Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

Proposal for Thesis Research in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

Title: Manipulating Machines: Designing Robots to Grasp Our World

Submitted by: Aaron Edsinger
MIT CSAIL
32 Vassar Street, 32-380
Cambridge, MA 02139

(Signature of Author)

Date of submission: January 9, 2005

Expected Date of Completion: June 2006

Laboratory: Computer Science and Artificial Intelligence Lab (CSAIL)

Brief Statement of the Problem:

Manipulation requires anticipatory actions. We preshape our grasp before making contact with an object and stiffen our arm in anticipation of lifting a heavy one. These actions precondition the manipulation engagement to minimize the effects of short timescale dynamics. The goal of the research proposed here is to contribute an approach to robot manipulation encompassing three principal components: force sensing and compliant manipulation in unstructured environments, a decentralized, behavior based cognitive architecture, and the integration of anticipatory sensorimotor cues into the robot's behavioral repertoire. A control architecture, pARC*, is proposed as a framework for incorporating anticipatory information into a behavior based decomposition. The approach will be demonstrated through a series of manipulation scenarios conducted on a new* 29 *degree-of-freedom force controlled humanoid robot named* Domo.

Contents

1 Introduction **4**

1.1 Research Components 5

1.2 Work Milestones 6

1.3 Roadmap 6

2 Compliant and Force Sensitive Manipulators **7**

2.1 Force Sensing Compliant and Series Elastic Actuators 7

2.2 Virtual Model Control 9

3 Behavior Based Decomposition **10**

3.1 Building Artificial Creatures 11

3.2 Review: Behavior Based Manipulation 11

3.3 The Components of Dexterous Manipulation 13

4 Implicit Predictive Models **13**

4.1 Efference Copy 14

4.2 Mataric's Navigation and Landmarks 15

4.3 The *pARC* Framework 16

4.3.1 *pARC* Terminology 16

4.3.2 Predictive and Behavior Layers 17

4.3.3 Signals, Wires, and Gates 18

4.3.4 Predictive and Behavior Kernels 20

4.3.5 *pARC* Example: Reaching to a Target 21

4.3.6 *pARC* Detailed Example: Hand Localization 23

5 Manipulation Scenarios **24**

5.1 Developing a Body Schema 25

5.2 Playing Tether Ball 27

5.3 Playing Karate Sticks 29

5.4 Putting Away Toys 30

6 Milestones and Timeline **31**

A The Robot Platform **37**

A.1 Head 37

A.2 Arms 38

A.3 Hands 39

A.4 The Sensorimotor System 40

A.4.1 Physical Layer 41

A.4.2 DSP Layer 41

A.4.3 Sensorimotor Abstraction Layer 42

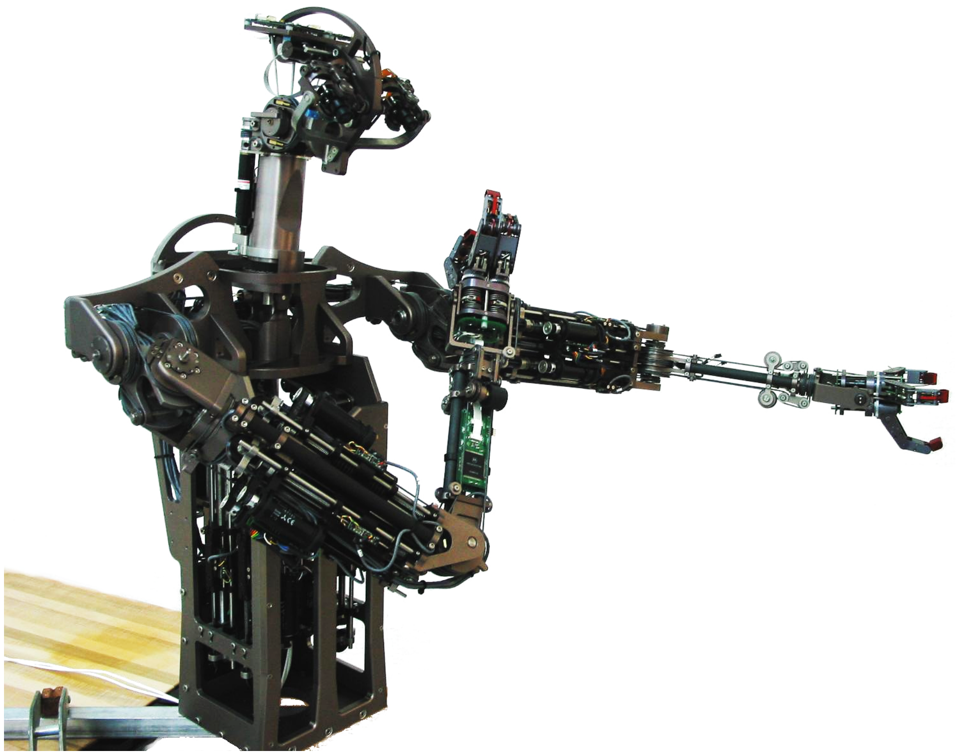


Figure 1: *The manipulation platform* Domo *being developed for this work has 29 active degrees of freedom (DOF), 58 proprioceptive sensors, and 24 tactile sensors. Of these, 22 DOF use force controlled and compliant actuators. There are two six DOF force controlled arms, two four DOF force controlled hands, a two DOF force controlled neck, and a seven DOF active vision head. The real-time sensorimotor system is managed by an embedded network of five DSP controllers. The vision system includes two FireWire CCD cameras. The cognitive system runs on a small, networked cluster of PCs.*

Abstract

Manipulation requires anticipatory actions. We preshape our grasp before making contact with an object and stiffen our arm in anticipation of lifting a heavy one. These actions precondition the manipulation engagement to minimize the effects of short timescale dynamics. The goal of the research proposed here is to contribute an approach to robot manipulation encompassing three principal components: force sensing and compliant manipulation in unstructured environments, a decentralized, behavior based cognitive architecture, and the integration of anticipatory sensorimotor cues into the robot's behavioral repertoire. A control architecture, pARC, *is proposed as a framework for incorporating anticipatory information into a behavior based decomposition. The approach will be demonstrated through a series of manipulation scenarios conducted on a new* 29 *degree-of-freedom force controlled humanoid robot named* Domo.

1 Introduction

Today's robots are not able to manipulate objects with the skill of even a small child. For robots to gain general utility in areas such as space exploration, small-parts assembly, agriculture, and even in our homes, they must be able to intelligently manipulate unknown objects in unstructured environments. Even a dog can turn a bone about with two clumsy paws in order to gain a better approach for gnawing. The Osprey, or fish hawk, has a 5 DOF foot which it uses to capture prey with remarkable dexterity [49]. These animals exhibit manipulation abilities not yet attained by robots.

Recent successes have been seen with robots that can navigate unstructured environments. These robots, such as the Mars Sojourner [24], use a behavior based architecture to accommodate a dynamic and unknown environment. However, these architectures haven't been as succesful in manipulation. We maintain that navigation and manipulation are fundamentally different in the timescales involved. By example, consider the classic inverted pendulum experiment of balancing a stick on the tip of your finger. A purely reactive controller would sense the current angle of the stick and move the finger in the appropriate direction. This controller works for a long stick where the timescale of the pendulum dynamics is long compared to the timescale of the physical dynamics of the motor system. As the stick gets shorter, the pendulum timescale becomes shorter and the reactive controller will fail. If the controller can anticipate the future sensorimotor state of the system in a feedforward term, it can remain stable for shorter and shorter timescales.

Manipulation is not simply a matter of building stable controllers, but the physical dynamics involved require anticipatory actions where navigation often can take a purely reactive approach. Correcting a grasp to prevent dropping an object requires tens of millisecond timescale adjustments, navigating down a cluttered corridor requires adjustments on the timescale of hundreds of milliseconds and seconds. Given ample computation, a sense-compute-act control loop could always be made fast enough to accomodate shorter timescales. However, if the physical dynamics of the motor system are fundamentally of a longer timescale than the robot-environment dynamics, anticipatory actions are required. Anticipatory actions advantageously bias the robot-environment dynamics in advance. They provide robustness to disturbances to an on-going behavior through feedforward control. They predict the when and where of future sensorimotor cues based on historical sensorimotor experiences. We preshape our grasp before making contact with an object and stiffen our arm in anticipation of lifting a heavy one. These actions precondition the manipulation engagement to minimize the effects of short timescale dynamics. For example, a correct grasp preshape lessens our dependence on minute grasp adjustments.

The goal of the proposed research is to contribute a novel approach to robot manipulation in unstructured environments. The approach is centered on integrating compliant and force sensitive manipulators into a behavior based architecture that supports anticipatory sensorimotor models.

A new 29 degree-of-freedom (DOF) upper-torso humanoid robot named *Domo* has been developed in order to investigate this approach. *Domo* is pictured in Figure 1. The robot exhibits different physical characteristics than typically found in manipulators. Traditional manipulators exhibit high stiffness and often use force-sensing load cells at the wrist and shoulder to deduce the forces acting on the arm. These manipulators can precisely control the position of each actuator but cannot directly control the force output. *Domo* exhibits reasonably high compliance in its joints and high fidelity force control at each joint. Its manipulator characteristics are analogous to human manipulators which are very good at controlling forces, but relatively poor at controlling joint position [20].

Domo is not designed to emulate the dexterity and sensing of humans. The robot's hands each have only 4 DOF, 12 tactile sensors, and are limited compared to a human hand. Traditional control methods used in manipulation are not well suited to the platform. Well defined models of objects have little value when using a manipulator without precise position control. However, the robot hardware developed does allow investigation of fundametal research issues surrounding manipulation. Namely, how to integrate visual perception, manipulator forces, and anticipatory behaviors in a tightly coupled interaction with the world.

We contend that the manipulation efficacy exhibited by a dog is not necessarily the result of finding optimal relations between a model of its paw (which has little dexterity) and a model of the bone (which is poorly approximated by a generic model). We view the dog as engaged in a tightly coupled interaction with the bone where it is modulating many different force based behaviors based on a stream of visual, tactile, and olfactory information. The dog's internal model of this interaction, if it is even correct to use the term 'model', is of predicted sensory consequences of the behaviors. Pushing the paw down on the bone with greater force results in less visual motion of the bone (due to increased friction between the bone and the ground). A hypothetical robotic dog doesn't need to construct a model of the bone, the paw, the ground, and the frictional forces. Instead it simply must know to increase the paw force when the optic flow of the bone is too large.

Clearly there are circumstances where it is preferable to take a traditional model based approach, and the proposed approach to manipulation will likely be imprecise and coarse at first. However, it should be demonstrably better in circumstances where traditional approaches fall short, such as in real world, dynamic and unstructured environments.

1.1 Research Components

The proposed approach to robot manipulation has three principal research components:

1. **Compliant and Force Sensing Manipulators**: Robots working in unstructured environments depend on unreliable perceptual features to guide their manipulators. The motor system of the robot should be capable of directly controlling the forces it exerts on the world. This allows a force-based decomposition of manipulation tasks and it allows the robot to safely move its manipulators when the precise location of objects in the environment are not known. The motor system should also be reasonably compliant, providing robustness to unexpected collisions and passive adaptation to unknown object features.

2. **Behavior Based Decomposition**: Manipulation requires a tight coupling between the object being manipulated, the robot sensory system, and the robot motor system. A behavior based decomposition divides the computational organization of a robot into an incremental series of layers. A layer exhibits externally observable coherent behavior. Successive layers are added incrementally and each results in improved or expanded coherent behaviors. A behavior based decomposition applied to a force controlled manipulator provides a tight, reactive coupling between the manipulated object and the robot. It forgoes the need for explicit models of the robot and the objects being manipulated. It also provides a systematic framework to integrate many low-level force sensitive behaviors in such a manner that higher-level, task oriented behaviors can be intuitively specified.

3. **Implicit Predictive Models**: The human motor system is characterized by large time delays. Consequently our cerebellum very likely functions as a predictive controller, anticipating the sensory consequences of movements before they occur and cancelling out self-generated

sensory stimuli [51]. A predictive model serves to anticipate sensory or motor consequences of the current sensory and motor state. Such a model is constructed from a history of sensorimotor experiences. An *implicit* predictive model fits into a behavior based decomposition. It lacks the explicit, centralized representation that is implied by the common use of the term model. Instead, the model is distributed across the layers of the behavior based framework. As behavior layers are incrementally added to the robot controller, the implicit predictive model is also expanded and improved upon. A common criticism of strictly reactive, behavior based architectures is that they lack state and the ability to exploit experience. Implicit predictive models can serve as means to integrate state into a behavior based decomposition. These models can then be used to:

(a) adapt behaviors, as in the case of modulating arm stiffness.

(b) generate behaviors, as in the case of grasp preshaping.

(c) amplify salient sensory signals. For example, a predictive model of self-generated optic flow can be used to amplify the optic flow generated by the external world.

(d) reduce noise in sensory signals. For example, a visual object tracker will often lose a tracked object for a few frames. Objects in the real world don't suddenly disappear and then reappear. A predictive model can be used to anticipate the continuation of the object trajectory despite noise in the tracker.

1.2 Work Milestones

There are three primary milestones to our work:

1. **Design and construction of the robot platform *Domo*.** The design is centered on providing a robust platform which can support rich, prolonged sensorimotor experiences during manipulation engagements.

2. **Implementation of a set of primitive behaviors for the robot**. These bootstrap the system to engage in basic exploratory manipulation acts. This includes development of force controllers for the manipulators, grasping postures for the hand, simple visual feature detectors, and an attentional system to guide the robot towards salient stimuli.

3. **Development of an anticipatory control architecture and its application to *Domo* through a series of manipulation scenarios**. The architecture is named *pARC*, short for *Predictive Architecture. pARC* serves as a tool to integrate new behaviors into the robot over time. *Domo*'s manipulation competency will be improved incrementally by expanding and refining the robot's behaviors, predictive models, and visual percepts. At each developmental stage, the robot will exhibit coherent and integrated behaviors. The complexity of the manipulation scenarios conducted is also increased at each developmental stage.

1.3 Roadmap

In the remainder of this document we elaborate on our principal research components and work milestones.

Section 2 describes the compliant and force sensing manipulators designed specifically for our research approach and presents a control methodology for them. Section 3 reviews behavior based approaches to manipulation and provides a behavior based decomposition to be used in this work. Section 4 expands the notion of implicit predictive models and formulates the *pARC* framework. A series of manipulation scenarios are described in Section 5. Section 6 outlines a timeline for the implementation of the proposed work. The mechanical and software design of *Domo* is described in Appendix A.

2 Compliant and Force Sensitive Manipulators

Humans are very good at controlling manipulator forces, but relatively poor at controlling joint position, as demonstrated by Kawato's [20] study of arm stiffness during multi-joint movements. Joint torque in the human arm is generated by an imbalance of tension between antagonist and agonist muscles which have inherently spring-like properties. Equilibrium-point control (EPC)[38] is an influential model for arm movement which posits that the spring-like viscoelastic properties of muscles provide mechanical stability for control. Joint posture and joint stiffness is maintained by modulating the tension of the agonist/antagonist muscle pair. EPC provides a method of arm control which does not require computing a model of the complex dynamics of the arm.

EPC is only part of the story of human arm control. However the notion that spring and damper like qualities in the manipulator can be exploited for stable and simplified control can be applied to robot limb control as well.

Based on previous work done on the robots Cog [9] and Spring Flamingo [46], we have built robot arms and hands specifically designed to support physical and simulated spring-damper systems. Our supposition is that, in the context of manipulation, compliant and force sensing manipulators can significantly modify the shape of the problem space into one that is simpler and more intuitive. These manipulators allow a force-based decomposition of manipulation tasks, allow the robot to safely move when the location of objects in the environment are not well known, and provide a robustness to unexpected collisions.

In this section we describe two related actuators, the Series Elastic Actuator[45] (SEA) and the Force Sensing Compliant Actuator (FSCA) [13]. We have developed the FSCA as an alternative to the SEA when very compact force sensing is required. We also describe an existing method of controlling these actuators called Virtual Model Control [26].

2.1 Force Sensing Compliant and Series Elastic Actuators

The 20 actuators in *Domo*'s arms and hands and the 2 actuators in the neck utilize series elasticity to provide force sensing. We place a spring inline with the motor at each joint. We can then measure the deflection of this spring with a potentiometer and know the force output by using Hooke's law ($F = -kx$ where k is the spring constant and x is the spring displacement). We apply this idea to two actuator configurations, as shown in Figure 2. The SEA places the spring between the motor and the load, while the FSCA places the spring between the motor housing and the chassis ground. There are several advantages to these actuators:

1. The spring and potentiometer provide a mechanically simple method of force sensing.
2. Force control stability is improved when intermittent contact with hard surfaces is made. This is an important attribute for manipulation in unknown environments.

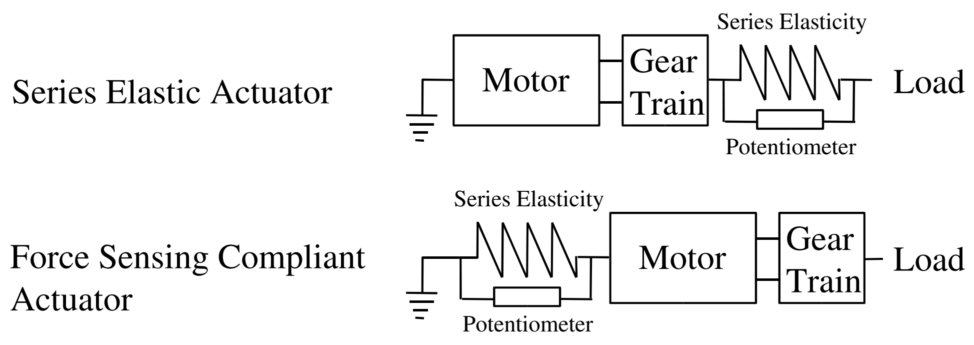


Figure 2: *Block diagram of the Series Elastic Actuator and the Force Sensing Compliant Actuator. The SEA places an elastic spring element between the motor output and the load. The FSCA places the spring element between the motor housing and the chassis ground. SEAs are used in* Domo*'s arms and neck. FSCAs are used in* Domo*'s hands.*

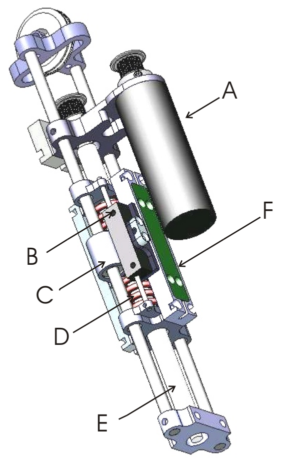
3. Shock tolerance is improved. The use of an $N : 1$ geartrain increases the reflected inertia at the motor output by N^2. This results in shock loads creating high forces on the gear teeth. The series elastic component serves as a mechanical filter of the high bandwidth forces, reducing the potential of damage to the gears.

4. The dynamic effects of the motor inertia and geartrain friction can be actively cancelled by closing a control loop around the sensed force. Consequently, we can create a highly backdrivable actuator with low-grade components.

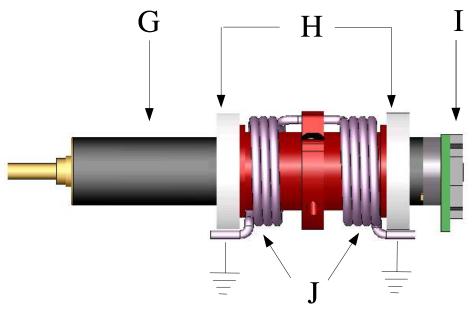
5. The actuators exhibit passive compliance at high frequencies. Traditional force controlled actuators exhibit a large impedance at high frequencies because the motor response is insufficient to react at this timescale. In an SEA, the impedance of the elastic element dominates at high frequencies.

The overall passive compliance exhibited by the SEA or FSCA is determined by the spring stiffness. If we consider that an external force applied to the actuator can only be counteracted by the spring, then we see that the mechanical impedance of the system is defined by that of the springs. The low impedance of the springs adversely affects the reaction speed, or bandwidth, of the system. For robot tasks achieved at a roughly human level bandwidth, this adverse effect is not large.

The differences between the FSCA and the SEA provide distinct advantages and disadvantages. The SEA, as pictured in Figure 3, uses a linear ballscrew and a cable transmission. The ballscrew provides greater efficiency and shock tolerance than a gearhead. The SEA is limited by the travel range of the ballscrew which creates packaging difficulties. The linear potentiometer must move with the motor output, precluding the use of continuous rotation configurations. In contrast, the FSCA can allow continuous rotation at the motor output as the potentiometer does not move with the motor. However, the elastic element is not between the load and the geartrain, decreasing the shock tolerance.



(1) SEA



(2) FSCA

Figure 3: ***(1)*** *Model of the cable-drive SEA. A brushless DC motor (A) imparts a linear motion to the inner drive carriage (C) through a precision ballscrew (E). The inner drive carriage transmits motion to the outer drive carriage (F) through two precompressed die springs (D). The deflection of the springs is measured with a linear potentiometer (B).* ***(2)*** *A simplified view of the FSCA. Two bearings (H) support the motor (G). The motor is attached to an external frame (ground) through two torsion springs (J). As the motor exerts a torque on a load, a deflection of the springs is created. This deflection is read by the torque sensing potentiometer (I).*

2.2 Virtual Model Control

The force sensing actuators in *Domo*'s arms and hands will be controlled using Virtual Model Control (VMC). VMC is an intuitive control methodology in the same category as EPC, as well as operational space control, developed by Kahtib [30]. It was developed initially for biped robots [26] which exhibited very naturalistic walking gaits using SEAs.

VMC represents the control problem in terms of physical metaphors about which we have a good natural intuition: springs and dampers. Virtual springs and dampers are simulated between the robot's links and between the robot and the external world. This allows force controlled movement of the manipulator with only a forward kinematic model. Dynamic models of the arm are not required.

The key idea of VMC is to add control in parallel with the natural dynamics of the arm. When we lift a milk jug into the refrigerator, we exploit the pendulum dynamics of the system to give the jug a heave. Traditional control methods override the natural dynamics of the manipulator. Instead, the manipulator follows a prescribed trajectory in joint space. A force sensing and compliant manipulator, however, can allow the natural dynamics to be exploited. Its trajectory is the composite of the natural dynamics interacting with a set of virtual springs and with the environment.

The robot Cog demonstrated exploitation of natural dynamics in a number of rhythmic tasks, including sawing, hammering, and playing with a Slinky [58]. With VMC, we add layers of springs and dampers in parallel with the natural dynamics. Figure 4 illustrates an example of applying VMC to safely guide *Domo*'s arm to reach towards a target. In this illustration, virtual spring-

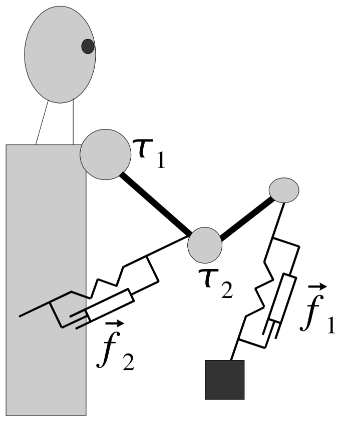


Figure 4: *A simple illustration of Virtual Model Control of an arm. Virtual springs and dampers are attached between the robot body and the arm ($\vec{f_2}$) and between the end effector and the reach target ($\vec{f_1}$). With a forward kinematic model we can determine the arm jacobian, J. These instantaneous forces can be mapped to desired joint torques: $\vec{\tau} = J^T F$*

dampers are used of the form $f = -k_s x + k_d \dot{x}$, where x is the spring displacement. Each spring-damper yields instantaneous forces on the arm, $\vec{f_1}$ and $\vec{f_2}$. The force $\vec{f_1}$ guides the arm towards a target. The force $\vec{f_2}$ repels the elbow of the arm away from the body to avoid collisions. With a forward kinematic model we can determine the arm jacobian, J, which relates the velocity of the end-effector (or elbow) to the joint angular velocities. The end-effector force then relates to the joint torque by $\vec{\tau} = J^T F$ [10].

The joint torques $\vec{\tau}$ can be commanded to the SEA with a simple PID controller, simulating the virtual springs. The stiffness of the arm can be controlled dynamically by modifying k_s, and sets of springs can be add incrementally, and in parallel, to the natural dynamics of the arm. Additionally, non-linear springs may be simulated to create specific spring behaviors.

3 Behavior Based Decomposition

Behavior based control architectures have proven very successful for navigating mobile robots in non-laboratory environments. Architectures of this class have been used on the Mars Sojourner [24], the Packbot robot deployed for military operations, and in the Roomba household vacuum cleaner. The control architecture in these robots decomposes the navigation problem into a set of interacting, layered behaviors.

We maintain that today, much of robot manipulation in unstructured environments is similar to where robot navigation was 20 years ago. For example, the Stanford Cart [42] built detailed models and plans at every step during navigation. It moved one meter every ten to fifteen minutes, in lurches, and movement of natural shadows during this time would create inaccuracies in its internal model.

Similarly, much of the current work in robot manipulation uses quasistatic analysis, where detailed static models are used to compute grasp stability, for example, at each time step. This is a classic *look-think-act* decomposition where the robot senses the environment, builds a detailed model of the world, computes the optimal action to take, and then executes the action.

Real world manipulation tasks involve unstructured and dynamic environments. In this setting, explicit models and plans are unreliable. A *look-think-act* approach to manipulation, at least at the lower levels of control, renders the robot unresponsive. For example, in the time it takes a robot to build a model and compute an action, a slipping object will likely have dropped from the robot's grasp. Manipulation is characterized by a high-bandwidth coupling between the manipulator forces and the object. A behavior based decomposition provides this coupling.

3.1 Building Artificial Creatures

A behavior based decomposition allows us to incrementally build the robot as an *artificial creature*. We use the term *creature* to suggest that a robot should, in principle, be left switched on to interact with the environment for extended periods of time. It should exhibit a coherent set of behaviors which are appropriately responsive to its environment. The robot should exist in the world as a (nearly) always-on entity, analogous to a living creature. The name is not, however, meant to imply the robot's relationship with biologically inspired models of robot design.

Currently, most humanoids are left to run only for the duration of an experiment. Our approach with *Domo* is to bootstrap the robot with a primitive set of exploratory behaviors that will generate structured sensorimotor patterns of activity. These sensorimotor patterns can then be used to build additional behaviors. At each point in the incremental construction of the robot controller, we impose the *artificial creature constraint*. This constraint, borrowing from Brooks [6], stipulates that the robot should exhibit:

1. **Coherence**: The interaction of many behaviors should appear outwardly coherent. This requires the appropriate switching of behaviors in response to a changing environment.
2. **Salience**: The robot should be responsive to salient perceptual stimuli. Saliency can be modulated by an attention system driven by a set of drives, as demonstrated with the robot Kismet [5].
3. **Adequacy**: The robot should generate behavior which achieves a set of prescribed goals. For *Domo*, these may be exploring its workspace, or grasping certain types of objects. These goals can be incrementally expanded over times.

The artificial creature constraint requires developing a method for integrating competing behaviors. The behavior selection problem is well studied and a variety of methods are available [5, 48]. The constraint also requires imparting the robot with a set of drives and a motivational system to attend to salient stimuli. We propose a system similar to those previously developed in our lab with Cog and Kismet.

3.2 Review: Behavior Based Manipulation

There is a wealth of literature on traditional approaches to manipulation. For an overview, see [35]. Work on behavior based decompositions of manipulation has been scarce, especially on real world humanoid robots. Unfortunately, this work is typically characterized by incomplete integration of the perceptual systems and motor behaviors. The complexity of the systems, or perhaps just the nature of research, hasn't allowed the humanoid platforms to be built as integrated, artificial creatures as described above. Here we review related work where a behavior based decomposition has been employed at least in part.

Some of the earliest work in behavior based manipulation was conducted by Brooks et al. with the robot Cog [9]. Cog, like our robot *Domo*, had two force controllable arms utilizing SEAs. It had a 7 DOF active vision head and a rudimentary force controlled gripper. The predominant work on the platform focused on active visual perception [15], multi-modal integration [2], and human imitation [50]. Williamson developed a set of rhythmic behaviors with the arms using neural oscillators [58]. However, the electromechanical robustness of the manipulators ultimately limited their utility in exploring the manipulation problem space. All of these systems were never integrated into a coherent framework.

Marjanovic [34] proposed the only truly integrative architecture for Cog. The proposed framework allows behavioral competencies to be embedded in a distributed network. The framework supports the incremental layering of new abilities and the ability to learn new behaviors by interacting with itself and the world. The learning is accomplished by autonomous generation of sensorimotor models of the robot's interaction with the world. Unfortunately, the system proved perhaps too general and only simple behaviors were learned in practice. Manipulation problems were never directly addressed. However, the framework does provide an example of an integrative approach to building behavior based robots.

One of the most thorough explorations of behavior based manipulation thus far has been achieved by Grupen et al. [44], in which an outline for a hierarchical framework for humanoid robot control is proposed. Their work is tested on a real humanoid platform, Dexter, which features 2 force sensing Whole Arm Manipulators (WAMS) with 7 DOF each, an active vision head, and 2 force sensing hands with 4 DOF each.

The Dexter project decomposes the robot controller into a set of control basis behaviors, each of which is a low-dimensional sensorimotor feedback controller. These behaviors are combined by projecting the control basis of one controller onto the nullspace of the other. Novel controllers can be learned with reinforcement learning techniques. For example, a grasping policy was learned which switches between two and three fingered grasps based on the state of the grasping interaction. They have also conducted work in incremental development of grasp controllers which do not require an a priori object model [21] and in learning haptic categories which can be used to associated visual and haptic cues with appropriate grasps [27].

The Sandini Lab has taken a developmental approach to humanoid robot manipulation, primarily with the robot Babybot [39, 43]. This robot has a single PUMA arm with coarse force control, a 16 DOF hand with only six actuators and passive compliance, and a 5 DOF active vision head. Their approach draws heavily on infant development and developmental psychology. Their approach utilizes development stages and non-model based control which fits into our notion of a behavior based manipulation system. Natale [43] proposes an actor-critic learning scheme for function approximation of sensorimotor activity during exploratory motions. This scheme utilizes a layered set of actor-critic modules which interact in a traditional behavior based architecture. They have also investigated tightly coupled visual and motor behaviors to learn about object affordances [14]. Knowledge about the object affordances is then exploited to drive goal-directed behavior.

3.3 The Components of Dexterous Manipulation

We propose that manipulation consists of nine different components [7]. These components can be concurrent, can occur multiple times during the manipulation engagement, and provide a high-level behavior based decomposition. Briefly, these components are:

1. **Deciding on actions.** A sequence of actions is determined based on the task at hand. In well characterized settings this sequence may be determined ahead of time; otherwise, it is generated by perceptually guided action selection mechanisms.
2. **Positioning sensors.** Sensors, such as a camera, need to be positioned to get appropriate "views" of the elements of the engagement. Positioning should occur as a result of a dynamically coupled loop between the sensor and the environment.
3. **Perception.** Perception continues throughout the engagement. Primarily the robot needs a good understanding of where things are and what objects with what properties are present before moving a manipulator in to engage.
4. **Placing body.** The pose of the robot body is adapted to allow an advantageous reach of the workspace and the ability to apply the required forces during the engagement.
5. **Grasping.** A generic dexterous hand must form a stable grip that is appropriate for any future force or transfer operations that are to be done with the object. This requires coordinating the many degrees of freedom in the hand and arm with visual and tactile perceptual streams.
6. **Force operations.** The central component of dexterous manipulation is the modulation of the interaction forces which occur between the manipulator and the object. The manipulator must apply appropriate forces and modify those forces in real time based on the response of the objects or material being manipulated.
7. **Transfer.** The manipulator transfers the object to a desired location, avoiding obstacles as necessary. This requires local knowledge of the environment.
8. **Disengaging.** The object is released from the grasp in the correct location and pose. This can be considered the inverse of grasping.
9. **Detecting failures.** The robot should detect when action has failed. This is a perceptual problem but requires feedback into the action selection process.

4 Implicit Predictive Models

Anticipatory actions can be generated with *implicit predictive models.* An implicit predictive model is a model of the robot's sensorimotor relationships based on prior historical experience. *Implicit* denotes that the model is distributed across the system. There is not a global model of the robot sensorimotor system and the environment. *Predictive* denotes possible use as forward model, where the future sensorimotor state is anticipated prior to occurrence.

Implicit predictive models are motivated by findings in neuroscience that the brain very likely uses distributed forward and inverse sensorimotor models to anticipate sensory consequences of motor actions. These models, commonly referred to as efference copy mechanisms, are reviewed below.

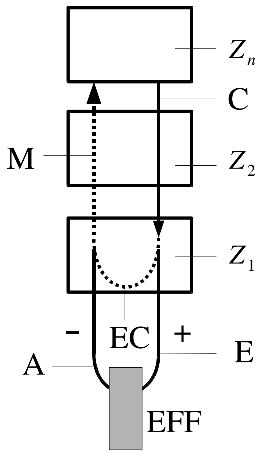


Figure 5: A reproduction of von Holst's original schematic of the reafference principle [57]. Descending brain centers $\mathbf{Z_n} - \mathbf{Z_1}$ have sensorimotor connections to a muscle effector, **EFF**. The action generated by motor command, **E**, generates the reafferent sensory signal **A**. The command **E** also generates in $\mathbf{Z_1}$ the efferent copy **EC**. The signal is subtracted from **A**. If **EC**, the predicted afferent signal, matches A, the actual afferent signal, then no ascending signal is transmitted. Otherwise, the difference is transmitted in signal **M**, which is a behaviorally relevant signal known as exafference.

4.1 Efference Copy

Efference copy mechanisms provide a creature with a means to distinguish between self-induced sensory signals and the behaviorally relevant signals generated by the external world. The notion of efference copy originates from experiments conducted by von Holst [57]. Through experiments with flies and the Mormyrid fish, von Holst developed the reafference principle, depicted in Figure 5. The principle postulates that a primary component in the organization of animal behavior is the generation of an expectation signal that predicts the sensory consequences of a motor action. Higher centers in the brain generate a movement command which produces a pattern of muscle activity in lower levels. This command is sent to the muscle and also generates the efference copy of the command through, perhaps, an inverse model of the sensorimotor relationship. The form and the role of the efference copy generation mechanism is still debated.

Blakemore et al. [51] have implicated the cerebellum in signalling the discrepancy between the predicted and actual sensory consequences of movements. Subjects controlled a robotic arm with their right arm to provide tactile stimulation to their left hand. Computer induced delays between the commanded arm movement and the predicted tactile stimulation resulted in cerebellar activity, corresponding to the signal M in von Holst's depiction in Figure 5.

Much of the work in efference copy has centered on developing neurally reasonable models of the mechanism, including models of time-delay compensation in song bird learning [56], grip force accommodation during bimanual manipulation [59], and the accommodation of variable object weights during lifting [18]. Kording and Wolpert have shown that the efference copy mechanism very likely incorporates a probabilistic model of sensorimotor relations [31]. It is also utilized to stabilize the dynamics and compensate for time-delays encountered during manipulation [12].

There has been notable focus on efference copy in human manipulation tasks, perhaps due to their suitability for experimental testing. There has been some work in applying the notion of efference copy to robotics. Datteri et al. [11] have formulated a predictive framework used to anticipate the visual trajectory of the end effector of an 8 DOF robot arm. Moller [41] has developed a framework for integrating prediction into a robot controller but hasn't demonstrated its application to a real robot. However, outside of gaze stabilization, there has been relatively little work in developing and applying predictive mechanisms to real robot controllers.

4.2 Mataric's Navigation and Landmarks

Mataric's early work with autonomous navigation and landmarks [36, 37] can be viewed as building implicit predictive models of the environment. The world is experientially encoded through structured activity in the world. Landmarks do not refer to an explicit model, but instead refer to sensorimotor relationships. Her work builds a map of the environment with spatial and temporal extent.

In *Catching Ourselves in the Act* [23], Hendriks-Jansen postulates that manipulation can be viewed as a generalized notion of a navigation and landmark building problem.

> The notions of navigation and landmark can be taken in a much wider sense than that connected with travelling through a landscape. One may think of an infant as learning to navigate the space within its reach by the use of its hands and eyes, of a piano player as learning to navigate the keyboard by performing situated patterns of activity in the form of scales...

Our approach is to use Mataric's navigation and landmark framework as a starting point for developing implicit predictive models in the context of manipulation. Where the navigation framework might embed relationships between wheel velocity and sonar readings, the manipulation framework may embed relations between interaction forces and arm trajectories. Before describing details of our approach, we first briefly describe Mataric's work with the robot Toto.

Toto is an omnidirectional three-wheeled base with 12 ultrasonic ranging sensors and a flux-gate compass. Using Brooks' subsumption architecture [8], Toto is capable for four high level behaviors: STROLL, AVOID, ALIGN, and CORRECT. These behaviors suffice to allow Toto to safely wander about and explore its world in a structured manner, following walls and avoiding obstacles.

By exploring its environment, Toto generates structured and temporally extended sensorimotor patterns. For example, if the robot repeatedly detects proximal objects on its right side while maintaining a stable compass bearing, then a counter is incremented indicating confidence in a 'right-wall' landmark. There is no explicit model of 'right-wall' in the robot. Instead, the notion arises from structured sensorimotor activity generated by the underlying behaviors.

Toto encounters a series of landmarks during its exploration and encodes these as nodes in a distributed graph. A map of the environment is built up over time during extended explorations. This map is not an explicit model of the world, but is a record of the structured sensorimotor experiences found in the world. Noisy sensors leads to uncertainty about the current robot state (location in the world) in model based approaches. Mataric's approach encodes the current robot state directly in terms of its sensory experiences, avoiding this type of uncertainty altogether.

Once a map is been built, Toto is able to navigate between landmarks on the map. First the robot needs to know its current location on the map. This is non-trivial because the map is an encoding of relative information, not absolute. Unfortunately, this issue is not adequately addressed in Mataric's work. Assuming that the current landmark is known however, then a graph path to a target landmark is calculated using a shortest-path algorithm. Toto can then traverse the landscape, depending on its low-level behaviors to handle local navigation, and using the computed graph-path to direct its overall course. We can also view Toto as having a predictive model of its environment. As the robot navigates, it can use its estimation of its location in the landmark graph to anticipate the sensorimotor experiences it will experience. For example, as the robot follows a right-hand wall and approaches a corner, it can predict the sensory pattern of a frontward wall landmark before reaching it.

Mataric's work with Toto provides, by example, an approach to developing implicit predictive models for manipulation, where:

1. Structured sensorimotor activity is generated by low-level manipulation behaviors.
2. A distributed representation of this activity embeds an experiential history of the robot's exploration of its environment.
3. This distributed representation, or implicit model, can be used to plan trajectories through the environment. It can also be uses to predict expected sensorimotor experiences as the robot executes a trajectory.

4.3 The *pARC* Framework

We propose the Predicitive Architecture, or *pARC*, as a general framework for integrating predictive information into behavior based systems. In the next sections we provide a description of the architectural primitives of *pARC* and show how they can be composed into a visually guided reach-to-target behavior.

4.3.1 *pARC* Terminology

- **Sensorimotor stream [y]**: A time varying stream of sensor and/or motor data. This may be raw sensor readings, motor commands, or a higher level representation such as a target trajectory in the image plane or the displacement of a virtual spring acting on the arm.
- **Saliency stream** [ϵ]: A measure of the *value* of an associated sensorimotor stream. The notion of value is dependent on the context of the sensorimotor stream use. For example, it may be a measure of the prediction error rate or the desirability of a particular manipulation object.
- **Signal** [$\mathbf{s^n}$]: A packet transmitted on a wire containing a time ordered set of n sensorimotor and saliency pairs.
- **Wire**: A conduit for signals. Signals are asynchronously clocked along a wire at each time-step. The signal value along a wire may vary, depending on the influence of gates acting on the wire.
- **Gate** [$\alpha, \beta, \mathbf{\Sigma}$]: A junction for a primary and secondary wire. Three gates, α, β, Σ, are described in the following sections. A primary wire transmits a signal s through a gate. A secondary wire can overwrite some or all of the signal passing through the gate, depending on the gate type.

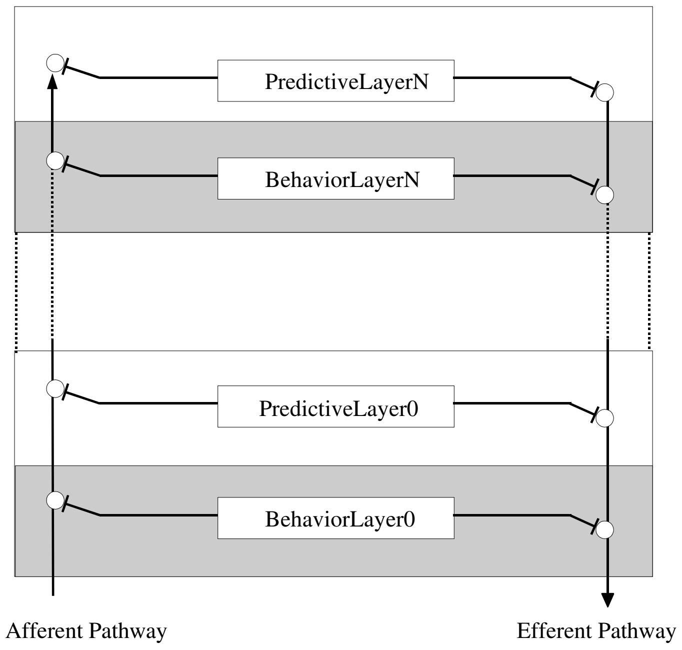


Figure 6: *The* pARC *framework takes a strongly layered approach. Alternating predictive and behavior layers represent developmental stages for the robot. Each incrementally improves the robot's behavioral repertoire and predictive models. At each developmental stage the robot exists as a coherent, embodied artificial creature. Two computational passes occur for each timestep in* pARC*. The first pass starts at BehaviorLayer0 and works upward to PredictiveLayerN. It transmits computed sensorimotor signals along the afferent pathway to higher layers for augmentation and refinement. The second pass starts at PredictiveLayerN and works downwards to BehaviorLayer0. It transmits computed sensorimotor signals along the efferent pathway to lower layers which can modulate and transform the higher level signals.*

- ***pARC* Kernel $[\mathbf{B}, \mathbf{P}]$**: A computational thread which computes a signal s^n given a set of input signals. For pedagogical purposes, we distinguish between behavior kernels, B, and predictive kernels, P. A behavior kernel computes a behavioral command while a predictive kernel computes the continuation of a sensorimotor stream forward in time. The kernel types are functionally identical in terms of their use an architectural primitive.
- **Behavior Layer**: A set of behavior kernels built on top of an existing predictive layer. The layer expands and refines the behaviors of lower layers.
- **Predictive Layer**: A set of predictive kernels built from the structured sensorimotor activity generated from a preceding behavior layer. The layer expands and refines the predictions of lower layers.

4.3.2 Predictive and Behavior Layers

A powerful feature of a layered control architecture is that it provides a framework for rudimentary behaviors to be augmented with more complex and refined behaviors over time. In *pARC* we take a strongly layered approach to building the robot's controller. As depicted in Figure 6, alternating predictive and behavior layers are added incrementally, representing the developmental stages of the robot. At each developmental stage the robot exists as a coherent, embodied artificial creature. Each layer augments and refines the robot's behavioral repertoire and predictive models. *pARC* utilizes two computational passes for each controller timestep. Referring to Figure 6, the first pass starts at *BehaviorLayer0* and works upward to *PredictiveLayerN*. It transmits computed sensorimotor signals along the afferent pathway to higher layers for augmentation and refinement. The second pass starts at *PredictiveLayerN* and works downwards to *BehaviorLayer0*. It transmits computed sensorimotor signals along the efferent pathway to lower layers which can modulate the higher level signals.

pARC implements the individual behavior and predictive kernels of each layer as a very lightweight thread utilizing a custom scheduler. Higher layers may suspend the threads of lower layers as an act of inhibition. Each thread communicates on the framework's afferent and/or efferent pathway. The afferent pathway allows higher layers to refine a particular sensorimotor prediction of a lower layer. The efferent pathway allows lower layers to have access to the refined prediction. However the lower layers do not require the higher layers in order to perform their computational task. In this way, we can treat each layer as an independent developmental stage yet provide a rich coupling between the layers. In principle, we can demonstrate the developmental stages of the robot by switching on the layers of *pARC* in real-time.

An implicit predictive model is distributed across multiple layers. Conceptually, we would like to start with a simple sensorimotor predictor with which to bootstrap the controller development. As behavior layers are added, the richness of the robot's interaction with the environment increases and perceptual features will also become more structured. We can then augment the initial sensorimotor prediction with a more refined estimate.

Layering also increases the robustness of the prediction. Predictive kernels may only be effective in subspaces of the sensorimotor workspace. A series of layered kernels, able to subsume control based on their saliency measure, can patch together an effective predictor which spans the entire workspace.

4.3.3 Signals, Wires, and Gates

pARC is a distributed system of many computational kernels which interact through signals, wires, and gates. A signal is a time varying packet of information transmitted on a wire. A gate manages the junction of two signals. These three architectural primitives are depicted in Figure 7.

A signal is typically denoted $s_j^{n_i}$, where j is the particular signal instance, n is the number of predictive time steps forward, and i is the value of n at a particular gate. For clarity, the subscripts and superscripts will be assumed implicit unless needed for illustrative purposes.

The superscript n is used to denote the number of predictive timesteps contained in a signal packet. Assuming that the robot's sensorimotor system is sampled into discrete time steps, y^n is the predicted continuation of the sensorimotor stream y for n steps into the future. We should note that often it may not be possible to compute the predicted continuation, or we won't need to, and $n = 0$.

A sensorimotor stream may be a combination of several sensory features or motor commands. For example, it could be the optic flow in an image and the commanded joint torques for the arm.

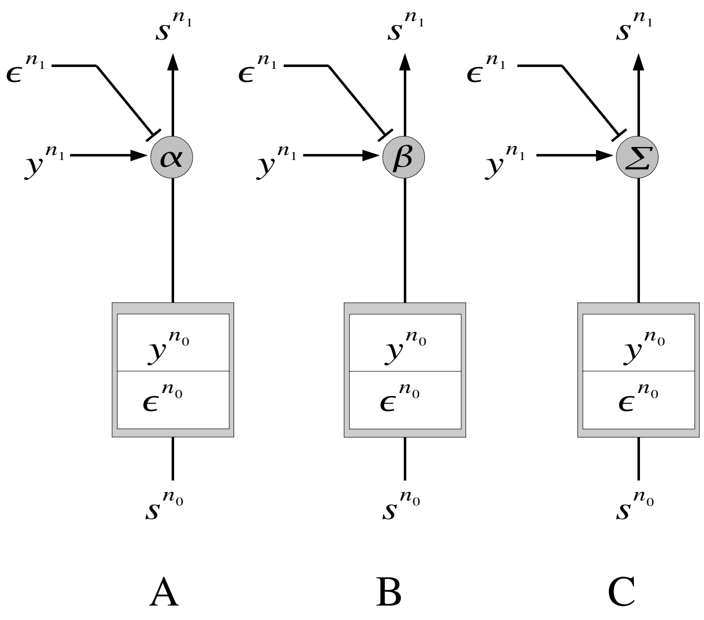


Figure 7: *A depiction of* pARC *signals, wires, and gates. A signal, s^{n_i}, is a packet of sensorimotor data, y^{n_i}, and saliency data, ϵ^{n_i} predicted n timesteps forward. A gate is a junction between a primary wire (vertical) and a secondary wire (horizontal). It allows new data of the same type to be merged with an existing signal and the number of prediction timesteps to be expanded from s^{n_i} to $s^{n_{i+1}}$. Three gates are defined, **(A)** α, **(B)** β, and **(C)** Σ, which modulate their merging behavior based on the saliency measure ϵ_n. Gate behavior is further described in the text.*

A saliency stream is a measure of the *value* of the associated sensorimotor datum. The notion of value is context dependent. If y^n is a predictive stream, then ϵ^n can be a measure of how well the prediction has recently measured against the actual sensorimotor values.

If y^n is a commanded value, then ϵ^n can be a measure of the command's priority. It is used for arbitration between competing commands. This value may be hard coded or dynamically computed based on the robot's attention and motivation system.

A gate merges sensorimotor and saliency data into an existing signal stream and outputs the result. Three types of gates are defined: α, β, and Σ. Each gate takes input from a primary and secondary wire. It can overwrite some or all of the signal on the primary wire with data on the secondary wire.

The Σ gate outputs the sum of the sensorimotor streams, y^n, on the primary and secondary wires. It also outputs the maximum ϵ^n for each time-step. This gate can be used to sum motor torque commands, for example, when simulating groups of virtual springs.

The α gate overwrites all of the data in s_n if the average ϵ^n on the secondary wire is larger than the average ϵ^n on the primary wire for all n time steps. This allows the secondary wire to subsume control of the primary wire and masquerade as a signal source from a lower layer, providing arbitration between competing behaviors.

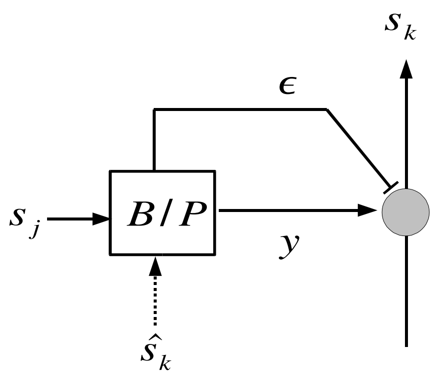


Figure 8: ***(Left)*** *A* pARC *kernel is a computational thread which inputs signals s_j and computes the output signal s_k. A predictive kernel, B, computes the continuation of s_j as s_k for n timesteps forward. The signals s_j and s_k may be in different sensorimotor spaces, in which case the kernel is a map from one space to the other. A behavior kernel, B, takes input signal s_j and outputs a behavioral command as signal s_k. If the kernel is predictive, the optional input signal $\widehat{s_k}$ is the actual value of the signal (versus predicted). It can be used in computing the prediction error and the stream saliency ϵ. For a behavior kernel, the ϵ value can be hardwired or computed dynamically based on attention and goal-driven signals.*

The β gate overwrites only those time-steps in s^n where the ϵ^n on the secondary wire is greater than the primary wire's ϵ^n. This allows the secondary wire to refine predictive data from lower layers.

4.3.4 Predictive and Behavior Kernels

A *pARC* kernel, depicted in Figure 8, is a computational thread which generally inputs one or more sensorimotor streams and outputs a new sensorimotor stream as well as a measure of the saliency of that output stream. Specific examples of kernels are provided in Section 5. For now, they are treated as generic input-output computations.

Behavior based systems are typically composed of many interacting behavioral modules. For example, Mataric's Toto robot used the modules STROLL, AVOID, ALIGN, and CORRECT. A behavior kernel in *pARC*, defined as B, is similar to these types of modules. As depicted in Figure 8, a generic behavior module takes as input signal s_j and outputs signal s_k in some other sensorimotor space. However, B also computes the saliency stream ϵ, which can be used for behavior arbitration. The ϵ value can be hardwired or computed dynamically based on attention and motivational signals. We refer the reader to [1] for examples of types of behavior modules.

A predictive kernel, P, inputs computes the predicted continuation of s_j as s_k for n timesteps forward. The kernel also computes the saliency of its prediction in the saliency stream ϵ based on the signal $\widehat{s_j}$. This an optional input signal of the actual value of the signal (versus predicted). P can have different forms depending on its use. If $n = 0$, then P is no longer predictive but can define a transform from one sensorimotor space to another. The exact form of P is an area of further research. We do stipulate that the predictive kernel should have the following characteristics:

1. The signals s_j, and s_k should be generated from the same underlying physical process. That is, they are not independent signals and P should be a realizable.
2. The signals s_k and s^k should be of relatively low dimension, making the learning of P tractable.

We should like a homogeneous form for P such that the same predictive kernel can be used independent of the sensorimotor modality. Unfortunately this isn't practical in non-trivial cases. Marjanovic [34] presents a homogeneous system which attempts to autonomously build sensorimotor transform functions based on correlations in the data stream. While this approach is compelling, it is not demonstrably practical for complex robot controllers.

Ideally, P is constructed online from sensorimotor data streams. It is tempting to view learning P as a matter of function approximation or a suitable candidate for HMM and POMDP machine learning algorithms. In some situations, this may be appropriate. In some cases, say in encoding the forward kinematics of the arm, a traditional model based approach may be the most direct route to success. Alternatively, we can view the construction of P as something akin to Mataric's building of landmark graphs based on sonar, compass, and wheel velocity sensors. Or it could be something as simple as a table lookup.

4.3.5 *pARC* Example: Reaching to a Target

Figure 9 provides an example of a robot controller decomposition using the *pARC* framework. The controller guides the robot arm towards visual targets. This ability is built incrementally. It constructs predictive models of gravity loading on the arm, target occurrence in the visual image, and hand occurrence in the visual field. It also implements behaviors which control the head pose, control the forces on the arm, and guide the end-effector to a target.

The controller in Figure 9 is best described at a macro level in terms of the roles and interactions of the assorted kernels. We break the description down as follows:

- **Hand Prediction**: This model predicts the occurrence of the hand in the visual field. An initial estimate is made with a forward kinematic model $FwdLoc$. The behavior kernel $LookAtHand$ uses this estimate to guide the head to foveate the hand location. By keeping the hand in the field-of-view, $LookAtHand$ allows predictive kernel $VisualLoc$ to improve the estimate through a feature based model of the hand. Finally, kernel $FlowLoc$ estimates the optic flow caused by hand movement and predicts the visual trajectory forward in time using, perhaps, a Kalman filter.
- **Gravity Prediction**: The model produces an estimate of gravity loading on the arm based on the joint angle stream. The initial kernel, $KineGrav$ uses a rough kinematic model and mass distribution to build the original estimate. Kernel $NullCompGrav$ is built based on arm motion generated from behavior $LookAtHand$. It builds a model which nulls the error between the prediction and the sensed forces as the arm is moved, essentially accounting for arm dynamics. The kernel $ContinueGrav$ estimates the gravitational loading forward in time by extrapolation of the loading trajectory.

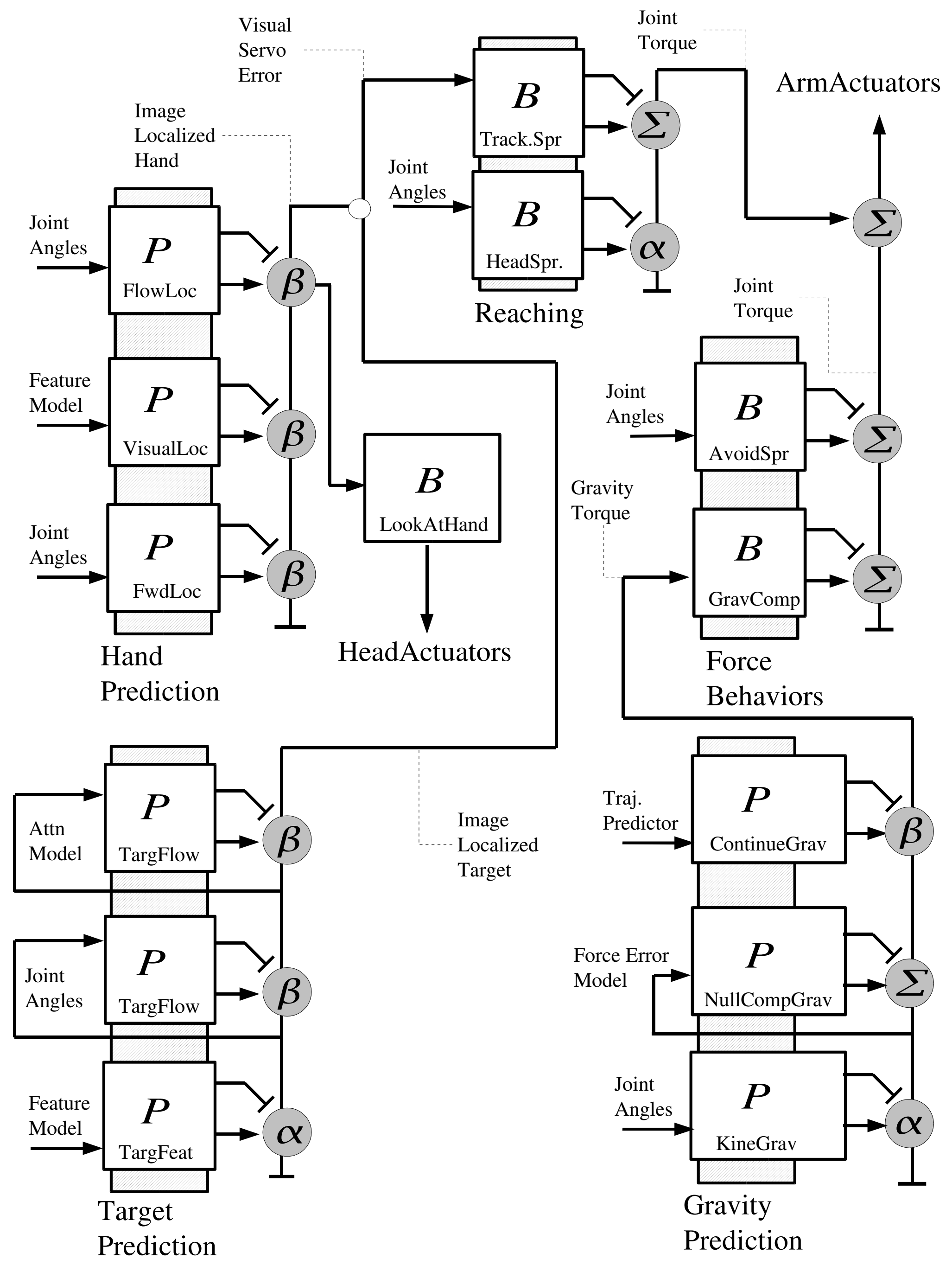


Figure 9: An example of a robot controller decomposition using the *pARC* framework. The controller guides the robot arm towards visual targets. This ability is built incrementally. It constructs predictive models of gravity loading on the arm, target occurrence in the visual image, and hand occurrence in the visual field. It also implements behaviors which control the head pose, control the forces on the arm, and guide the end-effector to a target. See the text for a more detailed description.

- **Force Behaviors**: These behaviors produce a joint torque stream to the arm. Behavior kernel *GravComp* simply counterbalances the arm against the gravity loading signal. This helps minimize the position errors accumulated in the arm due to the arm dynamics and its high compliance. The kernel *AvoidSpr* simulates a set of virtual springs between the robot torso and a point on the forearm. These springs are summed on top of the gravity signal and cause the arm to avoid collisions with the body.

- **Target Prediction**: This model detects reaching targets in the visual field and estimates the continuation of their trajectory. Kernel $TargFeat$ estimates the location of a salient object based on a visual feature model. The estimate is refined by applying an optic flow filter $TargFlow$, and an attention system saliency filter $TargAttn$. Extrapolation of the target trajectory in $TargAttn$ predicts the continuation of the target in the visual field.

- **Reaching**: A reaching to target behavior is achieved through two layers of kernels. The kernel $HeadSpr$ simulates a virtual spring attached from the end-effector to a point lying on a ray perpendicular to the image plane. While the head is tracking a target, this keeps the hand near the target and in the visual field. The kernel $TrackSpring$ visually servos the hand to the target in the image plane using a second virtual spring and the hand localization information.

4.3.6 *pARC* Detailed Example: Hand Localization

To further illustrate the *pARC* framework, we provide a detailed description of the hand localization component from the previous example. This subsystem is depicted in Figure 10.

The controller is bootstrapped with kernel *FwdLoc* which uses a purely kinematic model and the current robot joint angles, s_j, to compute $s_k^{0_0}$. This is the instantaneous estimate of the location of the hand in the visual image. *FwdLoc* computes the target saliency as ϵ^{0_0} which is set to a constant value as this is the first kernel in the model. If the hand falls outside of the image, then $\epsilon^{0_0} = 0$. The kernel's β gate will pass all signals up the afferent pathway.

The behavior $LookAtHand$ is then implemented. It uses the signal s_k to keep the hand foveated in the center of the camera image. $LookAtHand$ sits on the efferent pathway. Consequently, as additional kernels are computed, $LookAtHand$ will automatically more accurately track the camera to the future location of the hand.

The kernel $VisualLoc$ visually identifies the hand in the image. The identification is simplified by utilizing the current hand estimate, s_k to limit the visual search to probable locations. The visual identification is accomplished by a feature based model of the hand. This model is built from training data gathered using the *FwdLoc* and $LookAtHand$ behaviors. As the arm moves about its workspace, due to external behaviors or by manual guidance, the camera roughly tracks its location. Overtime, a feature based model of the hand can be built as the workspace background can be subtracted out and on average the hand appears at the estimated location. The feature model also provides $VisualLoc$ a quantitative means to calculate ϵ^{0_1} based on a statistical match to the model. $VisualLocs$'s β gate can then overwrite the kinematic estimate when the model estimate is better.

Finally, the kernel *FlowLoc* is implemented. This kernel inputs both the joint angles s_j and the current hand prediction s_k. The kernel predicts the continuation of the trajectory of s_k forward in time by 10 time steps. The prediction is computed using the optic flow at the hand image location and the angular velocity of the camera. The saliency of the prediction, ϵ^{10_2}, is a measure of how

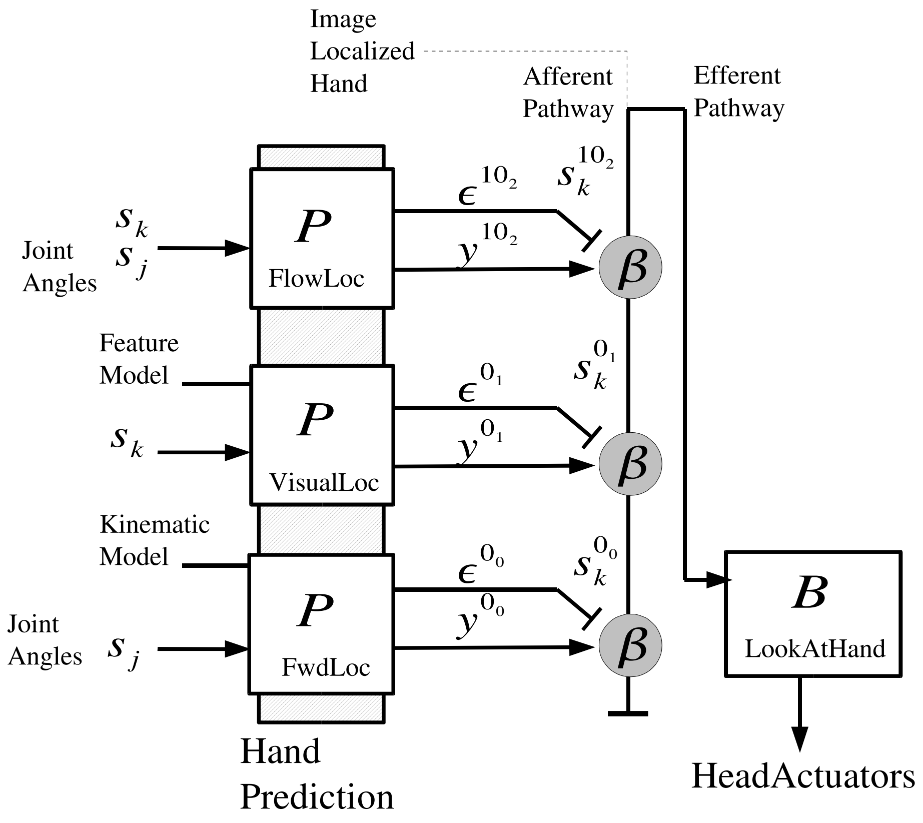


Figure 10: A detailed view of the hand localization and prediction kernels. See the text for a description.

well the kernel predicts forward in time. This value can be computed using the average prediction error looking backwards in time. Because there are no previous predictive samples in s_k, *FlowLoc*'s β gate will add all of the prediction samples to the signal. The first sample of y^{10_2}, corresponding to the current hand location, will be exactly the estimate from the previous layer, and the gate will have no effect on its value.

5 Manipulation Scenarios

The primary research directions for *Domo* will be incorporated into a set of manipulation scenarios. The scenarios provide a path for the robot to incrementally acquire a richer set of behaviors, both reactive and anticipatory. The scenarios are centered around play type activities for the robot which follow a developmental schema similar to a young child. The activities involve explorations of and interactions with children's toys and will be developed in specified stages. Following the artificial creature approach, they are integrated into a single cognitive system. The robot will be capable of discriminating between scenarios and selecting the appropriate play activity given the current environmental context. The robot's scenarios are: development of a body schema, playing tether ball with itself, playing karate sticks with a person, and putting away its toys. These are

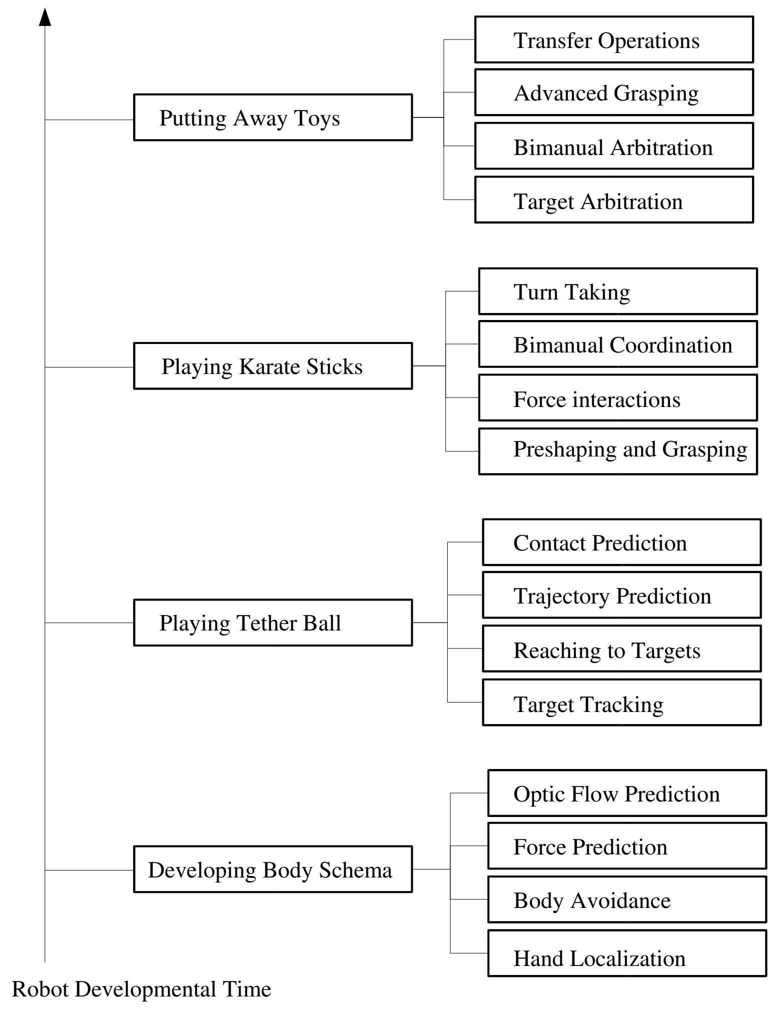


Figure 11: *The four manipulation scenarios and their principal components.*

described in further detail below. Figure 11 depicts the scenarios and their principal components. For each scenario we provide an overall description including which of the nine elements of our behavior based decomposition it addresses. We then describe each of the basic competencies to be developed, followed by a short discussion of the interesting features of the scenario.

5.1 Developing a Body Schema

Description: The initial stage of a developmental course for a humanoid robot involves active exploration of the sensorimotor space. The explorations provide sensorimotor data from which to build into the robot a sense of what its body 'feels like'. This is the notion of a body schema, a broadly used term to denote an egocentric view of the proprioceptive relationships between a body and its world [4]. The initial scenario for *Domo* involves developing a set of body schema modules. The modules provide anticipatory information about the relationship between the robot's actions and their sensorimotor consequences. This information provides a basis to build subsequent scenarios from.

Manipulation areas: Positioning sensors, Perception, Force operations

Hand localization: Hand localization involves estimating the occurrence of the robot hand in the visual field. It is an essential component of visually guided manipulation and a number of approaches exist. A typically engineered approach models the kinematic relationship between the arm pose and the camera pose. This requires accurate knowledge of the robot kinematics and the camera optics. Neither of these are available to *Domo*. The more biologically plausible approach done with Babybot [43, 39] involves learning the map between arm pose and visual coordinates by training a neural network over the transform. The robot's hand is first oscillated with a fixed frequency in the visual field. By filtering the image at that frequency, an visual segmentation is obtained. The segmentation is repeated throughout the robot's workspace to obtain the data set needed to train the neural network. The approach is biologically plausible as very young infants utilize a hand waving during early sensorimotor development. *Domo*'s hand localization will be achieved by similar means.

Body avoidance: Body avoidance protects the manipulator from collisions with the robot torso during reaching and when an interacting human guides the arm too close to the body. It is a behavior of practical importance but also contributes to the cognitive notion of the robot's body schema. It grounds the representation of the robot's trunk in terms of the manipulator. The approach taken with *Domo* is to apply a set of virtual springs between the manipulator and the body. This has already been implement using a set of nonlinear springs and dampers. The force function for the springs is:

$$F = \frac{k}{2}(1 + \cos(\frac{x\pi}{c})) - d\dot{x}$$

where k is the spring stiffness, x is the spring length, d is the damping gain, and c is the range of influence of the spring. Outside of the range, c, the force is defined as $F = 0$. The spring saturates at force $F = k$. The nonlinear aspect allows the springs' force to rapidly dissipate as the manipulator moves beyond a critical collision area. There are three spring behavior types: a spring between two points, a spring between a line and a point, and a spring between a plane and a point. The spring length is computed as the minimal distance between the two features. At each timestep, the virtual spring force acting on the arm is computed and translated into joint torques. Each spring is implemented as an independent behavior. *Domo*'s body avoidance is achieved by a virtual spring between the hand and the front body plane, a spring between the forearm and the vertical corner of the torso, and a spring between the elbow and a common collision point on the side of the torso.

Force prediction: Force prediction is an efference copy mechanism, widely studied in humans [51], where an expectation of the forces encountered during ballistic movements is compared with the real sensory experience. The expectation provides the robot with a sense of what its arms 'feels like'. Violations of this expectation can be used to amplify the saliency of a stimulus and draw the robot's attention. During interactions with objects and contact surfaces, force predictions can be used to modulate the stiffness of the arm. Predictive force models can be built during an exploration phase. The robot explores its workspace with a set of hardwired arm behaviors that allow it to experience a range of postures and ballistic trajectories. The sensorimotor data from the exploration is applied to learn a map from manipulator pose and velocity to manipulator torques. The actual torques sensed from the SEA are used as training data. The map need only be roughly correct as the prediction is not used directly in a tightly coupled control loop. Even knowing the correct sign of the expected manipulator torques can provide useful information to behaviors responding to disturbances in the environment. The learning algorithm used in developing the map is an open research question. An additional modality to be developed in force prediction is a model of the gravitational loading on the arm. This can be achieved with standard robotics techniques [10].

Optic flow prediction: The optic flow induced on a robot's camera during head and eye movements is a measure of the proprioceptive state of the robot and the objects found in the world. The ability to attend to salient stimuli depends on, in part, the ability to distinguish between self-induced optic flow and optic flow due to externally moving objects. This is a predictive problem with biological foundations. The Vestibulo Ocular Reflex (VOR) in primates generates compensatory eye movements to head motion. The reflex operates in an open loop fashion, generating anticipatory eye movements to stabilize the image. Even with VOR, the robot will generate self-induced optic flow, and building a prediction of this flow helps to distinguish between itself and the world. Optic flow prediction is one of the most studied areas in robotic anticipatory systems, including robotic work with stationary textured environments [52] and humanoid attentional systems [40]. A predictive model of egomotion flow can be learned based on the flow in static environments combined with the head's kinematic trajectory. The model can be built from purely kinematic information or learned using standard machine learning techniques.

Discussion: Anticipatory mechanisms at the body schema level serve two primary purposes: amplification of salient sensory stimuli and feedforward compensation in motor control. Saliency amplification emphasizes the relevance of sensory signals which lie outside the schema predictions. Externally generated forces on the manipulator, by contact or human intervention, will conflict with the force prediction module. This conflict serves as a cue to higher level behaviors. The same holds true for optic flow prediction. With hand localization, the predictive estimate of the hand in the image can serve to draw attention to occlusions by objects. The gravitational model of forces acting on the arms, and the body avoidance schema, provide feedforward signals to the robot's motor control. These bias the control of the manipulator to normalize against gravity and to avoid harmful collisions. The body schema modules chosen are tractable with respect to available analytical and machine learning techniques. In many cases, a lookup table may suffice. The success of each can be quantified by behavioral observation and by measurement of the predictive error when appropriate. Optic flow and force prediction can be measure by the ability to amplify external stimuli from egomotion generated stimuli. Body avoidance can be analyzed in terms of the manipulator workspace covered during random movements. The hand localization module can be studied in terms of its robustness to occlusions and its ability to match its prediction to a known hand location gained from oscillatory motion.

5.2 Playing Tether Ball

Description: Tether ball is playground game where a child bats at a moving ball with her hands. The ball is suspended from a pole by a rope. It is a game of basic visual and motor coordination that can be used as a constrained task for *Domo* to interact with the world in a structured manner. The ball is on a rope and will always, eventually, return to a set postion. This enables the robot to play the game without human intervention over long periods of time. It allows for extended interactions to generate sensorimotor data which, in turn, can be used to refine the body schema models from the previous scenario. It may also be played with one or two hands and provides opportunity to investigate simple bimanual action selection.

Manipulation areas: Positioning sensors, Perception, Force operations, Detecting failures

Target tracking: The first necessary component is a robust means of tracking a target, which in this case is a ball. The tracking involves performing an initial segmentation of the target from the environment, and also tracking its continuation between visual frames. The tracking can be accomplished with well established techniques such as the Lucas-Kanade Tracker [33]. The tracking involves an active coupling between the head pose and the object as well. In the specific case of a ball, simple template and color histogram models suffice to reliably segment the object. Additional depth information can be derived using the pixel based size of the ball. In this manner, *Domo*'s visual system can be bootstrapped to investigate the principal problem of visual-motor coordination. A more general approach to object segmentation and tracking will be needed for later scenarios.

Reaching to targets: Robot reaching to visual targets is a well studied area, both from an engineering perspective [10] and a biological perspective [43]. In the case of *Domo*, this involves a mapping a trajectory from the image localized target to the 4 DOF in the shoulder and elbow. The wrist is treated independently from the upper arm. Reaching which incorporates approach planning is deferred until a later scenario. Clearly, a traditional approach utilizing inverse kinematics could work. However, the compliance in *Domo*'s arms would introduce large offset errors. Additionally, this would require an precise model of the arm kinematics and of the target location. We propose investigating a hybrid approach utilizing feedforward joint postures and a Virtual Model controller. The sensorimotor exploration of the previous scenario can be used to build a simple map from visual coordinates to joint angles. A trajectory generation mechanism can then be used in a behavior which keeps the hand in the visual image using this feedforward map. If the robot is fixating the target and the hand is in the visual image, then VMC springs can be applied to visually servo the hand to the target (or its expected location). A related hybrid approach was used with the humanoid DB [19].

Trajectory prediction: The trajectory of the ball on the rope should follow a smooth path between external contacts. Predicting the continuation of this trajectory is critical to the robots ability to swing at the moving ball. This is a well constrained and structured problem. It involves predicting the time series continuation of a three dimensional signal. A general approach to this problem can be applied to other predictive domains for *Domo* as well. Two proposed areas of investigation are Kalman filters and temporal difference learning [53].

Contact prediction: Playing tether ball requires a turn-taking interaction between the robot and the ball. Contact is made and then the ball is allowed decelerate before contact is made again. Constructing transition points in the turn-taking requires prediction of contact. Contact prediction involves both detection of manipulator contact with the ball and anticipation of the event. Contact detection can be determined from models of the velocity and acceleration of the target. Conflicts with the underlying manipulator force schema can also be used to detect collisions. Anticipation of contact can achieved with similar methods to those used for trajectory prediction.

Discussion: This scenario builds off of many of the competencies gained in the body schema scenario. It is particularly interesting in that it affords a long duration sensorimotor experience to the robot which involves anticipatory visual motor coordination. This opens up an opportunity for on-line learning of how to best modulate the reaching strategy for the manipulator. One approach would be to use reinforcement learning to best modulate a set of reaching behaviors. The success of the control strategy is readily quantified by the error in hand to target position. The scenario involves challenging technicalcal subproblems but these can likely be addressed independently with different techniques. The realtime visual localization of the target in space is greatly simplified by the target type. This allows the scenario to focus on the development of the predictive components and incorporating the predictions into the manipulator control strategy.

5.3 Playing Karate Sticks

Description: Karate sticks is a children's game derived from Filipino stick fighting. Each child holds a broomstick type rod between two hands at about a shoulder's width apart. They play by thrusting their stick at the other in order to make contact between the two sticks. The goal is to read the orientation of the other's stick and match it such that the two sticks make contact at orthogonal orientations at approximately the center of the stick. In this scenario with *Domo*, a human would present the stick to the robot. The robot would reach and make a bimanual grasp of the stick at the appropriate locations. The human robot would then play the game with the robot predicting the pose of the human's stick at the time of contact. This leverages on the competencies of the tether ball scenario.

Manipulation areas: Positioning sensors, Perception, Grasping, Force operations, Detecting failures

Preshaping and grasping: The scenario requires development of simple predictive grasp preshaping based on the stick orientation. When the human presents the stick for play, the correct grasp shape and manipulator approach needs to be formed. This is a constrained version of the more general, and important, issue of grasp preshaping in that only a homogeneous power grasp is needed to grasp a cylinder along its axis. The hands on *Domo* are limited in the dexterity of grasps available and this task represents a good match between the capabilities of the hands and the exploration of preshaping abilities. However, this preshaping task requires robust identification of the object pose as well selection among differing approaches and grasp shapes depending on the object orientation. One method previously demonstrated on *Domo* is to build a look-up table map from image pose to hand and wrist pose.

Force interactions: Force interactions occur throughout the scenario, between the two arms and the stick, and between the two sticks. One mode of play involves maintaining stick contact while taking turns pushing on the opponents stick. This can be achieved by a VMC behavior which modulates manipulator endpoint forces at a given stick orientation. A second feature of the the force interactions is demonstrated by the relatively high contact forces which occur between the two sticks during play. The compliance in the manipulator accommodates this naturally and protects the robot geartrain. The force sensing provides a means of detection of contact, and consequently, anticipation of contact. In turn, the stiffness of the manipulator arms can be modulated by this anticipation.

Bimanual coordination: The scenario requires coordination between the two arms while adjusting the orientation of the robot's stick. When the desired orientation has been achieved, a fixed action pattern thrusting of the stick can occur, contingent on the turn taking nature of the game. The robot interweaves a stick orientation behavior and a thrusting behavior. The orientation behavior can take advantage of the force controlled nature of the arms. A mechanical force coupling occurs between the arms through the stick. As one arm leads a motion, the other can naturally adapt through the interaction forces with the stick. Because the joint angles are only controlled through the force controllers, and because the arms are compliant, precise coordination of the arms should not be necessary.

Turn taking: Karate sticks requires turn taking between the human and robot. Typically one player can lead the play by deciding when to thrust the stick and the other follows. One player also entrains the orientation of their stick to the other. In this manner, a human playing with the robot can lead the play interaction fruitfully. If the robot chooses the incorrect stick pose or an inopportune moment to thrust its stick, the human will naturally adapt its turntaking to follow. In this manner, the human-robot interaction scaffolds the play activity into one that is richer than the robot's abilities allows. The robot needs to read the orientation of the human's stick, a constrained visual problem given the simple geometry of the object. For richer turn taking to occur, the robot needs to predict the continuation of the stick trajectory and anticipate the thrusting of the stick.

Discussion: This scenario demonstrates a tight coupling between the robot and the world through interaction forces. It capitalizes on many of the features unique to the manipulators developed for the robot. Visual perception during the task is constrained to recognizing the stick pose and also possibly detection and tracking of the human's face, both of which are tractable problems. It allows an exploration of bimanual, force-based manipulation. It also provides an opportunity to explore human-robot turn taking during manipulation, where the turn-taking is driven by the manipulation interaction forces. One risk is that the robot's hands are not strong enough to maintain a stable grasp on the stick. In the case that it is dropped, the play can continue as the robot should be able to reach and regrasp the stick when it is offerred. The success of the scenario can be measured in the ability to match the human's stick orientation, the speed at which *Domo* can play the game, and the robustness of its preshaping and grasping behaviors.

5.4 Putting Away Toys

Description: This scenario of putting away toys is designed to incorporate elements of the three preceding scenarios and to generalize some of the perceptual and motor competencies into behaviors which operate in less structured environments. It involves the robot picking up objects from a table and placing them in a box. The objects are heterogeneous in size, shape, color, and texture. They are positioned on the table in an arbitrary fashion and may overlap and occlude one another. The box can be placed anywhere in the robot's reachable workspace.

Manipulation areas: Deciding on actions, Positioning sensors, Perception, Grasping, Force operations, Transfer, Disengaging, Detecting failures

Target arbitration: Target arbitration requires selecting an object, among many, on the table to grasp. This is a perceptually difficult problem, and a combination of techniques can be employed without requiring a full visual reconstruction of the object. Fitzaptrick et al. [17] approached the perceptual problem by having the robot poke at objects in the environment. The optic flow generated from the poking allowed a clean segmentation of the object from the background. A second approach to visual segmentation based on texture cues [29], developed in our lab, will be employed to segment homogeneous regions of texture from the image. A combination of the two approaches should allow *Domo* to determine a number of regions of interest for grasping. Arbitration among objects requires determining which object to reach for based on a saliency metric. The metric is computed from a set of hardwired 'graspability' cues such as size and, depth, and segmentation confidence. The arbitrator need also detect failures during grasping based on tactile and visual cues. Detection of a failure can allow a second attempt at grasping or allow a different object to be attempted.

Bimanual arbitration: *Domo* can use one or two hands to grasp an object on the table. Arbitration between single and two handed manipulation can be accomplished based on a predictive module which embeds the relationship between a set of visual features (size, roundness, etc) and the expected success using one or two hands. The module can be built through a set of prior manipulation engagements with objects where both types of manipulation are attempted. The bimanual coordination developed in the previous scenario can then be generalized to a richer set of objects, where the interaction forces between the two manipulators can be used to form a full armed pincer type grasp. Detection of single handed grasping failure can be used to signal the arbitrator to reattempt using a bimanual approach.

Advanced grasping: Grasp preshaping in the previous scenario was limited to orienting the wrist correctly in order to form a cylinder grasp on the stick. Preshaping is a fundamental anticipatory behavior used during manipulation. *Domo* has 1 DOF in the hand 2 DOF in the wrist which can be used to precondition the hand's engagement with an object. The limited dexterity of the hand still affords a rich set of preshapes. A simple approach to preshaping, already implemented on the robot, maps the object's primary orientation axis to a fixed set of hand postures. The mapping is hand coded and brittle. The approach can be expanded to use a prior set of manipulation engagements to build the map from. A coarse estimate of the grasp's correctness can be calculated from the total tactile sensation. This can be used to bias the map from visual features to hand postures in the appropriate direction. A primary grasping ability to be developed is distinguishing between grasp-from-above and grasp-from-the-side approaches to the object.

Transfer operations: The final component of the scenario is transferring a grasped object to the open box. The box can be visually localized base on its geometry, and the transfer can be completed by the previous reach-to-target behavior. An interesting extension is to incorporate locally guided path planning during the transfer. With basic depth cues from the objects on the table, a set of avoidance springs, as used in the body avoidance module, can be added to move the manipulator around protruding objects.

Discussion: This scenario is the least constrained and therefore the most challenging of the four. It allows exploration of the coupling between visual features, prior manipulation experience, and action planning. The set of objects available to the robot cannot be completely general given the limitations of the robot's hands. However the variety of objects used can illustrate anticipatory aspects of the preshaping and bimanual behaviors. The visual processes can be developed largely with existing software, but the success of the scenario depends largely on the ability to segment out the countours of individual objects. This success can be measured in the adaptability to different types of objects, the speed of execution, and the ability to form appropriate grasp preshapes and bimanual arbitrations.

6 Milestones and Timeline

The following milestones will be accomplished:

Date		**Milestone**
September	2004	Design, fabrication, and assembly of *Domo*.
December	2004	Testing of low-level controllers.
January	2005	Development multi-processor real-time control architecture and necessary software infrastructure.
February	2005	Integration of existing vision code into software infrastructure.
March	2005	Development of initial *pARC* software framework.
June	2005	Development of the Body Schema scenario
September	2005	Development of the Playing Tether Ball scenario
December	2006	Development of the Playing Karate Sticks scenario
March	2006	Development of the Putting Away Toys scenario
June	2006	Collection of experimental data. Completion of writing of dissertation.

References

[1] Ronald Arkin. *Behavior Based Robotics.* MIT Press, 1998.

[2] A. Arsenio and P. Fitzpatrick. Exploiting cross-modal rhythm for robot perception of objects. In *Proceedings of the Second International Conference on Computational Intelligence, Robotics, and Autonomous Systems*, December 2003.

[3] Lijin Aryananda and Jeff Weber. MERTZ: A Quest for a Robust and Scalable Active Vision Humanoid Head Robot. In *Proceedings, IEEE-RAS International Conference on Humanoid Robotics*, Santa Monica, Los Angeles, CA, USA., 2004. IEEE Press.

[4] Jose Bermudez, Anthony Marcel, and Naomi Eilan, editors. *The Body and the Self.* The MIT Press, Cambridge, Mass., 1995.

[5] Cynthia Breazeal. *Sociable Machines: Expressive Social Exchange Between Humans and Robots.* PhD thesis, MIT, Cambridge, Ma, June 2000.

[6] R. Brooks. Challenges for complete creature architectures. In *Proceedings of Simulation of Adaptive Behavior (SAB90)*, 1990.

[7] Rodney Brooks, Rodric Grupen, and Robert Ambrose. Autonomous Manipulation Capabilities for Space and Surface Operations. Proposal to NASA in response to BAA-04-02, October 2004.

[8] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2:14–23, April 1986.

[9] Rodney A. Brooks, Cynthia Breazeal, Matthew Marjanovic, Brian Scassellati, and Matthew M. Williamson. The Cog project: Building a humanoid robot. In C. L. Nehaniv, editor, *Computation for Metaphors, Analogy and Agents*, volume 1562 of *Springer Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1999.

[10] J. Craig. *Introduction to Robotics.* Addison Wesley, 2 edition, 1989.

[11] E. Datteri, G. Teti, C. Laschi, G. Tamburrini, P. Dario, and E. Guglielmelli. Expected perception: an anticipation-based perception-action scheme in robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-03)*, volume 1, pages 934–939, Las Vegas, NV, Oct 2003.

[12] P. Davidson and D.M. Wolpert. Motor learning and prediction in a varialbe environment. *Current Opinion in Neurobiology*, 13:1–6, 2003.

[13] Aaron Edsinger-Gonzales. Design of a Compliant and Force Sensing Hand for a Humanoid Robot. In *Proceedings of the International Conference on Intelligent Manipulation and Grasping*, July 2004.

[14] P. Fitzpatrick, G. Metta, L. Natale, S. Rao, and G. Sandini. Learning About Objects Through Action: Initial Steps Towards Artificial Cognition. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, May 2003.

[15] Paul Fitzpatrick. *From First Contact to Close Encounters: A developmentally deep perceptual system for a humanoid robot.* PhD thesis, Massachusetts Institute of Technology, 2003.

[16] Paul Fitzpatrick and Giorgio Metta. *YARP: Yet Another Robot Platform.* MIT Computer Science Artificial Intelligence Laboratory, http://sourceforge.net/projects/yarp0, 2004.

[17] Paul Fitzpatrick, Giorgio Metta, Lorenzo Natale, Sajit Rao, and Giulio Sandini. Learning about objects through action - initial steps towards artificial cognition. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA-03)*, volume 3. IEEE Press, 2003.

[18] J.R. Flanagan, S. King, D.M. Wolpert, and R.S. Johansson. Sensorimotor prediction and memory in object manipulation. *Canadian Journal of Experimental Psychology*, 55:89–97, 2001.

[19] Chris Gaskett and Gordon Cheng. Online Learning of a Motor Map for Humanoid Robot Reaching. In *Proceedings of the 2nd International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2003)*, Singapore, December 2003.

[20] H. Gomi and M. Kawato. Human arm stiffness and equilibrium-point trajectory during multi-joint muscle movement. *Biological Cybernetics*, 76:163–171, 1997.

[21] R. Grupen and C Coelho. Structure and Growth: A Model of Development for Grasping with Robot Hands. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)s*, Takamatsu, Japan, November 2000.

[22] Kuzanuao Uchiyama Haruhisa Kawasaki, Tsuneo Komatsu. Dexterous anthropomorphic robot hand with distributed tactile sensor: Gifu hand ii. *IEEE Transactions on Mechatronics*, 7(3):296–303, September 2002.

[23] Horst Hendriks-Jansen. *Catching Ourselves in the Act.* MIT Press, Cambridge, Mass., 1996.

[24] T. L. Huntsberger, H. Aghazarian, E. Baumgartner, and P. S. Schenke. Behavior-based control systems for planetary autonomous robot outposts. In *Proceedings of Aerospace 2000*, Albuquerque, NM, 2000.

[25] Point Grey Research Inc. *FireFly2 IEEE-1394 CCD Camera Manual.* "http://www.ptgrey.com/products/firefly2/firefly2.pdf", 2004.

[26] P. Dilworth J. Pratt, M. Chew and G. Pratt. Virtual Model Control: An Intuitive Approach for Bipedal Locomotion. *International Journal of Robotics Research*, 20(2):129–143, 2001.

[27] Roderic A. Grupen Jefferson A. Coelho Jr., Justus H. Piater. Developing Haptic and Visual Perceptual Categories for Reaching and Grasping with a Humanoid Robot. In *Proceedings of the First IEEE-RAS International Conference on Humanoid Robots*, Cambridge, MA, USA, September 2000.

[28] E. R. Kandel, J. H. Schwartz, and T. Jessell. *Principles of Neural Science.* McGraw-Hill and Appleton and Lange, New York, NY, 4th edition, January 2000.

[29] Charlie Kemp. Duo: A Human/Wearable Hybrid for Learning About Common Manipulable Objects. In *Proceedings, IEEE-RAS International Conference on Humanoid Robotics.* IEEE Press, 2003.

[30] Oussama Khatib. A unified approach to motion and force control of robot manipulators: The operational space formulation. *International Journal of Robotics and Automation*, 3(1):45–53, 1987.

[31] K. Kording and D.M. Wolpert. Bayesian integration in sensorimotor learning. *Nature*, 427:244–247, 2004.

[32] C. Lovchik and M. Diftler. The robonaut hand: A dexterous robot hand for space. In *Proceedings of the IEEE International Conference on Automation and Robotics*, volume 2, pages 907–912, Detroit, Michigan, May 1999.

[33] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[34] Matthew J. Marjanovic. *Teaching an Old Robot New Tricks: Learning Novel Tasks via Interaction with People and Things*. PhD thesis, MIT, Cambridge, Ma, 2003.

[35] Matthew Mason. *Mechanics of Robotic Manipulation*. MIT Press, August 2001. Intelligent Robotics and Autonomous Agents Series, ISBN 0-262-13396-2.

[36] Maja J Mataric. Integration of representation into goal-driven behavior-based robots. *EEE Transactions on Robotics and Automation*, 8(3):304–312, June 1992.

[37] Maja J Mataric and Rodney A. Brooks. *Cambrian Intelligence*, chapter Learning a Distributed Map Representation Based on Navigation Behaviors, pages 37–58. MIT Press, 1999.

[38] J. McIntyre and E. Bizzi. Servo hypotheses for the biological control of movement. *Journal of Motor Behavior*, 25:193–203, 1993.

[39] Giorgio Metta. *Babybot: a study into sensorimotor development*. PhD thesis, LIRA-Lab, DIST, University of Genoa, 2000.

[40] Giorgio Metta. An attentional system for a humanoid robot exploiting space variant vision. In *IEEE-RAS International Conference on Humanoid Robots*, 2001.

[41] R. Moller. Perception through anticipation - an approach to behavior-based perception. In *Proceedings of New Trends in Cognitive Science*, pages 184–190, Vienna, Austria, 1997.

[42] Hans Moravec. The Stanford Cart and the CMU Rover. In I. J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 407–41. Springer-Verlag, 1990.

[43] Lorenzo Natale. *Linking Action to Perception in a Humanoid Robot: A Developmental Approach to Grasping*. PhD thesis, LIRA-Lab, DIST, University of Genoa, 2004.

[44] Robert Platt, Oliver Brock, Andrew H. Fagg, Deepak R. Karuppiah, Michael T. Rosenstein, Jefferson A. Coelho Jr., Manfred Huber, Justus H. Piater, David Wheeler, and Roderic A. Grupen. A Framework For Humanoid Control and Intelligence. In *Proceedings of the 2003 IEEE International Conference on Humanoid Robots*, 2003.

[45] G. Pratt and M. Williamson. Series Elastic Actuators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-95)*, volume 1, pages 399–406, Pittsburg, PA, July 1995.

[46] Jerry Pratt. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots*. Ph.d., Massachusetts Institute of Technology, Cambridge, Massachusetts, 2000.

[47] Jerry E. Pratt. Virtual Model Control of a Biped Walking Robot. Technical Report AITR-1581, MIT Artificial Intelligence Laboratory, 1995.

[48] T. Prescott, P. Redgrave, and K. Gurney. Layered control architectures in robots and vertebrates. *Adaptive Behavior*, 7:99–127, 1999.

[49] A.M. Ramos and I.D. Walker. Raptors-Inroads to Multifingered Grasping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 467–475, 1998.

[50] Brian Scassellati. *Foundations for a Theory of Mind for a Humanoid Robot.* PhD thesis, Massachusetts Institute of Technology, 2001.

[51] D.M. Wolpert S.J. Blakemore, C.D. Frith. The cerebellum is involved in predicting the sensory consequences of action. *NeuroReport*, 12(11):1879–1884, July 2001.

[52] V. Stephan and H.M Gross. Neural anticipative architecture for expectation driven perception. *IEEE International Conference on Systems, Man, and Cybernetics*, 4:2275–2280, 2001.

[53] R.S. Sutton. Learing to Predict by Methods of Temporal Differences. *Machine Learning*, 3(1998):9–44, 1988.

[54] W. T. Townsend and Salisbury. *"Robots and biological systems : towards a new bionics?"*, chapter Mechanical design for wholearm manipulation. Springer-Verlag, 1993.

[55] William Townsend. The BarrettHand Grasper. *Industrial Robot: and International Journal*, 27(3):181–188, 2000.

[56] Todd W. Troyer and Allison J. Doupe. An associational model of birdsong sensorimotor learning i. efference copy and the learning of song syllables. *JNPHY*, 84(3):1204–1223, 2000.

[57] E. von Holst and H. Mittelstae. *The Behavioural Physiology of Animals and Man: The Collected Papers of Erich von Holst*, chapter The reafference principle. (1950). University of Miami Press, Coral Gables, FL, 1973.

[58] Matt Williamson. Neural control of rhythmic arm movements. *Neural Networks*, 11(7-8):1379–1394, 1998.

[59] A. Witney, S.J. Goodbody, and D.M. Wolpert. Learning and decay of prediction in object manipulation. *Journal of Neurophysiology*, 84:334–343, March 2000.

A The Robot Platform

Domo, as pictured in Figure 1, has 29 active degrees of freedom (DOF), 58 proprioceptive sensors, and 24 tactile sensors. 22 DOF use force controlled and compliant actuators. There are two six DOF force controlled arms, two four DOF force controlled hands, a two DOF force controlled neck, and a seven DOF active vision head. The real-time sensorimotor system is managed by an embedded network of five DSP controllers. The vision system includes two FireWire CCD cameras [25] and utilizes the *YARP* [16] software library for visual processing. The cognitive system runs on a small, networked cluster of PCs running the Linux operating system.

A.1 Head

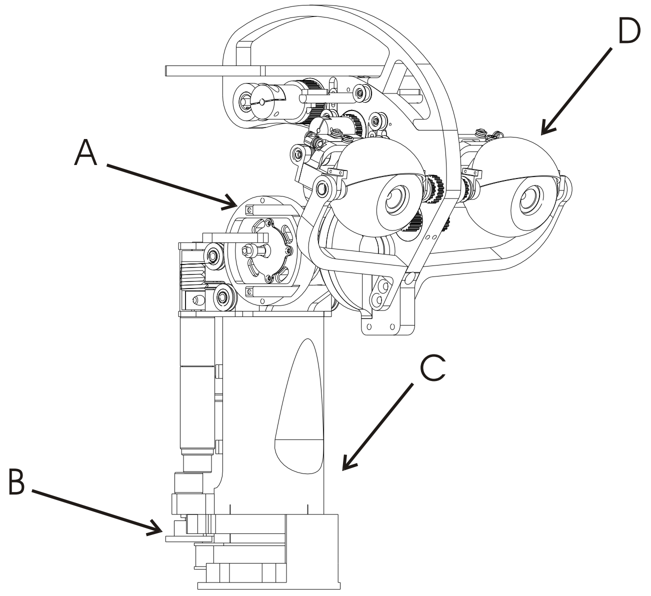


Figure 12: *Mechanical drawing of* Domo*'s active vision head. A SEA driven universal joint (B) combined with head pan (C) provides a compact ball-and-socket like three DOF neck. The upper neck provides roll and tilt through a cable-drive differential (A). Two FireWire CCD cameras (D) share a single tilt DOF and have two independent pan DOF. Expressive eyelids provide a final DOF.*

The design of *Domo*'s active vision head is an evolution from previous designs used for the robots Cog and Kismet [9, 5]. It is a copy of the head used in a new active vision project by Aryananda [3].

The head features 7 DOF in the upper head, a 2 DOF force controlled neck, and a stereo pair of FireWire CCD cameras. The upper head provides roll and tilt through a compact cable-drive differential. The two cameras share a single tilt DOF and have 2 independent pan DOF. The head also features one DOF expressive eyelids.

The head uses brushed DC motors with both encoder and potentiometer position feedback. The potentiometers allow for absolute measurement of position at startup, eliminating the need for startup calibration routines. The analog signal from the CCD cameras is digitized to a FireWire interface on boards mounted in the head, reducing noise issues related to running the camera signals near the head's motors. Physical stops are incorporated into all DOF to safeguard the head against potential software failures.

Unlike Cog's head which used one wide and one foveal camera per eye, *Domo* uses a single wide angle camera per eye. The camera used is the Point Grey OEM Dragonfly [25]. The cameras are IEEE-1394 FireWire devices with 640x480 (30fps) or 1024x768 (15fps) 24-bit color resolution. We use a 2mm focal length lens. The cameras are powered by the FireWire bus and provide software based synchronization of the dual framegrabbers.

Our primary design consideration for the upper portion of the head is that it be able to execute human-like eye movement. Human eye movements can be classified as: saccades, smooth pursuit, vergence, vestibulo-ocular reflex, and the optokinetic response [28]. *Domo*'s head is designed to accommodate all but the optokinetic response. Saccades require fast, ballistic movements of the eyes ($900 deg/s$) while smooth pursuit requires slow, controlled tracking movements of less than $100 deg/s$. Vergence requires independent control of the eye pan to view objects of varying depth. The vestibulo-ocular reflex requires either a head mounted gyroscope or correct kinematic information (we currently use the latter). Accommodating these features required careful design of the eye drive system and motor selection. We use a small gearhead motor (Maxon 8mm 0.5W with 57:1 gearhead) and an efficient cable-drive system for the eye pan.

A.2 Arms

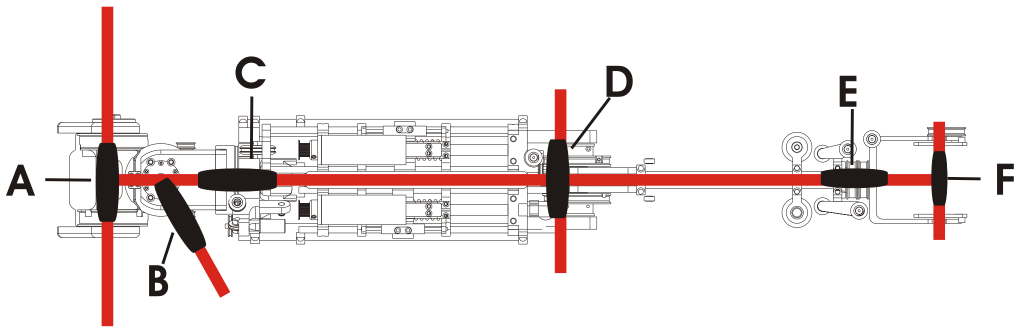


Figure 13: *The kinematic structure of* Domo*'s arms. A compact cable-drive differential at the shoulder provides pitch (A) and roll (B). These two DOF are driven by actuators placed in the robot torso. The bicep of the arm contains four other actuators: shoulder yaw (C), elbow pitch (D), and wrist roll (E) and pitch (F) driven by two cables routed through the elbow.*

Traditional arm designs assume that end-effector stiffness and precision are necessary qualities. A central pillar of our design approach to *Domo*'s arms is that the manipulators must be passively and actively compliant and able to directly sense and command torques at each joint.

Each arm joint is driven by an SEA containing a brushless DC motor. SEAs and FSCAs are covered in Section 2. As shown in Figure 13, a compact cable-drive differential at the shoulder provides pitch and roll. These two DOF are driven by actuators placed in the robot torso. The bicep of the arm contains another four actuators for shoulder yaw, elbow pitch, wrist roll, and wrist pitch. The drive-cables for the wrist actuators are routed through the center of the elbow joint.

Domo's arms are based on a cable-drive design similar to that of the WAM arm [54]. Cable-drive systems have significant advantages. They provide higher efficiency and lower backlash than spur geartrains. They allow us to place the actuators away from the driven joints. Much of the actuator mass, which dominates the total arm mass, can be moved off of the arm or as close to the shoulder as possible. This decreases the effect of the mass on the arm dynamics during ballistic movements. It also lowers the overall energy consumption of the arm by creating a very lightweight arm which in turn can use lower wattage motors. The cable-drive design also allows us to take a modular approach to the actuator design. We achieved a more efficient and standardized packaging of the actuators than usually possible with a direct-drive approach. A typical disadvantage of a cable-drive system is that a long cable acts as a stiff spring, limiting the end effector stiffness. *Domo*'s arms, however, are not stiff due to their inherent series elasticity, making this disadvantage negligible.

Custom brushless motor amplifiers and sensory signal amplifiers are embedded throughout the arm. The physical distribution of the actuator electronics minimizes wiring run-length and simplifies cable routing. The shoulder and the wrist have hollow centers, allowing for electrical cable routing through the center of joint rotation. This increases the electrical robustness of the arm by limiting cable strain and reducing the risk of snagging the cables during movement.

High backdrivability is often desirable in humanoid arms. It decreases the susceptibility to geartrain damage and facilitates human interaction. Our experience with Cog has shown that the motors in highly backdrivable arms require greater power and are prone to heat damage because the arm must hold itself up against gravity. A manipulator with low backdrivability can hold itself up against gravity indefinitely, but is prone to geartrain damage and is cumbersome for human interaction. *Domo*'s arms are statically non-backdriveable. However, force sensing allows them to be actively backdriveable while holding static postures with only a few watts of power consumption.

A.3 Hands

Hands for humanoid robots are notoriously difficult to design. Humanoid arms often impose constraints on the size, weight, and packaging of the hand while demanding sufficient dexterity, strength, and speed. These hands often lack the ability to directly sense the force applied by the actuator. They also tend to lack the mechanical robustness necessary for use in unstructured and unknown environments where impacts and collisions are common.

Humanoid hands typically use tactile sensors or load cells at the fingertips to gain force knowledge during manipulation. For example, the NASA Robonaut hand [32] utilizes Force Sensing Resistors to sense the pressure at the fingers.The Gifu Hand [22] employs a combination of load cells and tactile sensors, and the Dexter hand [21] utilizes 6-axis force cells in each finger tip. In either case, knowing only the fingertip forces may be insufficient when precise knowledge of the manipulating environment is not available. The fingertip position must be such that the force sensor makes contact with the object for the sensor to be useful.

In contrast, a controller that can command finger joint torque is able to execute a grasp with much less accurate information. The finger need only close with a desired force and the joint position will be determined by the object being grasped.

Each of *Domo*'s hands contains four modular FSCAs acting on three fingers, as shown in Figure 14. One actuator controls the spread between two fingers. Three actuators independently control

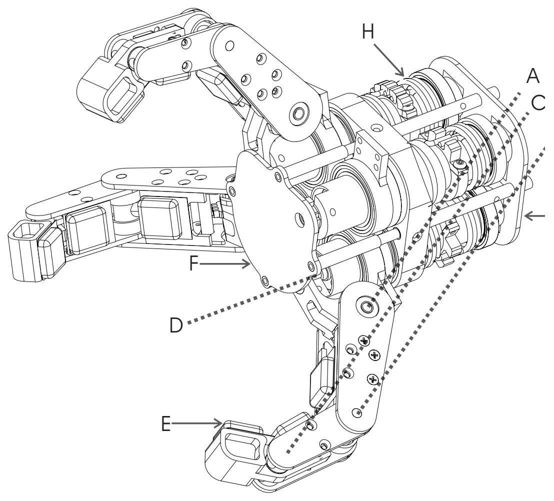


Figure 14: *Schematic drawing of the hand: Each of three fingers has three joints (A,B,C). Joint A is driven by an FSCA (H) through a cable drive. Joint B is passively coupled to A through a rigid cable drive. Joint C is passively linked by a compression spring to B. The spread between two of the fingers (about axis D) is driven by an FSCA I. The interior surface of each link in a finger has a tactile sensor (E) and the palm has an array of tactile sensors (F). Electronics for motor drive, sensor conditioning, force sensing, and controller interface reside at the rear of the hand (G).*

the top knuckle of each finger. The lower knuckles of the finger are passively coupled to the top knuckle. The passive compliance of the FSCAs is advantageous. It allows the finger to better conform to an object through local, fine-grained adjustments of posture.

The three fingers are mechanically identical, however two of the fingers can rotate about an axis perpendicular to the palm. These axes of rotation are mechanically coupled through gears, constraining the spread between the two fingers to be symmetric. By controlling the spread between two fingers, we can create a large variety of grasps. Force control of the spread allows for local adjustment of grasp by simply allowing the fingers to find a local force minimum.

The overall size, force capacity, and speed of the hand roughly conform to that of an human adult hand. We have modelled the kinematic structure on the Barrett Hand [55] which has demonstrated remarkable dexterity and grasping versatility.

A.4 The Sensorimotor System

The design of *Domo*'s sensorimotor system emphasizes robustness to common modes of failure, real-time control of time critical resources, and expansibility of computational capabilities.

These systems are organized into four broad layers: the *physical layer*, including sensors, motors, and interface electronics; the *DSP layer*, including real-time control; the *sensorimotor abstraction layer* providing an interface between the robot and the cognitive system; and the *cognitive layer*.

The first two layers are physically embedded on the robot while the latter two are processes running on the Linux cluster. The *cognitive layer* is the subject of Section 4.

A.4.1 Physical Layer

The *physical layer* is made up of the electromechanical resources physically embedded in the robot. This includes: 12 brushless DC motors and amplifiers in the arms, 17 brushed DC motors and amplifiers in the hands and head, a force sensing potentiometer at each of the 22 force controlled joints, a position sensing potentiometer at each of the 29 joints, a position sensing encoder at each of the 7 joints in the upper head, and an array of 12 FSR tactile sensors in each hand.

Our primary considerations in the design of the *physical layer* are the electromechanical robustness over time and the reduction of sensor noise.

Potentiometers tend to be noisy sensors. Each potentiometer sensor and FSR sensor on *Domo* has a signal conditioning amplifier mounted near the sensor. The amplifier provides both signal amplification and a passive low-pass filter to help reduce signal noise that may be picked up from long cable runs near the motors. The brushless motors used throughout the arms, where *Domo* encounters the longest cable runs, radiate less noise than brushed motors. In the head, the potentiometers are only sampled at startup to get the absolute position of the head for calibration, after which the encoders are used for position sensing. In the hand, we keep the cable run for the sensors as short as possible by placing the DSP controller node in the wrist.

The arm and hand motor amplifiers are embedded through out the arm (in contrast to the robot Cog, where they were kept off-board). The small form-factor custom amplifiers provide optoisolation between the motor and the controllers, 20Khz PWM switching, and current sensing and limiting. By embedding the amplifiers, we limit the noise creating inductive effect of switching high currents through long cables.

A.4.2 DSP Layer

The *DSP layer* provides real-time control over sensor signal acquisition and motor control. The layer currently incorporates five DSP nodes each running a 40Mhz Motorola DSP56F807. The nodes are mounted in the back of *Domo*'s torso. Each node communicates with the *sensorimotor abstraction layer* through a 1Mbps CAN bus channel. The number of nodes and the number of CAN bus channels is extensible as the sensorimotor requirements of the robot increase.

By using embedded DSP controllers, we gain complete control over the real-time aspects of the sensorimotor system. This avoids pitfalls commonly encountered when using PC based controllers, which can suffer from operating system timeouts and complicated startup routines. In contrast, the DSP controller starts up with a single switch and can safely function as a stand-alone unit.

Each DSP node controls up to eight joints in a 1Khz control loop. It also reads up to 16 analog sensor signals at 1Khz. The arm and hand nodes provide force control loops while the head node provides position and velocity control. Higher level controllers are implemented in the *sensorimotor abstraction layer*. In case of a failure, each node can choose to switch to a default safety mode. For example, if the CAN bus is disconnected, the arms switch into a zero-force mode, protecting them from moving people and objects in the environment. The arms resume activity when the CAN bus is plugged back in.

Each DSP node also performs some computation at 1Khz, such as implementing digital filters and calculating joint velocities. At a lower rate of 100hz it then communicates with the *sensorimotor abstraction layer*, relaying sensor values while accepting controller gains and commands.

A.4.3 Sensorimotor Abstraction Layer

The *sensorimotor abstraction layer* consists of a set of daemons running on a Linux node. It provides a substrate for the *cognitive layer* to execute on. The layer implements less time-critical motor controllers, interfaces with the CAN bus and the FireWire framegrabbers, and provides interprocess communication (IPC) infrastructure.

YARP [16] is a robot software platform developed in our lab. It enables message based IPC distributed across multiple Linux nodes. With *YARP* we can dynamically load processes and connect them into an existing set of running processes. It allows us to communicate data at a visual frame rate of $30hz$ and a sensorimotor frame rate of $100hz$.

A motor daemon interfaces with the CAN bus and polls the *DSP layer* at $100hz$ for sensory information. It implements a set of control modes available to each joint and loads the appropriate controller gains to the DSP nodes. A control mode may implement force control, joint position control, virtual spring control of joint position [47], joint velocity control, or control of the force at the end-effector. This set of control modes is made available to the *cognitive layer* through *YARP* IPC. The daemon also provides safety checks on all commands to the DSPs and places a DSP in a safety mode in case of a failure.

A vision daemon interfaces with the FireWire framegrabbers and provides visual data to the *cognitive layer* at $30fps$. It implements low-level motor controllers for the eyes, including tracking, saccades, smooth pursuit, vergence, and the vestibulo-ocular reflex.