimal virtual stiffness’) or not pulling long enough (in the case of suboptimal timing).

Shapes in panels 8,9,11and 12 are all created using the same timing signal. Further, the Bump in shape 8 and cusp in shape 11 are created using the same values of the virtual stiffness. Even though the values of the stiffness matrices were arrived at by random exploration (in range of 1-10), it is possible to observe some underlying order. During the first phase of trajectory synthesis (when the goal of moving from $X_{ini}$ to CP$_1$ is more dominant) the x-component of the virtual stiffness is 10 times the y-component. Hence as we observe in both panel 8 and 11, there is an exaggerated initial evolution of the trajectory more along the horizontal direction as compared to the vertical direction. As the time varying gain $\gamma_1$ increases, the temporal force to reach CP$_1$ also increases. At the same time CP$_2$ is also beginning to exert attractive pull with the triggering of $\gamma_2$. So in the later stages we see a smooth culmination of the trajectory at CP$_1$. During the second phase when force field generated by CP$_2$ is more dominant, we see that scenario in terms of the virtual stiffness is reversed (i.e $K_{zz}=10K_{xx}$). So there is an initial exaggerated evolution of the trajectory along the vertical direction, and in the later stages along the horizontal direction before finally terminating at CP$_2$. Based on the spatial arrangement of the CP’s, it is easy to visualize that in the global presence of such time varying goal induced force fields, we should get a bump in panel 8 and cusp in panel 9. The reader can easily visualize why with the virtual stiffness’ configuration in column 3, iCub manages to create a Bump in panel 9 and cusp in panel 12.
routines are automated, work is ongoing to upgrade the grasp module based on the shape perception-synthesis principles described in this paper. Note that the layer of sponge and the bristles of the paint brush add natural ‘compliance’ to the overall system and ensures safe and soft interaction during contact. Panels A-C show the set of shapes drawn by iCub during the explorative training phase. Apart from these, a majority of trials were enacted in the mental space. We can see that many shapes of figure 20 correspond to the predicted forward model outputs of figure 19.

3.5 iCub that writes ‘iCub’ – Generalization during compositional synthesis

As we saw in the previous section, iCub now has the motor repertoire to draw at least the simplest shape features i.e bumps, cusps and straight lines. Despite being simple, they are also the most common shape features we encounter (either existing independently or as local parts of more complex shapes). Further, all the other shape critical points derived in section 2 can be seen as combinations of these simple features. In this section we will explore the issue of how well the knowledge gained by iCub to draw these simple shapes can be exploited while generating more complex shapes/drawings.

The underlying idea we investigate is that since complex shapes can be ‘decomposed’ into combinations of primitive shape CP’s using CT, inversely can the motor actions needed to create them be ‘composed’ using combinations of the corresponding ‘learnt’ primitive actions? We classify the compositions into two types: 1) Multiple stroke compositions and 2) Single stroke compositions. Shapes like cross ‘X’, star ‘S’, contact ‘Co’, peck ‘P’ shown in figure 4 are examples of multiple stroke compositions. iCub can directly draw these shapes using the knowledge it gained in the previous section with the only addition of a ‘Pen up-Pen down’ action (PUPD henceforth). PUPD can be characterized as an additional reaching movement to a new target point in space once contact with the paper is withdrawn and can be easily achieved using the network of figure 11. Shapes like the English alphabet ‘S’ or the numeral ‘8’ are examples of the latter case. These trajectories are shaped in single stroke and contain a mixture of the primitive shape features dominant at different spatial locations. In these shapes consisting of many CP’s and completed in a single stroke, the critical factor is to maintain a measure of smoothness while transiting from one shape feature to another. The appropriate virtual stiffness’ values learnt previously for synthesizing the primitives can still be taken as a starting point to create a first prototype, followed by small amount of fine tuning to get the right smoothness and continuity while moving from one shape feature to another. Basically, by providing approximately the right parameters needed for synthesis (for example a maxima is always a maxima), the previously learnt knowledge drastically minimizes the space of exploration. One may imagine a scenario of a manual eye testing, where the optician is faced with the problem of estimating the right optical power of the eye glasses necessary for a patient. If he already knows the optical power of the old eyeglasses used by the patient, in most cases he can explore around that range to effortlessly estimate the new dioptric value (assuming there are no drastic changes, which is valid for us). To demonstrate how knowledge gained in synthesizing primitives can be efficiently exploited in creating more complex...
Figure 21. A collection of snapshots of iCub during its very first creative endeavors. The drawing board was placed approximately 32cms along the Y axis. On the drawing board, a 3cm thick layer of sponge (black color) was attached. Drawings were created by iCub on a white sheets of paper pinned to this layer. We created a special paint brush (12cms in length), thick and soft enough to be grasped by iCub’s fingers. Note that the layer of sponge and the bristles of the paint brush add natural ‘compliance’ to the overall system and ensures safe and soft interaction during contact.

Figure 20. Scanned images of the first drawings created by iCub. Panels A-C show series of explorative attempts to draw a ‘C’ and ‘U’. Panels D-F show shapes created in sequence while learning to trace a ‘U’. Panels L,J,K are shapes resulting from stiffness configuration in columns 1,3 and 2 of figure 19 respectively. Panels L and M show an example of a bad and good ‘C’.
shapes, we present 3 examples of composite shapes iCub learnt to draw quite effortlessly: 1) the English alphabet ‘S’ (6CP’s, single stroke); 2) writing its own name i.e. ‘iCub’ (multiple strokes, character ‘b’ is a composite shape); 3) a simple pencil sketch (marginally resembling the Indian freedom fighter Mahatma Gandhi).

The shape ‘S’ consists of six CP’s (2 end points and 4 bumps in different orientations as seen in figure 22: left panel) and is executed in a single stroke. Since the shape has 6 CP’s, during synthesis we considered 3 CP’s at a time (i.e initial point, Target1 and Target2), and used a FIFO to buffer new CP’s (and related virtual stiffness’ based on the nature of the CP) once an existing CP is reached. The motor knowledge iCub gained previously to draw bumps is applied to arrive at a first prototype of the shape ‘S’ (which itself as seen in figure 22 is an acceptable version). Fine tuning around the previously learnt virtual stiffness’ values (for bump CP), results in subtle changes in the resulting shapes (see shape2 and 3). Larger deviations on the other hand deteriorate the overall ‘S-ness’ (see shapes 4,5,8). A good enough solution is arrived at in just the 9th iteration (shape 9). In addition to the 12 primitive shape features, iCub now also has the motor repertoire to make one composite shape. All this learnt knowledge is memorized in the form of a heteroassociative memory, associating the AVP (of a shape, drawing, demonstration) to the corresponding ‘virtual stiffness and timing set’ that drives its synthesis. This knowledge then can be used further in the synthesis of more complex shapes (or AVP’s) that are either similar to the previously learnt shapes or contain them as sub-shapes. For example, iCub exploited the knowledge it gained while learning to draw a ‘C’ and ‘U’ to quickly learn a composite action to draw ‘S’. This new knowledge can be exploited while creating more complex shapes for which ‘S’ is a sub shape (like ‘8’ or others like a double helix etc).

Figure 22. Left Panel shows the goal shape ‘S’. The AVP of the shape consists of 6 CP’s (2 end points ‘E’ and 4 bumps ‘B’). Center and right panels shows the sequence of 9 shapes traced by iCub during learning iterations consisting of small random titillations around the range virtual stiffness values learnt previously for drawing bump CP ‘B’. As seen, right from the beginning a degree of ‘S-ness’ is present in all the drawn trajectories. In just the third trial we almost reach an acceptable prototype of ‘S’. Larger deviations form the previously learnt parameters deteriorates the global shape (trials 4-6). The ninth trial results in a ‘S’ of score 0.94 and the loop terminates.
Figures 24 and 25 present two more examples of synthesis of composite shapes 1) resembling a pencil sketch of Gandhi (side view) and 2) iCub writing its own name i.e ‘iCub’.  Both are examples of multiple stroke composite shapes. The former (i.e Gandhi sketch) is a composition of bumps, cusps and T shape features locally distributed in space. The creation of this drawing requires a sequence of three smooth strokes and two Pen up-Pen down operations (also implemented using PMP).

The target shape, its AVP, the solution set generated during learning trials and final solution (end effector trajectory) are shown in panels A,B,C and D of figure 24 respectively. The latter case is an example of moving from ‘shaping’ to ‘writing’ a string of characters on paper. The additional issue in this scenario is to plan the initial position to begin the next character. It is worth noting that the location from where the writing of the next alphabet begins is variable and depends on the spatial distribution of the alphabet itself. For example, ‘C’ is a shape that swells leftwards (towards the previous character) and ‘U’ is a shape that bulges rightwards. The initial position to begin the next alphabet must on one hand ensure uniformity in spacing and on the other hand avoid overlaps between successive characters. Even though this is an effortless exercise for human writers, for robots we need to create a module that outputs the new initial position (x,y,z in the drawing board) form where to start the next character, once the paint brush is withdrawn after completion of the previous alphabet.

Figure 23. Left panels show the evolution of iCub’s end effector trajectory along X,Y and Z axes as a function of time while drawing the shape ‘S’. The trajectories of motor commands synthesized at the intrinsic space which drive the action generation are shown in the right panel. Motor commands consist of 10 trajectories (J0-J9), corresponding to temporal evolution of the 7 DoF of the iCub’s left arm (J3-J9) and 3 DoF of the waist (J0-J2). Hence, the complete motor command set buffered to the actuators is a matrix of 7500 rows (iterations in time) each having 10 columns (values of the 10 joint angles at each time instance). The time base generator functions (t) are also shown.
Before performing experiments on the iCub itself, we conducted a series of simulations of the synthesis of a range of composite shapes, first on a 3 DoF planar arm (in Matlab) and then on the iCub simulation environment. This is easily possible because of the inherent modularity in the architecture. The virtual trajectory can be used as an attractor for any body chain /robot and PMP on the other hand guarantees the synthesis of motor commands necessary to transform the virtual trajectory to a real trajectory. In the iCub simulator, iCub creates the shape with its index finger. Small static boxes (unaffected by gravity) trace the trajectory followed by the index finger. Panel E shows the generation of the character string ‘iCub’ by a 3 DoF planar arm. In this case, the motion of the 3DoF in the intrinsic space during the synthesis of ‘iCub’ can also be clearly seen. Panel G shows a snapshot of iCub writing iCub in the simulator environment. In the simulator, iCub creates shapes with its index finger and the trajectory traced is marked using static boxes (green color). The front view and the left camera view of the final solution is seen. Note also that the inherent modularity in the shape synthesis architecture allows effortless portability to other body chains (like 3 DoF arm, iCub simulator or any other robot). In other words, the learnt virtual trajectory can serve as an attractor for any body chain and the PMP relaxation hand guarantees the synthesis of motor commands to transform the virtual trajectory to real trajectory created by the concerned body chain.

Figure 24. Panels A-D show the target shape (resembling a side view of Gandhi), its AVP, the set of shapes generated during learning trials and the final solution. Before performing the experiments using iCub we also conducted a series of simulations of synthesis of composite shapes using a 3DoF planar arm (in Matlab) and on the iCub simulator. Panel F shows the synthesis of the character string ‘iCub’ using a 3DoF planar arm. In this case, the motion of the 3DoF in the intrinsic space during the synthesis of ‘iCub’ can also be clearly seen. Panel G shows a snapshot of iCub writing iCub in the simulator environment. In the simulator, iCub creates shapes with its index finger and the trajectory traced is marked using static boxes (green color). The front view and the left camera view of the final solution is seen. Note also that the inherent modularity in the shape synthesis architecture allows effortless portability to other body chains (like 3 DoF arm, iCub simulator or any other robot). In other words, the learnt virtual trajectory can serve as an attractor for any body chain and the PMP relaxation hand guarantees the synthesis of motor commands to transform the virtual trajectory to real trajectory created by the concerned body chain.

Figure 25 presents an assorted collection of drawings made by iCub.
4. Discussion: Beyond Shapes and Shaping

The journey so far: Shapes are ubiquitous in our perceptions and actions. Simply stating, seeing and doing meet at the boundaries of a shape. To have a clear understanding of both seeing and doing, a clear understanding of dual operations of shape perception and synthesis is critical. Using the scenario of gradually teaching iCub to draw, a minimal architecture that intricately couples the complementary operations of shape perception/synthesis was presented in this article. The architecture further encompasses the core loop necessary for imitation learning i.e. complex set of mechanisms necessary to swiftly map the observed movement of a teacher to the motor apparatus of the student (specifically teacher’s end effector trajectories). To achieve this objective, in this article we gradually traversed
through a series of subsystems (each realizing a different transformation) finally culminating in
drawings attempted by iCub. From the results presented in section 2 and the experiments that follow,
it is evident that using the small set of shape features derived using Catastrophe theory, it is possible
to describe the shape of any demonstrated trajectory, line diagram or sketch in general. A diverse
range of complex forms (including characters in several worlds’ scripts) emerge through various
combinations of these finite set of shape atoms. Inversely from an action generation perspective, since
more complex shapes could be ‘decomposed’ into combinations of these primitives, the actions needed
to synthesize them could be ‘composed’ using combinations of the corresponding ‘learnt’ primitive
actions. In this way, previously gained experience can be efficiently bootstrapped in newer tasks, for
example, motor knowledge gained in drawing ‘C’ and ‘U’ is exploited while drawing ‘S’. This further can
be exploited in creating shapes like ‘8’ or a double helix and so on. Maximum effort in terms of motor
exploration is applied to during the initial phases to learn the basics. During the synthesis of more
complex shapes, composition and refinement of previous knowledge takes the front stage, exploration
being reduced to a handful of mental trials. Hence, learning in humanoids through mental simulation
of action and duly aided by self evaluations of performance is another core idea employed in the
proposed architecture. This becomes possible because of two reasons:

1) Action generation networks created using PMP naturally form forward/inverse models. Hence
while a trajectory of motor commands are synthesized at the intrinsic space, the forward model at
the same time predicts the consequence i.e. the evolving trajectory at the end effector, a crucial
piece of information for learning and improving motor performance. Simply, while the inverse
model is crucial for motor control, the forward model is crucial for motor learning, their
cooperation being a critical feature in our model.

2) Catastrophe theory analysis is used to arrive at the abstract representations of both the teacher’s
demonstration (AVP) and iCub’s self generated movement predicted by the forward model (AMP).
Direct comparison between AVP and AMP can be done in a systematic manner to evaluate the
score of the performance. At this point we diverge from imitation in a strict sense, considering that
what iCub learns to reproduce is the ‘Shape’ and not just the end effector trajectory. The
advantage is that based on the definition in the concrete motor goal (CMG), the same observed
trajectory can be recreated with a different scale, at a different location using a different end
effector/body chain.

*Virtual trajectories:* A central component that makes modularity and learning at this level possible
is the idea of learning to synthesize ‘virtual trajectories’ which then go on to couple with the
appropriate internal body model participating in the motor task. Generally speaking, the notion of a
‘goal’ is where the distinction between perception and action gets blurred i.e. goals may be thought as
both sensory and motor. Analogously, the notion of ‘virtual trajectory’ is where the distinction
between goal and action gets blurred. They are explicit enough to act as goals for the lower level
action generation systems and implicit enough to act as atomic components of higher level plans.
Virtual trajectories may be thought as a more detailed description of a higher level motor goal (like CMG, or a linguistic description of a task). They are simple to the level that learning them does not become complex. They are complex to the level that they contain sufficient information to trigger the synthesis of motor commands (in bodies of arbitrary redundancy and complexity like the 53 DoF iCub). Hence, virtual trajectories play a significant role in both action learning and action generation in our architecture. While section 3.2 dealt with the problem of how a range of virtual trajectories can be synthesized between a discrete set of critical points (represented in CMG) by modulating the stiffness and timing, section 3.4 dealt with the problem of learning to synthesize the correct virtual trajectory such that the shape created by iCub is similar to the shape shown by the teacher. As elaborated through the results presented in sections 3.3, 3.4 and 3.5 virtual trajectories are just created mentally during the action planning/learning phase and do not really exist in the physical space. They rather have a higher cognitive meaning in the sense that they act as an attractor to the internal body (body+tool) model involved in action generation and play a crucial role in deriving the motor commands needed to create the shape itself (The aqueduct analogy was often used to illustrate this or see figure 12 which shows both the virtual trajectory and the real trajectory). In this context, virtual trajectories also symbolize an end effector independent description of action. Once a correct virtual trajectory (or attractor) to synthesize a circle is learnt, the acquired motor knowledge can be used to draw a circle on a paper or run a circle on a football field. Additionally, as shown in section 3.5 this motor knowledge gained while drawing a circle can be exploited during the creation of more complex shapes either similar to a circle (like ellipses etc) or shapes in which circle is a local element (like a simple flower, face).

Imitation: The study on neural basis of imitation is in its first steps. The cognitive, social and cultural implications of imitation are well documented today (Rizzolatti and Arbib, 1998, Schaal et al 1999). Iacoboni (Iacoboni 2010) in his minimal neural architecture of imitation singles out three major subsystems involved in imitation: a brain region that codes an early visual description of the action to be imitated, a second region that codes the detailed motor specification of the action to be copied, and a third region that codes the goal of the imitated action. Neural signals predicting the sensory consequences of the planned imitative action are sent back to the brain region coding the early visual description of the imitated action, for monitoring purposes ("my planned action is like the one I have just seen"). Experimental evidence from numerous brain imaging studies (Perrett et al., 1994, Rizzolatti et al., 1996, Grafton et al., 1996, Rizzolatti et al., 2001, Iacoboni et al., 1999, Iacoboni et al., 2001, Koski et al., 2002) suggest that the inferior frontal mirror neurons which code the goal of the action to be imitated receive information about the visual description of the observed action from the superior temporal cortex and additional somatosensory information regarding the action to be imitated from the posterior parietal mirror neurons. Efferent copies of motor commands providing the predicted sensory consequences of the planned imitative actions are sent back to STS where a matching process between the visual description of the action and the predicted sensory
consequences of the planned imitative actions takes place. If there is a good match, the imitative action is initiated; if there is a large error signal, the imitative motor plan is corrected until convergence is reached between the superior temporal description of the action and the description of the sensory consequences of the planned action. The use of forward model output for monitoring purposes is further validated by the fact that there is greater activity in superior temporal sulcus during imitation than in observation (if STS merely encodes visual description of actions its activity should be same during observation and imitation). Two possible explanations were proposed a) this increased activity may be due to increased attention to visual stimulus and b) due to efferent copies of motor commands originating from the fronto-parietal mirror areas for monitoring purposes. FMRI studies on specular vs anatomical imitation studies confirm that the predictive forward model hypothesis is indeed true (Iacoboni et al 2001). It is interesting to note that the proposed computational machinery clearly resonates with all these findings, the AVP coding for the early visual description of the action to be imitated, virtual trajectory coding for a detailed motor representation necessary for action generation, and the forward model outputs of PMP being used for monitoring purposes in the form of the AMP.

**Future Upgradations/Missing features:** We note here that the predicted forward model output is uncorrupted by the effects of execution, hence the drawings made by iCub (Figures 20, 25) are not as smooth as their predicted forward model counterparts (Figure 19, 22, 24). This discrepancy is because of two reasons: a) There is a small latency while buffering the motor command set (7000 X 10 matrix) to the actuators; b) Even though the trajectory is created in X-Z plane, the PMP relaxation takes place in 3 dimensional end effector space, with the additional constraint of keeping motion along Y-axis almost zero. However as we saw in figure 12 (panel b) there are slight inaccuracies (in the range of few mm) during which the paint brush pushes the drawing board causing inaccuracies in the drawings. On the other hand, the drawing board is not really vertical and has a small inclination due to which also the same second order effects are caused. At present, we do not control these second order effects. However, with the incorporation of touch and force sensing in iCub, we hope to deal with these issues in future versions of this architecture. Despite these inaccuracies, we can clearly see that overall essence of the shape is preserved in all drawings.

The Perception and Synthesis of ‘Shape’: Finally, the computational scheme proposed in this paper opens a range of new avenues in the investigation of perception and synthesis of ‘shape’ in general. The first task currently underway is the extension of the catastrophe theory framework for classification of shapes of 3D volumes (building on the work of Koenderink et al, 1986) and the extension of PMP framework to iCub’s fingers. Presently, PMP is being employed to coordinate the left arm-waist-right arm chain of iCub. Using the shape information derived from CT, geometric information of the object of interest (like, length, width, orientation etc) already available from the 3D reconstruction system, and PMP for action generation, we hope to build up the existing architecture for more subtle manipulations tasks, learning affordances of objects in moderately unstructured worlds.
physical cognition (trap tube paradigm, Tomasello et al) are under consideration to be included in the next ‘curriculum’ for iCub. A related subtask is imitation of outcomes, in addition to end effector trajectories (in tasks like object possession etc).

**Neuromotor rehabilitation**: In the context of the scenario of teaching iCub to draw presented in this paper, an interesting question to ask is whether the inverse of this is possible? i.e., from the perspective of neuromotor rehabilitation, how can robots provide assistance while teaching subjects or patients to perform motor actions (like handwriting, drawing). Note that the inverse scenario makes it possible to investigate motor learning as it occurs in human subjects. To investigate this issue, we have ported the proposed computational framework to the bimanual manipulandum (braccio di ferro, ) and the first set of experiments of teaching subjects to draw ‘shapes’ with their non-dominant hand (coupled to the BDF) is underway. An assistance module that optimally regulates assistance based on the performance of the student is being designed. With the BDF we are also investigating a three way interaction scenario between expert-BDF-student (expert and student coupled to the either arms of the manipulandum) during handwriting learning experiments. The goal for the robot here is to acquire an internal model of the training session (case histories) and use this knowledge to intelligently regulate assistance to the trainee when the expert is disconnected in the later stages.

**Shape of a ‘Signal’**: The third question we are investigating in the context of shape perception/synthesis is about the possibility of characterizing shapes of ‘signals’ in general independent of the sensory modality through which they are sensed (visual, auditory, haptic). Does multimodal sensory fusion partially result due to resonance between shape critical points computed different sensory modalities? For example it is well known from experience that certain forms of music resonate well with certain forms of dance or even the existence of numerous metaphors that connect different sensory modalities like ‘chatter cheese is sharp’. That humans are very good at forming crossmodal synesthetic abstractions has been known right from the early experiments of Kohler (Booba-Kiki effect,). When Kohler showed subjects two shapes, one a spiky angular shape and other a bulgy rounded shape and asked subjects to determine which one was ‘booba’ and which one was ‘kiki’, 98% people named the spiky shape as Kiki and the bulgy one as Booba. According to neuroscientist VS Ramachandran, the kiki visual shape has a sharp inflection and the sound ‘kiki’ represented in in the hearing centers of your brain, also has a sharp sudden inflection. Hence the brain performs a cross-modal abstraction, recognizing the common property of jaggedness, extracting it, and so reaching the conclusion that they are both ‘kiki’. The problem we are trying to address is the development of a formal framework that will allow autonomous robots to perform such cross-modal abstractions and multimodal sensory representations grounded in its sensory motor experiences. Recent results from sensory substitution (hearing to seeing for the blind, see ) also substantiate the primacy of ‘Shape’ being a central element in crossmodal mapping. We are also experimentally investigating the resonance between ‘shapes’ computed in different sensory modalities through a
series experiments using the VAICON motion capture system where the subject has to perform movements that he thinks resonates with the musical chord he hears.

*iCubArt and iCubthought*: How do children learn about the star, a house, a face, a flower or their own names and who they are? How are concepts formed and what are the contents that associate to a certain concept? Mainly, its the form of the concerned object i.e its shape, its geometric characteristics, its utility value and how you can ‘Act’ on that object. Since we get most of this information form the various subsystems presented in this article, can we then begin teaching iCub to heteroassociate linguistic labels (or symbols) to atleast some simple objects (and 2D sketches of them to begin with)? Will iCub really know who the character string ‘iCub’ that it scribbles on the drawing board really is? Will the primitive iCubArt presented in this paper be ever driven by iCubthought? Future research will seek to answer and realize some of these questions.

Acknowledgement

This research was conducted under the framework of EU FP7 projects iTalk and Humour. The authors thank the European commission for the sustained financial support and encouragement.
Appendix 1. Terminal attractor dynamics of PMP-networks

Intuitively, terminal attractors may be thought as a deadline for a grant proposal submission. A month before the deadline, the temporal pressure is less intensive, but goes to a peak the night before the deadline, and diverges afterwards. This feature is evident in the ‘shape’ of the time base generator function $\Gamma$ in figure A1 (and also figures 8, 9, 12, 19). They basically allow controlling the timing of the relaxation of a non linear dynamical system to equilibrium (without the need of a clock). Also note in the ‘deadline’ analogy that terminal attractors act under the presence of a goal (like deadlines are for specific goals). If the goal is reached before the deadline, then there is no temporal pressure (in equation A1, this occurs when $X = X_T$, so $F = 0$, so goal is reached). So it’s like a time varying gain that increases in magnitude as the deadline for the goal approaches. Formally, terminal attractor concept can be incorporated in the VTGS and PMP framework as follows:

Summarizing, a PMP-network is described by the following set of non-linear dynamics equations:

\[
\begin{cases}
F = K_{ext} (x_T - x) \\
T = J^T F \\
\dot{q} = A_{uu} T \\
\dot{x} = \Gamma(t) J \dot{q} \\
x(t) = \int_{t_0}^{t} \dot{x} \, dt 
\end{cases}
\]  
(A1)

In order to demonstrate that in this way the target is reached after a time equal to $\tau$ (the duration of the TBG) 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 linear bound:

$\gamma_{\min} z < f(z) < \gamma_{\max} z$  
(A2)

where $\gamma_{\min}$, $\gamma_{\max}$ are two positive constants. By denoting with $\gamma$ any value inside the $\gamma_{\min} \rightarrow \gamma_{\max}$ interval, we can write the following equation:

\[
\frac{dz}{dt} = -\frac{d\xi / dt}{1 - \xi} \gamma z 
\]  
(A3)

from which we can eliminate time

\[
\frac{dz}{d\xi} = -\frac{\gamma z}{1 - \xi} 
\]  
(A4)
The solution of this equation is then given by:

$$z(t) = z_0 (1 - \xi(t))^\gamma$$  \hspace{1cm} (A5)

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. Since this applies to both limits of the bound we can write the following bound:

$$z_0 (1 - \xi(t))^{\gamma_\text{min}} < z(t) < z_0 (1 - \xi(t))^{\gamma_\text{max}}$$  \hspace{1cm} (A6)

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 (figure A1) but the terminal attractor property of the model is maintained for a wide range of values of $\gamma$. 

**Figure A1.** Time-base generator (TBG) for terminal attractor dynamics - $\Gamma(t)$ - obtained from a minimum jerk time function - $\xi(t)$ - with assigned duration $\tau$. 