

A FRAMEWORK FOR LEARNING AND CONTROL IN INTELLIGENT HUMANOID ROBOTS

OLIVER BROCK^{*,†}, ANDREW FAGG^{*,‡}, RODERIC GRUPEN^{*,§}, ROBERT PLATT^{*,¶},
MICHAEL ROSENSTEIN^{*,||} and JOHN SWEENEY^{*,**}

*Laboratory for Perceptual Robotics, Department of Computer Science,
University of Massachusetts, Amherst, MA, 01003, USA*

[†]oli@cs.umass.edu

[‡]fagg@cs.umass.edu

[§]gruppen@cs.umass.edu

[¶]rplatt@cs.umass.edu

^{||}mtr@cs.umass.edu

^{**}sweeney@cs.umass.edu

Received 2 October 2004

Revised 8 March 2005

Accepted 14 March 2005

Future application areas for humanoid robots range from the household, to agriculture, to the military, and to the exploration of space. Service applications such as these must address a changing, unstructured environment, a collaboration with human clients, and the integration of manual dexterity and mobility. Control frameworks for service-oriented humanoid robots must, therefore, accommodate many independently challenging issues including: techniques for configuring networks of sensorimotor resources; modeling tasks and constructing behavior in partially observable environments; and integrated control paradigms for mobile manipulators. Our approach advocates actively gathering salient information, modeling the environment, reasoning about solutions to new problems, and coordinating *ad hoc* interactions between multiple degrees of freedom to do mechanical work. Representations that encode control knowledge are a primary concern. Individual robots must exploit declarative structure for planning *and* must learn procedural strategies that work in recognizable contexts. We present several pieces of an overall framework in which a robot learns situated policies for control that exploit existing control knowledge and extend its scope. Several examples drawn from the research agenda at the Laboratory for Perceptual Robotics are used to illustrate the ideas.

Keywords: Humanoid control; motion planning; human–robot interaction; grasping; hierarchical systems.

1. Introduction

Humanoid robots are designed to interact with humans in human environments and to act as surrogates in environments where humans choose not to go. Future applications include residential service, elder care, agriculture, military logistical

*Authors are listed in alphabetical order.

support, and the exploration of space. These are dynamic environments that are only partially observable/controllable and in which robots must address multiple, changing objectives — we call such environments “open” systems. Our view is that significant research effort should be focused on how robots can function effectively as components of such complex systems. This includes issues of knowledge and representation, programmability, and control that are inter-related and must be capable of accommodating the special circumstances characteristic of this domain.

Several technical challenges must be met to achieve human–humanoid systems. Humanoid robots typically afford significant redundancy in the way that tasks may be accomplished. Not only is there redundancy in the number of distinct alternatives for solving a task (e.g. grasp an object with either the left or right hand), but within each of these distinct choices, there are often excess mechanical or sensory degrees of freedom. In this case, redundancy can be used to satisfy additional objectives such as optimizing posture, energy utilization, or to support multiple subtasks simultaneously. An individual, or society of individuals, can use this flexibility to plan for contingencies to address environmental uncertainty and stochasticity.

Open environments require humanoid robots to learn and plan at run-time and in novel situations. Therefore, mechanisms for exploring possible solutions are required that ensure the safety of the machine and surrounding environment. Models of the consequences of action can support model checking techniques to prove that functional constraints are satisfied. In addition to safety, constraints can be used to focus exploration on fruitful states and actions.

Additionally, abstraction can provide a means of conditioning the search for action plans. This approach has received a great deal of attention from the artificial intelligence community for decades. Macros and “chunking” mechanisms,^{37,45} policy options,^{53,62} schemata,^{1,14} and temporally-extended actions²⁸ have all been justified on this basis. These techniques permit learning and planning systems to reason about solutions at several spatial and temporal levels. In this work, we propose a form of behavioral abstraction as an expressive language for user-level programming, autonomous planning and learning, and as a mechanism for generalizing plans new situations.

In this paper, we present an integrated framework for programming, learning, representation, and execution of skills by robotic systems motivated by these scientific goals. Humanoid robots are treated as local collections of resources situated in open environments and embedded in *ad hoc* networks of sensorimotor resources. These information networks can include humans and other robots. We consider how collections of resources can be configured on-line to accomplish tasks. To do so, a combinatoric basis for closed-loop control is proposed from which a robot learns to construct behavior (Sec. 2). We address discrete abstraction and environmental affordances in this framework and provide a means of configuring multi-objective controllers using null space projections. To create practical robot systems, we describe structure in the form of constraints that condition learning and planning. Model checking techniques are employed to eliminate unsafe states

and actions and to stage learning problems (Sec. 3). By exploiting this structure, we illustrate how hierarchical behavior can be constructed and propose that control can be factored into declarative and procedural components in order to explore generalization and re-use. In Sec. 4, we illustrate many of these ideas in a sequence of case studies from grasping and manipulation including grasp control, haptic categorization, whole body grasping tasks, and learning by imitation for mixed-initiative controllers. Finally, Sec. 5 discusses hybrid methods that exploit behavioral abstraction to support prediction and planning in this framework.

2. A Combinatoric Basis for Behavior

Open environments introduce possibly infinite varieties of disturbances that change from one instance of a task to the next. Moreover, humanoid robots can typically implement many possible solutions to common problems and may, therefore, respond resourcefully to run-time feedback. To take advantage of this capability, we propose a primitive organizational structure in the form of parametric closed-loop controllers.

Our framework starts by considering closed-loop control because: (i) nearly all action in robot systems eventually takes the form of closed-loop control; (ii) it provides a basis for error suppression and robustness in stochastic environments; (iii) it is well suited to multi-objective frameworks, and (iv) it supports descriptions of the coupled robot-world behavior as dynamical systems. This last feature is an important aspect of knowledge and representation that is couched in the underlying continuous-time behavior of control systems.

Several research efforts have explored the use of closed-loop control to represent primitive actions for robot systems.^{31,34,35} Our approach extends these approaches by explicitly factoring controllers into *objectives*, which capture the declarative structure of the solution, and the implementation of the solution in terms of sensors and effectors.²⁸ This latter component speaks to the procedural details of a strategy and allows the robot to consider solving the same kinds of problems in many different ways depending on the context.

2.1. The control basis

Primitive control is defined in the form of fixed, closed-loop response to specific stimuli by engaging specific effector resources. A particular closed-loop controller serves a single objective. For this reason we will sometimes refer to single, closed-loop controllers as *segmental actions*. This definition is consistent with descriptions of the segmental organization of the spinal cord. These reflexes are specific stimulus-response mappings in service to a single objective like the *withdrawal* reflex that extracts one's hand from a fire. They can be coupled in intersegmental arrangements such as the contralateral *extension* reflex that can accompany the withdrawal reflex. These arrangements serve two, concurrent objectives: one to extract a limb receiving a painful stimulus, and another to extend the other limb in a protective behavior.

Although our primitives are exclusively closed-loop, segmental reflexes in the spinal cord need not be.

The control basis is a combinatoric basis for all such segmental responses written $(\Phi \times 2^{\mathcal{S}} \times 2^{\mathcal{E}})$. A control design relates objectives, $\phi \in \Phi$, with combinations of sensations, $s \in \mathcal{S}$, and effectors, $e \in \mathcal{E}$. Unlike traditional control design methodologies, there is no commitment, at this stage, to predefined sensorimotor primitives for given tasks. Instead, the control basis generates all behavior supported by the sensorimotor apparatus and the objectives native to the robot. We will begin by defining the three components of a controller in more detail and then describe our approach to creating practical, adaptive optimal, learning robots.

2.1.1. *Sensations* — s

It is often impossible to identify features *a priori* that are globally meaningful in open systems. Therefore, we feel that it is important not to commit to a fixed set of special purpose operators. Instead, our approach is to employ an adaptive perceptual system that generates its own features by composing appearance-based primitives into high-level features that serve as inputs to controllers from the control basis.^{48,50}

There are several alternatives that meet this design goal in the literature. One such approach is the so-called “N-jet” that employs a set of multi-scale Gaussian derivative filters.³⁶ A vector of filter responses at various scales centered at the characteristic scale^{39,65} are used to capture local structure in the signal. Gabor filters²⁰ and wavelets⁶¹ are other options for describing signal shape.

In previous work, we have used primitive visual features that are local, oriented, and appearance-based. Oriented derivatives of 2D Gaussian functions were used to form a steerable (and rotationally invariant) basis by normalizing for orientation on the image plane.⁵⁵ A vector of filter responses from several Gaussian derivative operators at multiple scales is called a *texel*. Texels can be extended to incorporate other attributes (color, depth) using the same techniques and generalized to other types of signals (acoustic, force). On the image plane, texels are oriented by steering the responses of the first derivative operators. Spatial combinations of these primitives can express a wide variety of shape and texture characteristics at various degrees of specificity. For example, we combined oriented texel features on the image plane using geometric, topological, conjunctive, and disjunctive relations between features to improve specificity.⁴⁸

Constructing such features is a search process that aims to optimize the discriminative power of the composite feature measured in terms of the Kolmogorov–Smirnov distance between two conditional distributions of a random variable.⁴⁹ The result of the search is incorporated into a Bayes net classifier for estimating the conditional probabilities of important observable categories. These categories can denote visual recognition policies⁵⁰ or can serve as visual affordances that act as sensor references, s , for controllers in the control basis. This latter approach

avoids segmentation, object recognition, or the detection of pre-designated “salient” features^{30,42,59} — it compares signals directly. The learning framework attempts to construct appearance-based “queries” in the form of vectors of filter responses that constitute reliable references for control tasks.

2.1.2. Objective functions — ϕ

Primitive objectives, $\phi \in \Phi$, are scalar potential functions that map independent configuration space variables to real numbers. They represent the native objectives of the system, which can be realized in many unique implementations depending on how sensors and effectors are employed. In general, sensations, s , define goals and obstacles relative to the current state of the robot and thus influence the shape of the potential field. We have explored the use of domain general primitive objective functions in order to guarantee that the robot is not constrained to specific strategies for anticipated tasks, but is instead capable of adapting to the run-time environment using all the sensory and motor facilities it can bring to bear.

We have constructed potential fields that describe the *error* of a kinematic configuration with respect to:

- (i) the forces and moments that contacts with the environment can transmit;⁹
- (ii) the “hitting probability” determined from neighboring obstacles and goals;¹²
and
- (iii) the potential for generating force and velocity in arbitrary kinematic chains (often described as kinematic conditioning).²³

A brief note on notation: a subscripted ϕ denotes either an effector resource allocation (ϕ_e) or merely a label for the objective function (ϕ_{label}); the usage should be clear in context.

2.1.3. Effectors — e

The most primitive effector resources are subsets of the humanoid robot’s actuated configuration space. These subsets can include series and parallel kinematic chains and constitute the independent variables for the native potential functions.

To optimize objective ϕ , one can use a greedy descent of the form $\Delta e = -K * \nabla_e \phi$, where

$$\nabla_e \phi = \frac{\partial \phi(s)}{\partial e}.$$

The shape of the potential function depends on both the input and output resources engaged. For any given objective function, the control basis provides for many different choices for generating the input stimuli and output effector and the value of these choices will vary with operational context. The action derived from $\phi_e(s)$ defines a segmental coordination policy for the participating resources. As a trajectory unfolds, the sensitivity of the potential to individual configuration

variables changes, so that effector coordination changes over time in service to the governing objective.

2.2. Discrete abstraction of continuous time

The closed-loop processes described in Sec. 2.1 respond essentially to continuous inputs and generate continuous outputs. The decision about which segmental controller to apply to the environment, however, is dependent on the broader context of the open system. Since open systems can introduce significant variation in operating context, acquiring enough information to make optimal control decisions is considerably harder. Open systems exacerbate problems associated with hidden (or missing) state. However, several results suggest a partial solution to this problem. Takens’s theorem describes how patterns in the behavior of deterministic dynamical systems are related to missing or hidden state variables.⁶⁴ Likewise, solutions to Partially Observable Markov Decision Processes (POMDPs)³³ rely on observing the system over time, and Hidden Markov Models⁴³ can recognize categories in signals by parsing sequences of observable events according to a robust transition model. In each of these approaches, observing the system over time is the key.

In previous work, assertions about the stability of the coupled system have been used to form the state space for such systems as in the attractor landscape proposed by Huber²⁸ or the limit cycles proposed by Schaal.⁵⁸ Generalizations of these ideas can be used to advantage in our framework as well. Here, closed-loop controllers interact with the environment over extended periods of time and interaction dynamics reveal important information about the coupled system. The approach is illustrated schematically in Fig. 1, where a simple asymptotically stable controller from the control basis is depicted. The center panel of Fig. 1 plots the potential, $\phi(t)$, against the time rate of change of the potential computed by the controller as it approaches fixed points in the field.¹¹

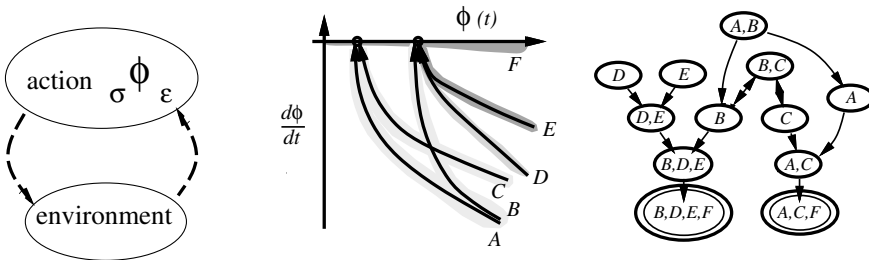


Fig. 1. Left: an element of the control basis interacts with a partially observable environment as a coupled system. Middle: this interaction produces a time series represented on a phase portrait. When run-time experiments match dynamic models (labeled A through F), the hidden environmental state can be recovered. Right: a discrete state estimate for this control process consists of the pattern of membership in these models.

Our hypothesis holds that a situated controller produces relatively few models that can be used to distinguish between important control contexts for that situation. For example, asymptotically stable controllers converge as time goes to infinity — a condition where $\frac{d\phi}{dt} \approx 0$ — represented by membership in model F . Here, the system has reached a fixed (equilibrium) point, or progress has halted for some other reason. This latter case could arise when external constraints (workspace restrictions, or resource failure) eliminates progress toward extrema in the potential function. The middle panel in Fig. 1 illustrates five other models, A through E , that show a trajectory toward convergence. Each model represents a unique basin of attraction exhibited when the controller interacts with the environment. For example, multiple models can be observed empirically when a grasp controller (Sec. 4) interacts with several objects. Only those objects that occur frequently in this control context must be modeled to inform control decisions.

Models such as these form a domain theory for the situated controller and several domain theories may be relevant for each controller — each describing a particular control context and each couched in the agent’s interaction with the environment. We call such information *interaction-based state*.

Given enough observation over a long enough period, patterns of membership in such models can be recognized and used to make optimal control decisions. The discrete transition model in Fig. 1 describes how the state of a situated controller can evolve over time. As such, it can be used to recover hidden state as well to predict the possible future outcomes of the control process.

2.3. Model checking — Discrete Event Dynamic Systems

The coarse-grained control over the combinatorics of exploration is provided by a Discrete Event Dynamic Systems (DEDS) specification. It constrains the range of interactions permitted with the environment to those that:

- satisfy real-time computing constraints;
- guarantee safety specifications; and
- are consistent with kinematic and dynamic limitations.

In this formalism,^{47,54,60} the state of the underlying system is assumed to evolve with the occurrence of a set of discrete events, some subset of which is controllable. There are many tools for analyzing and interacting with such control processes. One may prove, for instance, that certain states cannot occur. These tools also provide a means of investigating the role of constraints as “bootstraps” for a learning system. Such a mechanism influences the occurrence of controllable events such that no prohibited or uncontrollable event can violate functional constraints on the system. A complete supervisor takes the form of a nondeterministic finite state automaton in which states are functional assertions about patterns of membership in the empirical dynamic models that must be either preserved or excluded and transitions represent possible concurrent control situations.

In the context of the control basis approach, logical conditions on legal patterns of membership in the governing dynamic models (Sec. 2.2), influence the range of control options that the system may explore. In a similar fashion, this DEEDS supervisor allows the introduction of additional domain knowledge and preferences into the control architecture. Constraints expressed in the DEEDS-based task description and/or the resource model can be used to “teach” the system how to explain new concepts incrementally as in classical approaches to *shaping* and *maturation*.

2.4. Multi-objective control

A *redundant* system has excess sensory and motor degrees of freedom with respect to a particular task. Among the important properties of such systems is the opportunity to address multiple objectives simultaneously. The dominant mathematical framework for describing multiple objectives in systems with excess degrees of freedom employs the Moore–Penrose pseudoinverse.^{44,67} In these systems, there may be an infinite number of solutions and a particular solution can be selected based on a secondary criteria. Often, as in Moore–Penrose, solutions that produce economy of motion (like least-squares) are employed to recommend one particular solution among the alternatives. However, this means that homogeneous solutions exist as well and secondary objectives can be incorporated into the control design without disturbing the primary objective. To do so, secondary actions are projected into the null space of a linear system describing the primary goal. This has been demonstrated to optimize the posture of a manipulator while reaching to goals in configuration space,⁶⁸ to navigate in the presence of unmapped obstacles,¹² or multi-robot constraints,⁶³ and to optimize kinematic conditioning during walking²⁹ or grasping.⁵²

When control actions are derived from greedy descent on an artificial potential, it is a simple matter to approximate the scalar field locally and to describe the local equipotential manifold. Any action that projects exclusively into the equipotential manifold will cause no effect on the primary objective. The *subject-to* operator “ \triangleleft ” proposed by Huber makes use of null space projections to eliminate destructive interactions between control tasks in a multi-objective framework.²⁷ A pair of controllers, $\phi_{\text{sub}} \triangleleft \phi_{\text{sup}}$ (read ϕ_{sub} subject-to ϕ_{sup}) will descend the potential field of the superior controller and will superimpose only those components of subordinate actions that produce no change in the value of the superior potential. This approach is extended to handle multiple cascaded objectives in a straightforward way.

3. Structure for Policy Formation and Reuse

The control basis represents a basic organization of sensors and effectors through stimulus-response relationships, ϕ_e^s . As such, it defines the primitive instruction set available to the robot. The set of primitive actions that may be expressed, $\Phi \times 2^S \times 2^E$, is quite large. This is good from the perspective of expressive power

but bad from the standpoint of computational complexity and policy formation. While the system may be capable of associating novel combinations of features in the sensor data with unorthodox combinations of effectors to accomplish a specific stimulus-response process, it is not clear that these unlikely actions will ever be explored by purely randomized search.

A means of organizing the large space of control actions and control states is to exploit hierarchical structure in the form of repeated sub-tasks. Many tasks require that the solutions to more primitive subtasks be repeated several times in the pursuit of more abstract objectives. These subtasks may be captured in the form of sequences of primitive control instructions. Learning the solution to a task with this kind of structure can be represented using hierarchical sensorimotor policies.

Precompiled control knowledge in the form of control policies can be used as temporally extended actions with built-in contingencies for responding to run-time contexts by recruiting appropriate sensory and motor resources. The same potential for employing interaction to recognize run-time context in primitive controllers (Fig. 1) may be useful in configuring temporally extended actions. The important features of the artificial potential, ϕ_i , that collectively give rise to an *admissible* control law may, under some conditions, be applied to value functions describing temporally extended actions. Therefore, policies, $\pi \in \Pi$, can also be used to carve up the state space into basins of attraction and produce a stable closed-loop response.

For example, navigation of a legged robot is a task with such repeated temporal structure. Huber explored hierarchy in the control basis framework with “Thing,” a quadruped robot with a 12-dimensional configuration space.²⁷ The task is specified in the full configuration space of the robot. However, significant translational or rotational motion of a legged robot can only be accomplished by repeating similar leg motions that constitute a locomotion gait. Therefore, the facility for developing new navigation policies benefits from temporally extended actions for translational or rotational locomotion gaits. However, each instance of the temporally extended action need not result in the same sequence of leg motions. For example, the translation gait may react to information garnered during execution to avoid obstacles. The performance of learning systems that use temporally extended actions for rotation and translation was enhanced significantly relative to cases without access to such control knowledge.

4. Grasping and Manipulation

To this point, we have motivated a research agenda for humanoid robotics that addresses challenges introduced by open systems, and proposed mechanisms capable of dealing with variability and redundancy. We advocate a representation for primitive behavior that can be used to structure exploration in learning systems and facilitates generalization and re-use.

There is, perhaps, no better domain in which to test these hypotheses than that of manipulation systems — particularly, dexterous manipulation systems. This domain presents many plausible solution strategies for interacting with a wide assortment of objects. It requires rich models of manual interactions that emerge implicitly from multiple aspects of hierarchical grasping tasks. More than other robotics problems, it stresses the representation of control knowledge derived from a huge variety of training data involving tactile, haptic, visual, proprioceptive, and motor systems.

In this section, we present a comprehensive array of case studies in grasping and manipulation that demonstrate the approach presented in the paper thus far. They offer a strong argument for the applicability of our framework to a wide range of tasks. We introduce a control basis for grasping tasks and demonstrate a simple, robust policy for pick-and-place tasks. We present a means of modeling control dynamics to create haptic categories and visual affordances in order to inform control decisions. We apply these techniques to whole body grasping tasks in which contacts with the environment can occur with the entire surface of a humanoid robot. We show how manipulation tasks can be learned and how exploration can be structured by inferring the intention of human supervisory inputs. We propose a learning sequence that culminates in a means of learning to exploit dynamics. Finally, a means of reasoning about behavior by planning through abstract, forward models of situated behavior is discussed.

4.1. *Grasp control basis*

The dominant traditions in grasp planning require that the complete geometry of the object be determined *a priori* so that a planner can enumerate all possible grasps and sort them by quality, which may include: robustness to perturbation forces and errors in contact placement; the friction coefficient required to produce wrench closure; the degree of mobility and constraint in the grasped object; the configuration of the manipulator; and task constraints on the configuration before and after the grasp. The robot executes a solution near the top of the list.^{8,18,19,26,32,38,46}

In contrast, the control basis approach advocates searching for control configurations that produce high quality system dynamics. This is an on-line process that does not require complete object geometry and considers the full complement of sensory and motor resources. Grasp controllers displace contacts on the surface of an object with unknown geometry so as to descend grasp error functions.^{11,25} Tactile sensor information is used to compute a potential field and actions descend the gradient of the potential with respect to the contact configuration. This process is depicted in Fig. 2.

The objective of the grasp controller in Fig. 2 is to minimize the squared wrench residual ϵ :

$$\epsilon = \boldsymbol{\rho}^T \mathbf{M} \boldsymbol{\rho} = \left(\frac{1}{k} \sum_{i=1}^k \boldsymbol{\omega}_i \right)^T \mathbf{M} \left(\frac{1}{k} \sum_{i=1}^k \boldsymbol{\omega}_i \right), \quad (1)$$

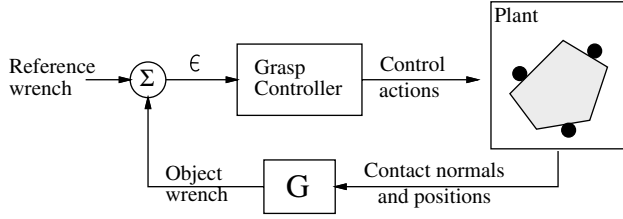


Fig. 2. Grasp synthesis as a control problem. The feedback transfer function, G , computes the net perceived wrench due to contact. The net wrench is compared with the reference wrench and appropriate contact displacements are generated.

where ρ is the wrench residual and expresses the net wrench over k contacts, \mathbf{M} is any normalization matrix that is positive definite, and ω_i is the object frame wrench resulting from the i th contact force. Without loss of generality, \mathbf{M} is assumed to be identity.

The squared residual, ϵ , is minimized by following the negative gradient of ϵ with respect to the contact configuration variable. Minima in Eq. (1) (configurations where $\frac{\partial \epsilon}{\partial \mathbf{q}} = 0$) are potential grasp solutions. To compute the first derivative, we require an expression for the contact wrench as a function of contact coordinates. This function depends directly on the geometry of the object. Two local surfaces types illustrated in Fig. 3 are considered for this purpose.

Force residual control is derived from a model of contact wrenches on a sphere. Moment residual control is derived from a model of contact wrenches on an infinite plane defined by the contact normal. The artificial potentials defined this way are

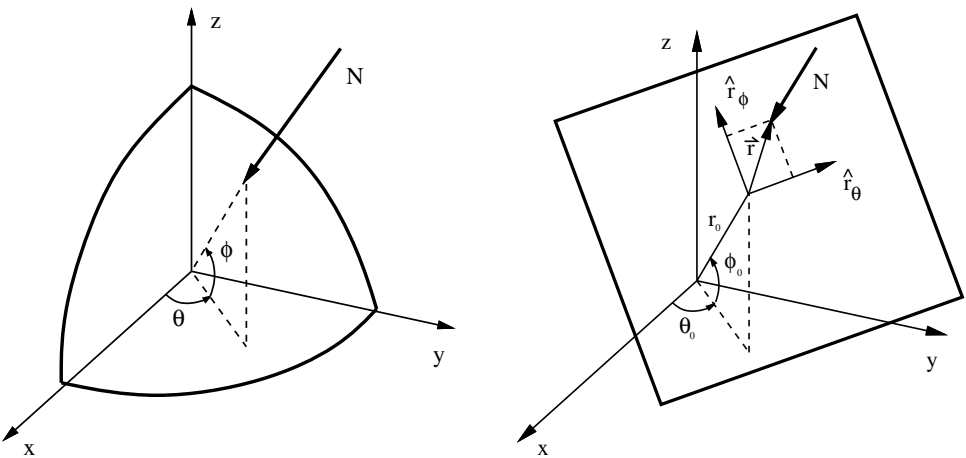


Fig. 3. Left: the residual force potential is derived from a local surface model of contact wrenches on a sphere. Right: the residual moment potential is derived from contact wrenches on an infinite plane defined by the contact normal.

unimodal so that gradient descent yields unique equilibria on the sphere and the infinite plane, respectively.^{11,24} However, on general closed surfaces there are potentially many minima that are low quality grasp solutions. We have previously shown that optimal contact configurations are achieved on regular convex objects by first minimizing the force residual (i.e. performing gradient descent on the residual force potential), and then minimizing the moment residual.⁹

4.2. *Intersegmental control for grasp conditioning*

Intersegmental controllers consider multiple objectives simultaneously to produce coordinated behavior. We have formulated a successful class of grasp control as the composition of three individual objectives; force conditioning, moment conditioning, and the overall kinematic condition of the hand. The independent objectives of grasp formation are combined by null space projection:

$$\phi_{\text{kinematic}} \triangleleft \phi_{\text{moment}} \triangleleft \phi_{\text{force}}, \quad (2)$$

which reads $\phi_{\text{kinematic}}$ subject to ϕ_{moment} subject to ϕ_{force} ,⁵² where the “subject-to” operator projects a subordinate control input onto the null space of a superior controller. This control expression controls the contact configuration to address a strictly prioritized set of objectives: force residual, followed by moment residual, and then kinematic conditioning of the hand. The kinematic conditioning controller has equilibria in configurations where the major axis of the finger’s velocity ellipsoid⁶⁸ is aligned with the contact normal. This optimizes the manipulator for precise, controlled displacements tangent to the object’s surface. The concurrent controller thus preserves the priority of objectives and the asymptotic stability of the participating controllers.

4.3. *Pick-and-place tasks*

Pick-and-place tasks involve localization, collision-free reaching tasks, force closure, transport, and placement of an object. These tasks are typically treated as if they are independent skills, but since they occur in concert in virtually every manipulation scenario, it is important to consider methodologies that can stitch them together appropriately.

In one experiment,⁹ we explored several pick-and-place tasks involving wooden blocks randomly placed on a table top using a Utah/MIT hand mounted on a GE P50 robot arm (see Figs. 5 and 6). Objects in the study were convex, regular and irregular prisms with polygonal cross sections (triangles, rectangles, pentagons, and hexagons) approximately 20 cm in height and with diameters of roughly 5 cm. A calibrated camera mounted directly overhead was used to localize objects and places on the table top. A user selected objects by clicking a mouse on the image from this camera, first to designate a target object, and second, to identify the position where the object should be placed on the cluttered table top. The arm and hand comprise 21 degrees of freedom that are controlled without further user input and

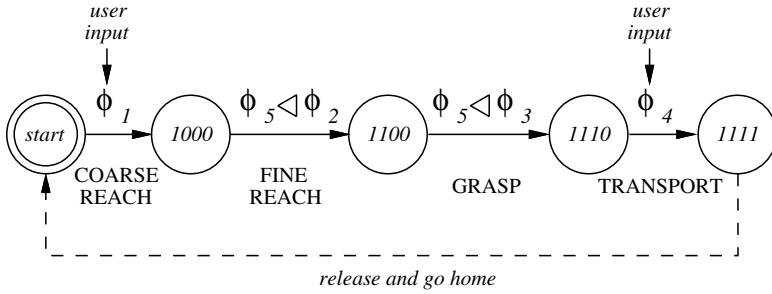


Fig. 4. The PICK-AND-PLACE policy in the form of a finite state automaton with high-level user inputs, and the individual motor schemata with which it is implemented. Five primitives (ϕ_1 to ϕ_5) are used to create four unique actions: COARSE REACH (ϕ_1), FINE REACH ($\phi_5 \triangleleft \phi_2$), GRASP ($\phi_5 \triangleleft \phi_3$), and TRANSPORT (ϕ_4). Each state description represents the convergence status of each of these four controllers; 1 asserting near equilibrium, and 0 otherwise.

without models of the objects according to the policy illustrated in Fig. 4. Three types of primitive controllers are employed (Sec. 2.1.2), including a version of the grasp controller described in Eq. (2). Five primitives (ϕ_1 to ϕ_5) for pick-and-place tasks are defined and are used to construct four unique actions shown in Fig. 4. The state descriptions in each node of Fig. 4 represent the status of these concurrent controllers — a 1 asserting that the corresponding controller is near equilibrium and a 0 asserting that it is not.

The pick-and-place cycle starts as the user specifies a target object by clicking on an overhead visual image of the robot’s workspace. The selection completes the specification of COARSE REACH implemented using ϕ_1 — a harmonic function path controller. This action controls a collision-free trajectory through the cluttered workspace to the neighborhood of the specified object using five degrees of freedom in the arm while 16 DOF in the hand remain fixed.

Following convergence of ϕ_1 , the policy executes a FINE REACH schema. This is a multi-objective control mode that uses the “subject-to” operator (Sec. 2.4) to combine a kinematic conditioning controller, ϕ_5 , in the null space of motion controllers, ϕ_2 . The qualitative effect of FINE REACH is the independent motion of each finger toward the object’s surface until a tactile goal is achieved for each fingertip. While these concurrent tactile *searches* are underway, controller ϕ_5 evaluates the collective kinematic condition of the hand and creates arm movements that optimize manipulability given the relative fingertip positions.

Next, the GRASP schema is engaged. GRASP is similar to FINE REACH except that the wrench closure objective determines the independent finger movements using ϕ_3 rather than a tactile search. Once again, ϕ_5 is a subordinate objective that engages the arm to optimize the kinematic condition of the hand while the grasp is formed. The resulting multi-objective controller propagates constraints imposed by the grasping task, through the hand posture, to the arm posture. This control sequence concludes with the arm and fingers in a grasp configuration on the object.

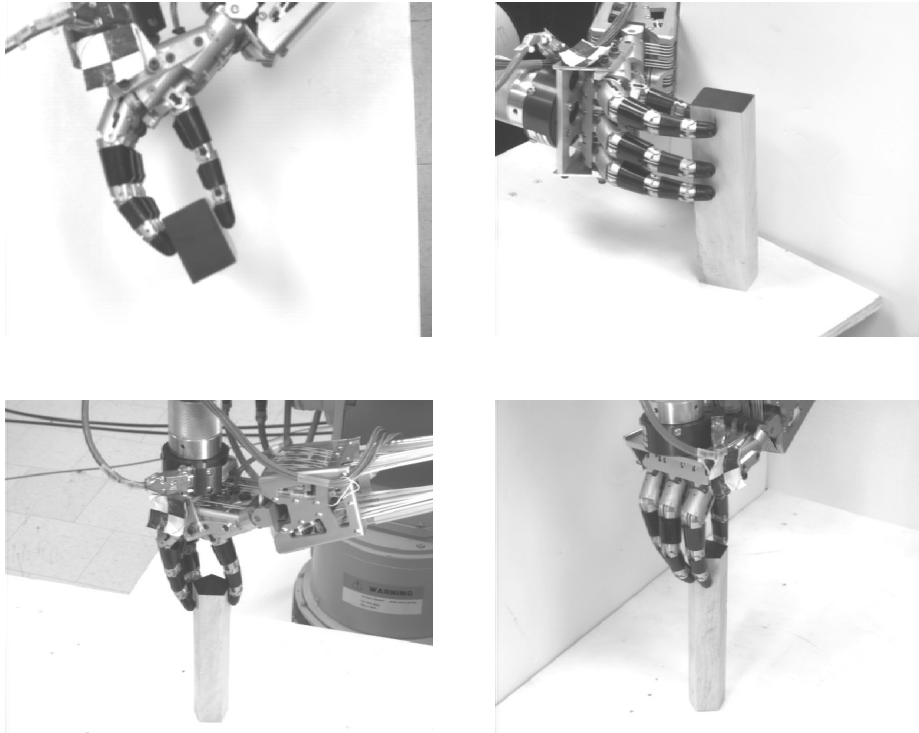


Fig. 5. Grasps synthesized by PICK-AND-PLACE and executed using a Utah/MIT hand. The top row shows two perspectives of the lateral grasp of a rectangular prism, while bottom row shows the overhand grasp of a pentagonal prism.

Figure 5 shows several examples of convergent contact configurations at this stage of the PICK-AND-PLACE schema.

In the final stage of a pick-and-place task, the user designates a target location for the object on the cluttered table top. This completes the specification of the TRANSPORT schema, ϕ_4 . If a path exists, the object/hand/arm tracks a collision-free trajectory to place the object at the target location where it is released. Figure 6 depicts the execution of a complete pick-and-place cycle.⁹

Our Utah/MIT hand does not incorporate tactile sensing. Therefore, contact positions and normals were estimated visually for this demonstration. Due to the absence of actual tactile feedback, contact displacements and corresponding tactile probes were executed over the visual model of the object, and the physical interaction between the manipulator and the object was limited to the execution of the end grasp configuration. When failures occurred, they often started with errors in the visual estimate of contact normal and object cross-section. As a consequence, the grasp would be imprecise and ultimately an inaccurate release would sometimes cause the object to wobble and fall at the target location. However, even in this

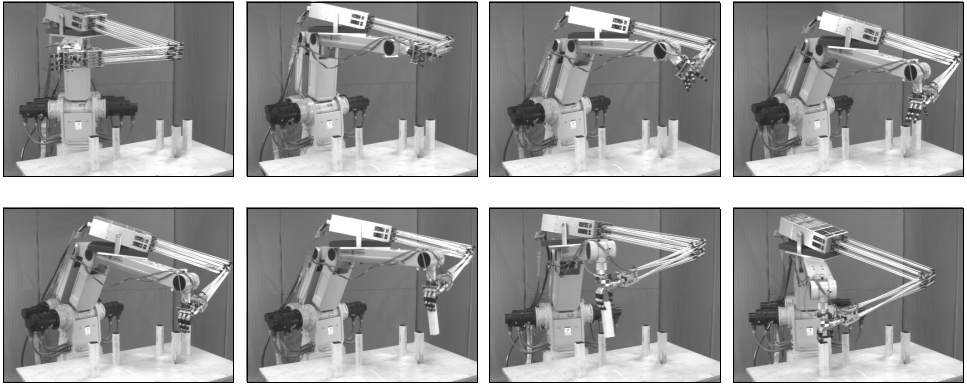


Fig. 6. A sequence illustrating the temporally extended behavior of the PICK-AND-PLACE schema.

situation, imprecision at one stage was suppressed by feedback control at subsequent stages to accomplish the task.

4.4. Haptic affordances

Many factors affect the outcome of a grasp: object shape and scale, friction between object surface and fingers, kinematic constraints, sensory information available, and the set of finger surfaces employed in the grasp operation. Our position is that these factors cannot, in general, be modeled adequately *a priori*. However, by observing a feedback time series during grasping operations, the impact of such information can be inferred. To illustrate this idea, we evaluated the value of estimating grasp state using the transient response of the situated moment residual controller (Sec. 4.1). To do so, each grasp configuration is associated with the observation $\mathbf{o} = [\epsilon \ \dot{\epsilon}]^T$, where ϵ is the squared moment residual and $\dot{\epsilon}$ is the control action resulting in the current situation. A sequence of such observations is generated by every grasping task, $\mathcal{O} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$.

Figure 7 is a specific example of the more general discussion of Fig. 1 summarizing the observed dynamics of the grasp controller for two contacts and an irregular triangle. The left panel in Fig. 7 shows this sequence for a typical two-contact grasp of an irregular triangle. The contact configuration at the attractor, (c), corresponds to a minimum of the squared moment residual ϵ , where control inputs, $\dot{\epsilon}$, are approximately zero. Three such basins of attraction (corresponding to the three unique combinations of two contacts on three faces) are shown in the right panel of Fig. 7.

Each basin of attraction constitutes a model: M_1 , M_2 , and M_3 , describing how observations evolve over time during an extended interaction between the grasp controller and the object. In this sense, grasp controllers are filters describing interactions with objects in terms of a real valued correlation to discrete prototypes, and models M_1 , M_2 , and M_3 describe the haptic interactions afforded by the object via

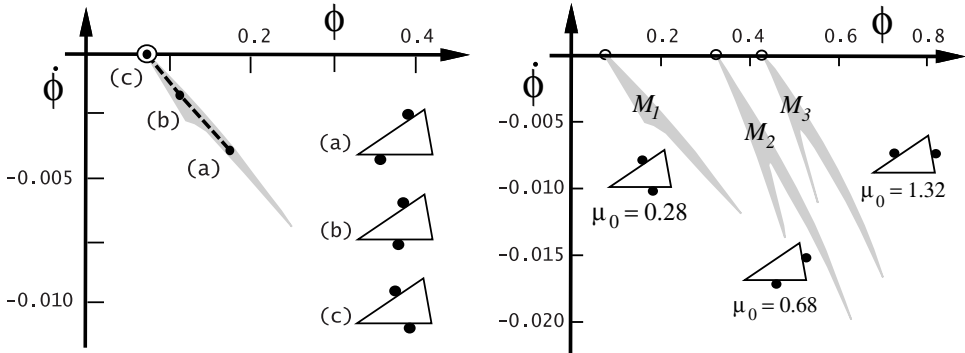


Fig. 7. Left: the evolution of a grasp trial: (a) is the initial configuration, (b) an intermediate configuration, and (c) the convergent configuration. The inset diagrams show the location of the contact points on the object in each configuration. Right: the complete phase space plot for the moment residual controller with two contacts and the irregular triangle. The inset diagrams show the convergent grasp configurations and associated μ_0 , the minimum friction coefficient required for force closure in the configuration.

two-fingered grasp controllers. Figure 7 also shows convergent grasp configurations and a measure of quality, μ_0 (the minimum friction coefficient required for force closure in this configuration) at each attractor.

The state description is based on two discrete sets: the control basis $\Phi = \{\phi_1, \phi_2, \dots, \phi_n\}$, that represents all unique contact parameterizations of the grasp controller, and the vector $\mathbf{q} = [p_1 \ p_2 \ \dots \ p_m]^T$ that indicates which of m models are consistent with observations derived from the active grasp process. Each p_i is a Boolean variable asserting that model i explains feedback observations ($p_i = 1$), or is independent of these observations ($p_i = 0$). The corresponding grasp state is denoted by the tuple $s_i = (\phi_i, \mathbf{q})$ instead of the continuous observation variables.

The discrete state space supports control decisions that lead to optimal grasp configurations. For a three-fingered robot hand, fingertip grasps designated using $\{T, 1, 2\}$ for thumb, first, and second finger, respectively, generate a set of four grasp controllers that employ at least two contacts:

$$\Phi_{\text{grasp}}(r) = \{\phi_{T,1} \ \phi_{T,2} \ \phi_{1,2} \ \phi_{T,1,2}\}. \quad (3)$$

Each element of $\Phi_{\text{grasp}}(r)$ specifies a unique combination of contact resources with which to measure positions and normals, estimate moment residual, and execute differential contact displacement. Each of these parameterizations, therefore, also produces different grasp interactions that depend on object geometry, hand kinematics, the number of contacts employed, and initial conditions.

Since there may be more than one object whose identity and pose may be unknown, in general, the grasping process must include actions that gather sufficient information to accomplish the task. We view grasping with incomplete state, therefore, as a sequential information gathering activity that culminates in an optimal

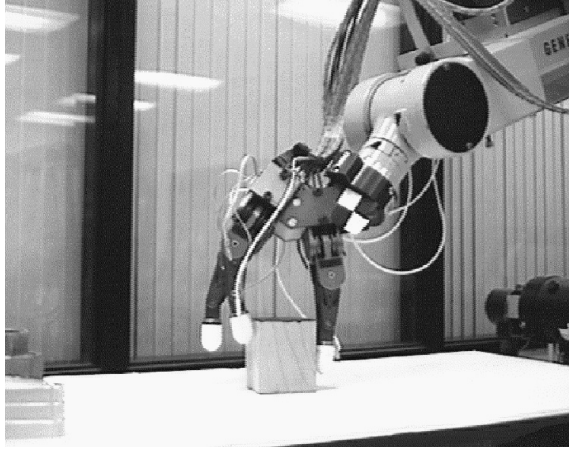


Fig. 8. The Stanford/JPL hand used to learn robust, adaptive grasping policies using interaction-based state observations.

force closure condition. The goal is to engage sequences of grasp controllers that recover critical information about the object and ultimately, accomplish the optimal grasp.

Coelho⁹ used these interaction-based representations in conjunction with reinforcement learning techniques to acquire grasping policies automatically with no prior knowledge other than that captured in the native control basis. The platform for this experimental effort included a three-fingered, nine degree of freedom (DOF) Stanford/JPL hand⁵⁷ equipped with Brock fingertip load cells for tactile sensing. Our hand is mounted on a five DOF GE P50 robot arm (Fig. 8). The PICK-AND-PLACE schema (Fig. 4) was used in a sequence of real and simulated grasping problems on randomly chosen cylinders, cubes, and triangular prisms whose identities are unknown to the robot. The objects were wooden blocks with heights of 10 cm (cube, triangular prism) and 20 cm (cylinder).

Thirty-five grasp trials for each object type and control parameterization were used to build models of interaction dynamics yielding 420 grasp trials from which 61 unique models were derived. To learn grasping strategies, a controller was selected randomly from the control basis and used to generate observations $\mathbf{o} = (\epsilon, \dot{\epsilon})$. Models that explain \mathbf{o} were used to form the Boolean membership pattern, \mathbf{q} . The state (ϕ_i, \mathbf{q}) was used to index into a state-action value table and the corresponding best action was selected.⁹

For purposes of evaluation, learning trials consisted of fifteen independent runs over 1,600 simulated grasp trials and were compared to the performance of an oracle that knew object identity (but not orientation) and selected the best controller from the outset of the grasp. Performance was measured as $(1 - \mu_0)$, where μ_0 is the minimum coefficient of friction necessary for wrench closure, and by counting the number of tactile probes necessary en route to the final grasp.

The learning algorithm acquired a single supervisory policy for unknown objects that identifies haptic category in order to configure grasp controllers appropriately. In so doing, it improved the average performance on all objects (except the cylinders, where all grasp controllers yield the same quality). Most grasps executed by the learned grasping policy result in high quality grasp configurations, with scores in the range [0.8, 1.0] for all objects. The most dramatic value of the learned policy is that significantly fewer tactile probes are required to achieve this μ_0 performance. The oracle required approximately 20 tactile probes for cylinders, and 30 for cubes and triangular prisms. The median number of probes for the learned grasping policy for all objects was 4. The interaction-based state representation and the feedback time series dramatically improve performance in this regard by the efficient recovery and use of information.

4.5. *Visual affordance and haptic categories*

Given haptic categories and expected performance, we illustrated an approach to learning visual features that inform grasp control in a simulated “look-reach-grasp” system.¹⁰ In this experiment, the control system is presented with the same cylinders, cubes, and triangular prisms of various sizes and orientation that were used to learn haptic categories (Sec. 4.4). We aimed to use the mechanisms designed to recognize generic categories in sensor signals described in Sec. 2.1.1 to learn visual affordances for grasping tasks. An image of the object was stored and the grasp controllers were engaged. Haptic categories that discriminate high quality grasps from low quality grasps were used by the visual feature learning system to acquire feature constellations that predicted “reach” parameters (wrist configuration) as well as the ultimate quality of grasp. The learned mapping from visual features to parameters for a motor act constitutes an affordance for grasping.^{15,16} In a sense, the visual affordances acquired subsume information initially hidden in a prolonged sequence of tactile probes.

Figure 9 illustrates one set of features that was learned for triangular prisms. In this case, two distinct feature constellations were discovered, consisting of two and three oriented texels, respectively. Once an affordance is reliable and robust, it recommends an initial hand and wrist configuration with which to initialize the grasp controller. We anticipate that the acquisition of visual affordance can significantly reduce the number of expensive tactile probes en route to a grasp and simultaneously improve the ultimate grasp quality.

The discovery of such visual affordances extends the capacity of the whole system by creating new, task-specific sensors that optimize behavior in situations that need not be anticipated during the design of the system.

4.6. *Whole body grasping*

Although the robot grasping literature typically assumes that fingertips alone will be used to grasp objects, non-fingertip contacts are also possible. Potential contact points may exist on the proximal phalanges of fingers, the palm, arms, torso, legs,

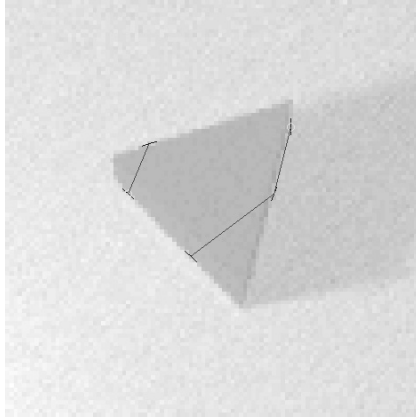


Fig. 9. A constellation of texels in a geometric relationship were discovered for the class of objects labeled triangular prisms. This affordance for grasping captures the overall shape, recommends a three-fingered grasp, and identifies a fruitful hand position and wrist orientation relative to the spatial feature with which to initialize the grasp controller.

other robots, or even inanimate objects. We use the term “whole body grasp” to refer to grasps that use contacts on arbitrary body surfaces.⁵¹

Unfortunately, whole body grasping is more complex than fingertip grasping because the quality of a whole body grasp is highly dependent on manipulator morphology. In general, contacts separated from one another through few degrees of freedom are said to be kinematically *connected*, while contacts separated from each other by many degrees of freedom are more *mobile*. Whole body contacts can be on less dexterous robot surfaces such as the palm of the hand, the surface of the arm, or the chest, and these contacts are typically more kinematically *connected* than fingertip contacts. Therefore, the kinematics of potential contact surfaces must be considered during grasp synthesis. For example, Figs. 10(a) and (b) illustrate two possible grasps of a cylinder. Figure 10(a) shows a humanoid hand grasping a cylinder by opposing the thumb with the fingers. Figure 10(b) shows the hand grasping a cylinder by attempting to oppose the thumb and index finger with the ring finger. While it is possible that both contact configurations form a wrench closure, the grasp illustrated in Fig. 10(a) is more stable and better able to apply large grasp forces than the grasp shown in Fig. 10(b). Therefore, when whole body contacts are used in grasp synthesis, it is important to consider the differing ability of contacts to apply forces in grasp-appropriate directions and how this depends on kinematic configuration.

4.6.1. *Virtual fingers*

One way manipulator kinematics can inform the grasp search process is through the use of “virtual fingers.” A virtual finger is a set of fingers or other hand/arm surfaces that function as a unit and provide a single oppositional force.⁴⁰ For example,

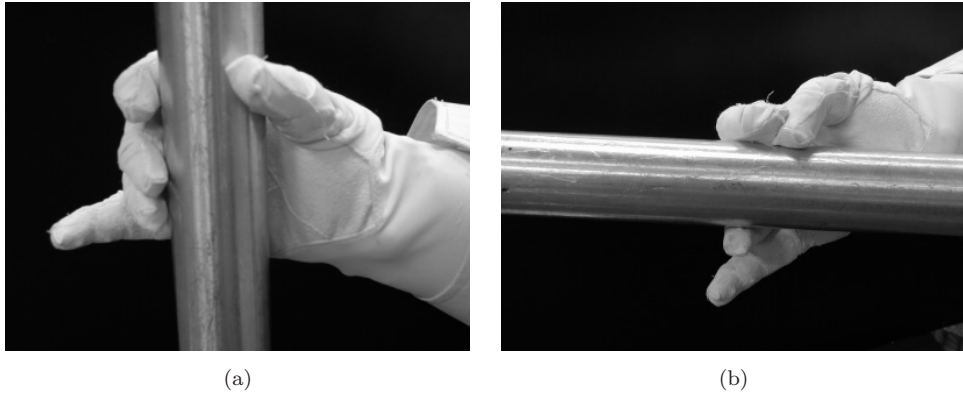


Fig. 10. (a) Robonaut hand illustrating a grasp that takes advantage of hand kinematics by opposing thumb and fingers. (b) Robonaut hand illustrating a grasp where the index finger and thumb oppose the ring finger.

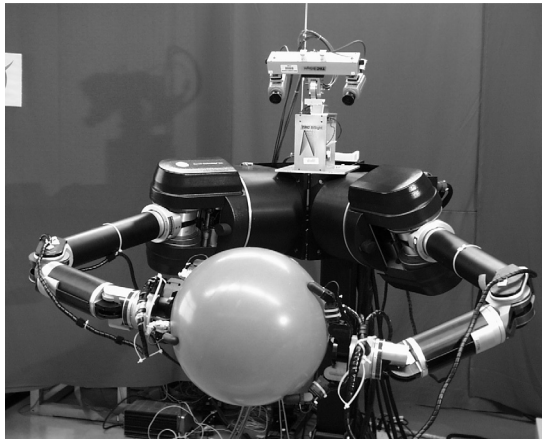


Fig. 11. Dexter, the UMass humanoid, incorporates two 7 DOF Whole Arm Manipulators (WAMs), two 3-fingered Barrett hands with six axis fingertip load cells, a 4 DOF BiSight stereo head, and a 4 microphone audio array.

humans often use their four fingers as a single virtual finger in opposition to the thumb. Figure 10(a) illustrates this on a humanoid robot hand.

In general, a virtual finger combines information from two or more constituent contacts to yield a new virtual finger with contact and sensing properties at least as good as the original contacts. For example, Fig. 11 shows Dexter, the UMass humanoid, grasping a large ball using two virtual fingers — each composed of three primitive contacts on each hand. Although the constituent contacts are hard contact types, each of the resultant virtual fingers effectively has many of the properties of a soft contact type and is able to sense local object curvature.

In the current work, we do not take advantage of the improved effective contact type, but approximate the virtual finger as a frictionless contact with a position and normal equal to the average of the constituent contacts:

$$\vec{x}_{VF} = \sum_{i \in VF} \vec{x}_i / |VF|, \quad (4)$$

$$\vec{n}_{VF} = \sum_{i \in VF} \vec{n}_i / \left| \sum_{i \in VF} \vec{n}_i \right|, \quad (5)$$

where \vec{x}_i and \vec{n}_i are the location and normal, respectively, for the i th contact in the virtual finger. This single averaged contact is used by the grasp controller as if it were a single contact.

By parameterizing a grasp controller with virtual fingers instead of primitive contacts, the grasping system can exercise finer-grained control over what type of grasp the controller generates. This is useful in the context of whole body grasping because the grasp controller can be forced to move tightly coupled contacts as a unit rather than individually.

4.6.2. Synthesizing whole body grasps through controller funneling

One of the effects of kinematic constraints on contact mobility is an increased number of local minima in the grasp potential function. While virtual fingers impose some structure over grasp synthesis, it is still possible for kinematic constraints to cause the grasp controller to settle into non-optimal equilibria.

Controller funneling can help in this situation because the number of local minima in the grasp function greatly depends on exactly which whole body contacts are used. In controller funneling, two or more controllers are executed sequentially with the first controller conditioning (or preparing) the system such that the second controller executes successfully.⁷ In the context of whole body grasping, grasp controllers parameterized by proximal contacts are executed before controllers parameterized by distal contacts. This procedure assumes that the proximal-contact grasp controller has fewer local minima than the distal controller and that after the proximal controller executes, the distal controller is more likely to succeed than would otherwise be the case.

As an intuitive example of a situation when this assumption appears to be true, consider a humanoid robot reaching toward a big ball as illustrated in Fig. 11. Assume that there are three relevant opposition spaces: opposition between palms, opposition between thumb and fingers on each hand, and opposition among the fingers on a single hand. In this case, the grasp processes might work as follows: the robot first grasps the object with two arms, each arm terminated with a single virtual finger. Once the two arms are placed on the object in a satisfactory way, the thumb and fingers are recognized as two separate virtual fingers. These contacts are displaced on the object surface without significantly moving the arms until they

too are optimally placed. Lastly, the fingers may be optimized with respect to each other without significantly displacing the hand.

We tested virtual fingers and proximal-to-distal controller funneling using a simulation of the NASA JSC Robonaut.¹³ Robonaut is a dexterous humanoid robot designed and built at NASA’s Johnson Space Center. In our experiments, the Robonaut hand executed enveloping (palmar) grasps on cylinders presented in random orientations. We first evaluated the performance of a grasp controller parameterized with a distal opposition space for 40 trials. Second, we evaluated performance when a grasp controller parameterized with proximal contacts executed

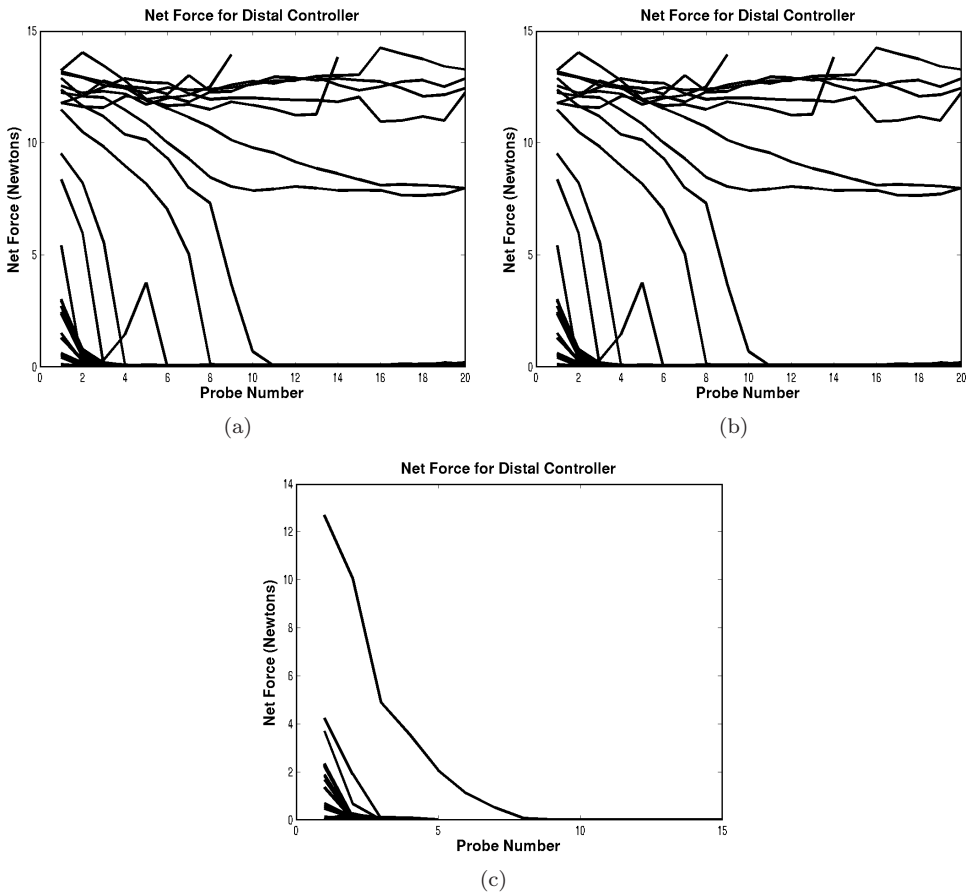


Fig. 12. The results of an experiment using virtual fingers and proximal-to-distal controller funneling. A simulation of the Robonaut hand was used to execute enveloping grasps on cylinders presented in random orientations. Each trial starts with the hand in a random orientation above a 4 cm radius cylinder. (a) illustrates the net force behavior of the grasp controller parameterized with distal contacts. (b) shows net force for the grasp controller parameterized with proximal contacts. (c) shows net force for the distal grasp controller that executed after executing the proximal controller first.

before a controller parameterized with distal contacts. The results are illustrated in Fig. 12. Figure 12(a) shows that when the distal controller executes without first having executed the proximal controller, the controller may fail to converge (possibly by entering a configuration similar to that illustrated in Fig. 10(b)). Figure 12(b) illustrates that the grasp controller parameterized with proximal contacts always converges for this object. Finally, Fig. 12(c) illustrates that when the proximal controller executes first, the distal controller always succeeds. For more information on whole body grasping, see Ref. 51.

4.7. Learning by demonstration

Parents have a great deal of input into the development of motor skills by their children. It is our goal to develop robot interfaces that enable human “parents” to interact with robots in a similar manner. For example, we have begun to develop learning methods that attempt to *explain* inputs from a human operator (the parent) by searching for elements of the control basis that produce similar actions in the current context.¹⁷ Subsequent exploration can then be focused exclusively on such plausible controllers to generate an internal representation of the teleoperated task expressed in terms of value functions native to the device. Methods like this exist for extracting physical models capable of reproducing teleoperator inputs in a variety of control situations. Bentivegna and Atkeson² showed that one could extract both primitive controllers and the conditions under which a primitive controller could be activated from a teacher demonstration of a skill. Miyata *et al.*⁴¹ showed how motion capture data of human subjects can be projected onto several scalar objective functions and then used in conjunction with a motion controller to generate approximations of the training data.

Our framework for learning by demonstration compares observed teleoperator inputs with the control signals generated by a set of hypothesis controllers. When presented with a new configuration of objects, the recognition system first visually extracts coarse object models that capture approximate pose, shape, size, and color distribution of the objects. The set of hypothesis controllers captures the set of actions *afforded* by these objects^{21,22}: one controller is parameterized for each distinct way in which each object may be grasped. The controllers themselves may be used to grasp the objects, but in this context they also serve as virtual sensors that quantify the degree of match between the actions commanded by the teleoperator and those of the hypothesis controllers. We say that a hypothesis controller for grasping or releasing an object explains a sequence of teleoperator-commanded movements when the following conditions hold: (i) each step in the sequence reduces controller error, (ii) the error measured by the hypothesis controller at the end of the sequence is small, and (iii) the end of the sequence is punctuated by a tactile event (as an object is grasped or released by the hand).

Figure 13 shows the initial and final configurations of objects used in a preliminary experiment to reliably extract the intentions of a teleoperator during a

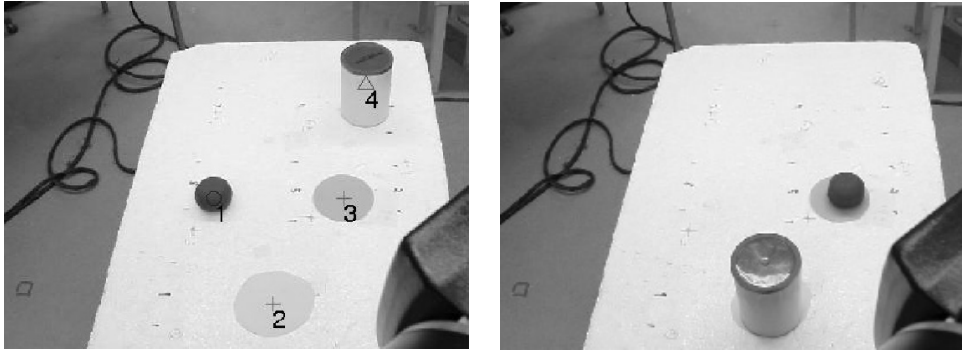


Fig. 13. Left: configuration of objects at the beginning of the demonstrated task, taken from Dexter’s right camera. The picture has been annotated by the vision system; the numbers correspond to object identifiers. The vision system identifies a number of affordances for each object. The ball has two affordances: one for a top grasp, and an (unavailable) affordance for placing other objects at that position on the table. The two targets each have a single affordance for placing objects on top of them. The can has three affordances: a top grasp, a side grasp, and an (unavailable) affordance for placing objects at its position on the table. Right: the configuration of objects after the task has been completed. Note that here, the affordances for the previous positions of the can and the ball are now activated.

sequence of pick-and-place tasks. Figure 14 shows an excerpt from the experiment that illustrates the system inferring the intentions to pick up the can and place it on the bottom target. Note that the time series represents a significant compression of the observed teleoperator movements that were sampled at 50 Hz. Because each hypothesis controller is also paired with a description of the original object, this sequence of high-level actions can be described as a combination of the object properties and the way in which the object was grasped. Furthermore, when presented with a novel situation the same sequence of actions can be executed automatically. This execution is performed using the same set of controllers used in the recognition process, relativized to the new locations of the participating objects. Typically, the automated execution demonstrates a significant improvement in speed and accuracy over that of the human teleoperator.

4.8. Learning to exploit dynamics

Some tasks, like swinging a golf club, can benefit from fully integrated policies that exploit fine-grained dynamics for the benefit of the expert player. We have constructed a learning method to do this by using supervision in the form of control policies derived from the control basis. This kind of supervisor incorporates structure derived from discrete-event model checkers, it can be explained in terms of sequences of objectives, and it can focus on specific perceptual and motor resources.

To learn expert-level performance, our learning framework⁵⁶ uses a combination of supervised learning with an actor-critic architecture for Reinforcement

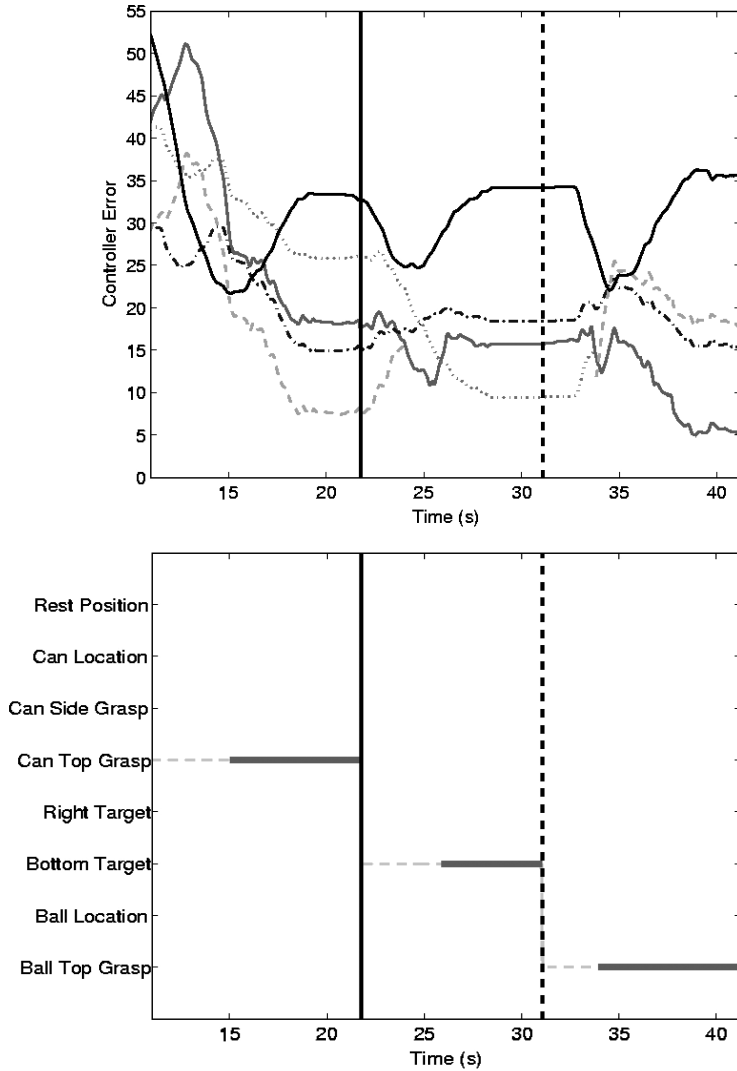


Fig. 14. Top: the graph shows the controller errors for five affordances over a portion of the demonstration. During this period of the experiment, the teleoperator grasped the can and released it on the bottom target in Fig. 13. Next, the arm was moved to grasp the ball. The affordances shown in this graph correspond to a rest position of the arm above the table (solid black line on top for a majority of the trial), the top grasp of the ball (solid grey), placing an object on the bottom target (dotted), placing an object on the right target (dot-dash), and a top grasp of the can (dashed). The solid vertical line shows the time at which the teleoperator performs a top grasp of the can. The corresponding affordance becomes unavailable shortly thereafter; this is shown by the line disappearing. The release of the can is shown by the dashed vertical line. After releasing the can on the bottom target, its affordance becomes unavailable, and the can affordance becomes active again. Bottom: the graph shows the estimated intended action extracted by the system (solid) and the true intended action (dashed) of the teleoperator, over the same period of the experiment. Grasp and release by the teleoperator is denoted as in the top graph.

Learning (RL). One of the key advantages of our learning framework is that a supervisor can be constructed easily from the control basis. Furthermore, with a stable controller as the supervisor, the RL system quickly learns about favorable control actions but without the risks normally associated with exploratory learning.

Almost all RL methods that incorporate supervisory input do so by modifying the value function exclusively. In these approaches, the value function encodes an implicit representation of the required controller. Actor-critic methods, on the other hand, modify separate data structures for the controller (the “actor”) and the value function (the “critic”). This obviates the need for a costly search for the best-valued action at each control cycle. Moreover, separate data structures allow the actor to be modified directly by standard supervised learning methods. Essentially, the actor learns to mimic the behavior of its supervisor but adjusts this behavior using its own exploratory actions.

Figure 15 shows the results of this framework applied to a point-to-point reaching task for one of Dexter’s arms. The goal was to lift a tool to a ready position using minimal effort. After about 25 trials, the system learns how to clone the supervisor’s behavior, and by 60 trials, of learning the supervised actor-critic architecture shows statistically significant improvement ($p < 0.01$) over the supervisor alone. After 120 trials, the overall effect of learning is approximately 20% reduced effort despite an increased average movement time from 4.16 s to 4.34 s (statistically significant with $p < 0.05$).

4.9. Learning mixed-initiative controllers

The supervised actor-critic framework can also accept inputs from a human supervisor using a teleoperator interface. As an example, Fig. 16 shows a sequence of

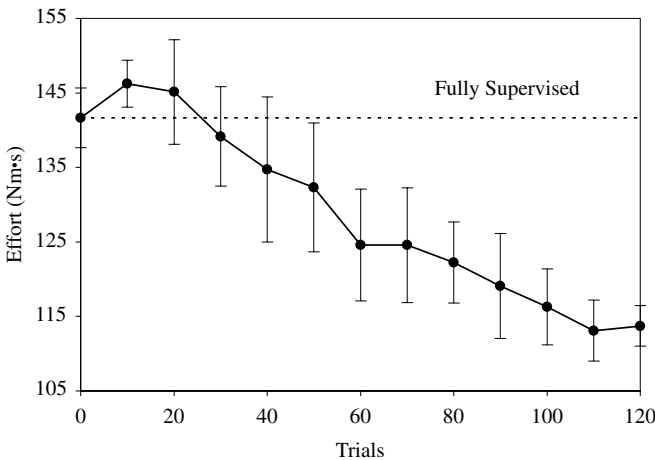


Fig. 15. Effects of learning for the point-to-point reaching task averaged over 5 runs of 120 trials each.

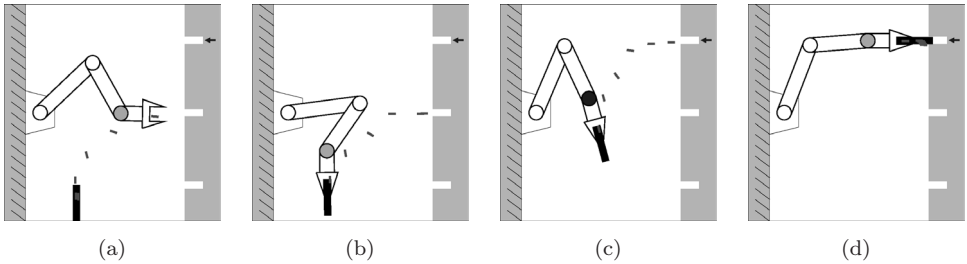


Fig. 16. Screen shots from a simulated peg insertion task. In (a), a successful re-grasp of the peg is predicted by the actor; (b) with no knowledge of the target, the actor makes initial progress toward the middle slot; (c) a small correction by the human operator places the robot on track for the upper target, after which (d) the learned controller completes the sub-task.

frames during a simulated peg insertion task. With no initial control knowledge about the task, the actor is completely dependent upon teleoperation input from the human supervisor (via a mouse). After several trials, however, the actor has gathered sufficient information with which to propose useful actions. Short bars in the figure depict the effects of these actions, as projected forward in time by prediction through a kinematic model. In the leftmost panel, for instance, the bars indicate to the operator that the learning system will translate and rotate the end effector for successful re-grasp of the peg. In this scenario, the actor has no knowledge of the target slot and so immediately after re-grasp [panel (b)] the learning system “proposes” insertion into the recently visited middle slot. A momentary command from the operator is sufficient to push the system into the basin of attraction for the upper target [panel (c)]. Finally, the rightmost panel shows successful completion of the sub-task a short time later under full control by the actor. Although human and machine traded full control of the robot in this example, the underlying learning architecture supports the entire spectrum from full autonomy to full control by a supervisor.

5. Hybrid Planner-Reactor Frameworks

This section discusses work underway to integrate planners and multi-objective control policies so that robots can reason about behavior at the frontier of its task knowledge. Policies for reaching, grasping, and manipulation implicitly model transition dynamics and, therefore, can be used to predict possible future states. These outcomes can be engaged selectively by a human operator, they can be explored stochastically by a learning algorithm, or they can be evaluated by a planner.

In contrast to algorithms like reinforcement learning, which rely on stochastic search to evaluate the value of actions, planners can make significant use of predictions of future states. Global planning techniques permit powerful forms of prediction. However, the computational complexity of exhaustive predictions of

future state quickly overwhelm the planner and prevent the resulting behavior from responding to changes in a timely fashion.

In this section we introduce methods for real-time motion generation and collision avoidance for robots with many degrees of freedom, such as humanoid robots. By combining planned motions with behavioral hierarchies from the control basis, task-related behavior can be achieved in a way that preserves run-time flexibility, permits reactive systems to consider objectives at run-time that have not been explicitly planned, and preserves the notion of redundancy and null space in planned actions. In addition, the closed-loop behavior from which actions are composed guarantees performance in the presence of disturbances related to the task.

5.1. *Real-time path planning*

The computational requirements of path planning in high-dimensional configuration spaces challenges the integration of planning methods and reactive behavior. These computational requirements are a consequence of the exponential size of the configuration space that must be searched for a collision-free motion. For any particular motion planning problem, only a small fraction of the configuration space is relevant to the solution. We present a motion planning approach that restricts search to a relevant subset of configuration space, sufficiently reducing the computational complexity of motion planning to allow closed-loop re-planning in response to changing environments.

Decomposition-based motion planning³ is motivated by the insight that large portions of configuration space represent configurations of the robot that are not physically attainable are obviously not part of a reasonable solution (the robot wrapping around a chair, for example), or are part of a large set of slight variations of relevant configurations. Decomposition-based motion planning techniques employ information obtained in the workspace to differentiate between relevant and irrelevant regions of configuration space. The overall motion planning problem is divided into two subproblems. First, a low-dimensional problem is solved in the workspace. The solution to this problem captures relevant free space connectivity in the workspace and can be computed efficiently. Subsequently, this information is employed to determine an answer to the original motion planning problem in configuration space. This can be accomplished in various ways. We have successfully explored the application of reactive methods based on multi-objective controllers³ and efficient sampling-based motion planning techniques.⁶⁶

Preliminary implementations of decomposition-based motion planning methods have been used to generate the motion of an eleven degree-of-freedom mechanism operating in a multi-room indoor environment with moving obstacles at rates required for closed-loop behavior.³ The significant increase in performance is the result of a conscious trade-off between efficiency and completeness. As a consequence

of this trade-off, the proposed approach can fail, even if a path exists. In these cases, a more computationally expensive planning method can then be employed to solve the motion planning problem.

5.2. Real-time path modification

The elastic strip framework permits the real-time modification of a previously planned path in a high-dimensional configuration space.⁵ In this approach, multi-objective, closed-loop controllers are employed to incrementally modify an existing motion plan without violating specified motion constraints. This modification is subject to various constraints associated with physical limitations of actuators, postural stability, pick-and-place constraints, power consumption, or to make the motion appear human-like. The elastic strip framework can be viewed as an augmentation to conventional path planners, providing them with the capability to execute sophisticated, task-oriented, multi-objective motion for mechanisms with many degrees of freedom in dynamic environments. Substantial changes in the environment, however, can invalidate the motion and prevent the incremental path modification procedure from maintaining constraints. In such a case, a global planner must be invoked to determine a new path that satisfies all required constraints.

Starting with the path generated by a motion planner to bring the end-effector into a position determined by the grasping controller, for example, the elastic strip framework augments the path with an approximation of the free workspace surrounding it. This workspace volume can be viewed as a tunnel of free space through which the robot is moving. This tunnel implicitly represents homotopically equivalent alternatives to the current path.⁴ These paths all respect the planners motion constraints and thus can be viewed as the null space of the original plan. The selection of incrementally modified paths is performed by multi-objective controllers that encode various constraints. Since the entire volume of the tunnel satisfies the planners criteria, any incremental modification of the original path entirely contained by the tunnel is guaranteed to be free of collision. In addition to the integration of arbitrary constraints, the elastic strip framework allows for the automatic suspension of desirable, but not critical constraints when they conflict with constraints whose violation would lead to catastrophic failure.⁶

For example, Fig. 17 shows five consecutive snapshots of a humanoid with 34 degrees of freedom. The initial trajectory specifies motion without posture changes. This trajectory is modified in real-time during execution of the motion while observing three different constraints: the obstacle, which is being lowered during the motion, is avoided, the overall balance of the mechanism is maintained, and among all the possible motions conforming with the previous requirements, those that appear human-like are preferred. Note that the actuation of bipedal walking is being ignored in this experiment.

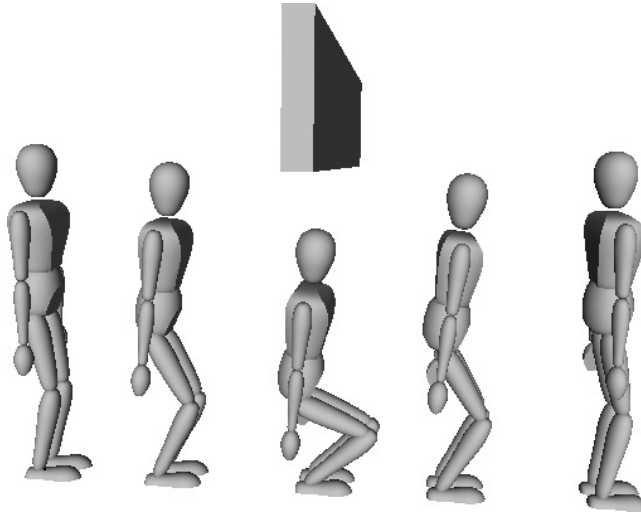


Fig. 17. Five consecutive snap-shots illustrating incremental modification of humanoid motion considering obstacle avoidance, balance constraints, and preferred posture. The obstacle, being lowered during the motion, is avoided while the overall balance of the humanoid is maintained using motions that appear human-like.

6. Conclusion

Humanoid robots introduce significant challenges associated with open environments, including variability, redundancy, and missing state. Moreover, humanoids are intended to operate in human environments and to collaborate efficiently with human partners. The most striking technological challenge is related to robot control frameworks for interacting physically and forcefully with the environment. This is an extremely rich domain that has seen far less progress than technologies for collision-free motion and environmental mapping for mobile platforms. This is because representations of control knowledge underlying manual skill must be far more sophisticated than knowledge of freespace. Therefore, this paper focuses on a decade of research at the Laboratory for Perceptual Robotics toward building robot grasping and manipulation systems.

We present a representation couched in control dynamics that support lifelong learning of intricate models which describe generic interactions between a robot and the world. The control basis is free of spurious minima, is uncommitted to pre-conceived solutions to pre-defined tasks, and provides a combinatoric basis for exploration and learning. Importantly, the control basis provides a natural, *factored* modeling framework that serves to distinguish the *objectives* from the *implementation* of behavior. This is an important point, because it allows behavior to be re-used in similar (but not identical) situations. We advocate an *interactionist* state representation that matches observations gathered *in situ* to prototypes in order to create discrete interaction categories. This allows one to use *model checking*

techniques employing discrete event dynamic simulations to structure exploration in learning systems and well understood techniques for building verifiable *multi-objective* control systems.

A variety of demonstrations are presented to illustrate the expression of skills for grasping and manipulation using this framework, including: grasping, integrated pick-and-place tasks, learning haptic and visual affordances for grasping, generalizing to whole body grasping tasks, learning by recognizing intent, learning to exploit fine-grained dynamics, and learning mixed-initiative control.

Finally, we explore the relationship of planning techniques with reactive policies that capture critical control knowledge. In addition to using controlled behavior directly as forward models in planning architectures, we introduce a high performance, decomposition-based planning methodology. Finally, techniques that capture analogs of linear null spaces in motion plans using the elastic strip technique are presented to form a hybrid planner-reactor.

Acknowledgments

This article has benefited significantly from conversations with Justus Piater, Manfred Huber, and Jefferson Coelho over several years. The ideas contained herein were influenced by this dialog, although the opinions we express are not necessarily endorsed by them. We also thank Steve Hart and Deepak Karupiah, current members of the LPR who are helping to carry on this discussion. We would also like to thank Robert Ambrose, William Bluethmann and Myron Diftler at NASA Johnson Space Center for providing access to Robonaut. The work described herein was supported in part by the NSF (CDA 9703217), DARPA-MARS (DABT63-99-1-0004), NASA (NAG9-1445#1), and the University of Massachusetts. Any opinions, findings, conclusions, or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

References

1. M. Arbib, T. Iberall and D. Lyons, Coordinated control programs for movements of the hand, Technical Report COINS 83-25, University of Massachusetts, 1983.
2. D. Bentevegna and C. Atkeson, Learning how to behave from observing others, in *SAB'02-Workshop on Motor Control in Humans and Robots: On the Interplay of Real Brains and Artificial Devices*, Edinburgh, UK, 2002.
3. O. Brock and L. E. Kavraki, Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces, in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Seoul, Korea, 2001, pp. 1469–1474.
4. O. Brock and O. Khatib, Real-time replanning in high-dimensional configuration spaces using sets of homotopic paths, in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Vol. 1, San Francisco, USA, 2000, pp. 550–555.
5. O. Brock and O. Khatib, Elastic strips: A framework for motion generation in human environments, *Int. J. Robot. Res.* **21**(12), 1031–1052 (2002).

6. O. Brock, O. Khatib and S. Viji, Task-consistent obstacle avoidance for mobile manipulation, in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Washington DC, USA, 2002, pp. 388–393.
7. R. Burridge, A. Rizzi and D. Koditschek, Sequential composition of dynamically dexterous robot behaviors, *Int. J. Robot. Res.* **18**(6), 534–555 (1999).
8. I. Chen and J. W. Burdick, Finding antipodal point grasps on irregularly shaped objects, in *Proc. IEEE Int. Conf. Robotics and Automation*, Vol. 3, Nice, France, May 1992, pp. 2278–2283.
9. J. Coelho, Multifingered grasping: Haptic reflexes and control context, PhD thesis, University of Massachusetts, Amherst, MA, September 2001.
10. J. Coelho, J. Piater and R. Grupen, Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot, in *Proc. 1st IEEE-RAS Int. Conf. Humanoid Robots*, IEEE-RAS, September 2000.
11. J. A. Coelho and R. A. Grupen, A control basis for learning multifingered grasps, *J. Robot. Syst.* **14**(7), 545–557 (1997).
12. C. Connolly and R. Grupen, On the applications of harmonic functions to robotics, *J. Robot. Syst.* **10**(7), 931–946 (1993).
13. M. Diftler, C. Culbert, R. Ambrose, R. Platt and W. Bluethmann, Evolution of the nasa/darpa robonaut control system, in *IEEE Int Conf. Robotics Automation*, 2003.
14. G. L. Drescher, *Made-up Minds — A Constructivist Approach to Artificial Intelligence* (MIT Press, 1991).
15. A. Fagg, A computational model of the cortical mechanisms involved in primate grasping, Ph.D. thesis, Department of Computer Science, University of Southern California, 1996.
16. A. Fagg and M. Arbib, Modeling parietal-premotor interactions in primate control of grasping, *Neural Networks* **11**(7–8), 1277–1303 (1998).
17. A. H. Fagg, M. T. Rosenstein, R. Platt and R. A. Grupen, Extracting user intent in mixed initiative teleoperator control, in *Proc. American Institute of Aeronautics and Astronautics Intelligent Systems Technical Conference*, 2004.
18. B. Faverjon and J. Ponce, On computing two-finger force-closure grasps of curved 2D objects, in *Proc. IEEE Conf. Robotics and Automation*, Vol. 1, April 1991, pp. 424–429.
19. C. Ferrari and J. Canny, Planning optimal grasps, in *Proc. 1992 Int. Conf. Robotics and Automation*, Vol. 3, Nice, France, May 1992, pp. 2290–2295.
20. D. Gabor, Theory of communication, *J. Inst. Elect. Eng.* **93**, 429–457 (1946).
21. J. J. Gibson, The optical expansion-pattern in aerial location, *Am. J. Psychol.* **68**, 480–484 (1955).
22. J. J. Gibson, *The Senses Considered as Perceptual Systems* (Allen and Unwin, 1966).
23. R. Grupen and K. Souccar, Manipulability-based spatial isotropy: A kinematic reflex, in *Workshop on Mechatronical Computer Systems for Perception and Action*, Halmstad, Sweden, June 1–3, 1993.
24. R. A. Grupen, J. Coelho and K. Souccar, On-line grasp estimator: A partitioned state space approach, Technical Report 92-75, CS Department, University of Massachusetts, October 1992.
25. R. A. Grupen and T. C. Henderson, A control paradigm for general purpose manipulation systems, in *Proc. IEEE Conf. Robotics and Automation*, Philadelphia, PA, April 1988, pp. 710–715.
26. G. Guo, W. A. Gruver and K. Jin, Grasp planning for multifingered robot hands, in *Proc. IEEE Int. Conf. Robotics and Automation*, Vol. 3, Nice, France, May 1992, pp. 2284–2289.

27. M. Huber, A hybrid architecture for adaptive robot control, Ph.D. thesis, University of Massachusetts, Amherst, MA, May 2000.
28. M. Huber and R. Grupen, A feedback control structure for on-line learning tasks, *J. Robots Autom. Syst.* **22**(3-4), 303-315 (1997).
29. M. Huber, W. MacDonald and R. Grupen, A control basis for multilegged walking, in *Proc. IEEE Conf. Robotics and Automation*, Minneapolis, MN, April 1996.
30. D. P. Huttenlocher and S. Ullman, Recognizing solid objects by alignment with an image, *Int. J. Comp. Vision* **5**(2), 195-212 (1990).
31. A. Ijspeert, J. Nakanishi and S. Schaal, Learning attractor landscapes for learning motor primitives, in *Advances in Neural Information Processing Systems*, eds. S. Becker, S. Thrun and K. Obermayer, Vol. 15 (MIT Press, 2003), pp. 1547-1554.
32. J. W. Jameson and L. J. Leifer, Automatic grasping: An optimization approach, *IEEE Trans. Syst. Man Cybernet.* **SMC-17**(5), 806-814 (1987).
33. L. P. Kaelbling, M. L. Littman and A. R. Cassandra, Planning and acting in partially observable stochastic domains, *Artif. Int.* **101**(1-2), 99-134 (1998).
34. O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, *Int. J. Robot. Res.* **5**(1), 90-98 (1986).
35. D. E. Koditschek and E. Rimon, Robot navigation functions on manifolds with boundary, *Adv. Appl. Math.* **11**(4), 412-442 (1990).
36. J. Koenderink and A. van Doorn, Representation of local geometry in the visual system, *Biol. Cybern.* **55**, 367-375 (1987).
37. J. Laird, P. Rosenbloom and A. Newell, Chunking in Soar: The anatomy of a general learning mechanism, *J. Mach. Learn.* **1**, 11-46 (1986).
38. Z. Li and S. Sastry, Task-oriented optimal grasping by multifingered robot hands, *IEEE Trans. Syst. Man Cybernet.* **11**(10), 681-689 (1988).
39. T. Lindeberg, *Scale-Space Theory in Computer Vision* (Kluwer Academic Publishers, 1994).
40. C. MacKenzie and T. Iberall, *The Grasping Hand* (North-Holland, 1994).
41. N. Miyata, K. Oguri, J. Ota and T. Arai, Human lift-up motion generation based on identification of time-variant performance index, in *Proc. 2002 Int. Conf. Intelligent Robots and Systems (IROS)*, IEEE/RSJ, 2002, pp. 2503-2508.
42. H. Murase and S. K. Nayar, Visual learning and recognition of 3D objects from appearance, *Int. J. Comp. Vision* **14**, 5-24 (1995).
43. K. Murphy, Dynamic Bayesian networks: Representation, inference, and learning, Ph.D. thesis, University of California Berkeley, Berkeley, CA, July 2002.
44. Y. Nakamura, *Advanced Robotics: Redundancy and Optimization* (Addison-Wesley, Reading, MA, 1991).
45. A. Newell and H. Simon, *Human Problem Solving* (Prentice Hall, Englewood Cliffs, NJ, 1972).
46. V. D. Nguyen, Constructing stable grasps, *Int. J. Robot. Res.* **8**(1), 26-37 (1989).
47. C. M. Özveren and A. S. Willsky, Observability of discrete event dynamic systems, *IEEE Trans. Automat. Contr.* **35**(7), 797-806 (1990).
48. J. Piater, Visual feature learning, Ph.D. thesis, University of Massachusetts, Amherst, MA, May 2001.
49. J. H. Piater and R. A. Grupen, Toward learning visual discrimination strategies, in *Proc. Computer Society Conf. Computer Vision and Pattern Recognition*, Fort Collins, CO, June 1999, IEEE, pp. 410-415.
50. J. H. Piater and R. A. Grupen, Feature learning for recognition with Bayesian networks, in *IEEE Proc. 15th Int. Conf. Pattern Recognition*, September 2000.

51. R. Platt, A. Fagg and R. Grupen, Extending fingertip grasping to whole body grasping, in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 2003.
52. R. Platt, A. H. Fagg and R. Grupen, Nullspace composition of control laws for grasping, in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, Lausanne, Switzerland, 2002.
53. D. Precup and R. Sutton, Multi-time models for temporally abstract planning, in *Proc. Advances in Neural Information Processing Systems*, Vol. 10 (MIT Press, 1997).
54. P. J. G. Ramadge and W. M. Wonham, The control of discrete event systems, in *Proc. IEEE* **77**(1), 81–97 (1 January 1989).
55. R. Rao and D. Ballard, An active vision architecture based on iconic representations, *Artif. Int.* **78**, 461–505 (1995).
56. M. Rosenstein and A. Barto. Supervised actor-critic reinforcement learning, in *Learning and Approximate Dynamic Programming: Scaling Up to the Real World*, eds. J. Si, A. Barto, W. Powell and D. Wunsch (John Wiley & Sons, Inc., New York, 2004), pp. 359–380.
57. J. K. Salisbury, Kinematic and force analysis of articulated hands, Ph.D. thesis, Stanford University, May 1982.
58. S. Schaal and D. Sternad, Programmable pattern generators, in *Proc. Int. Conf. Computational Intelligence in Neuroscience*, North Carolina, USA, 24–28 October 1998, pp. 48–51.
59. B. Schiele and J. L. Crowley, Object recognition using multidimensional receptive field histograms, in *Proc. 4th European Conf. Computer Vision*, Cambridge, UK, 1996.
60. M. Sobh, J. C. Owen, K. P. Valvanis and D. Gracani, A subject-indexed bibliography of discrete event dynamic systems, *IEEE Robot. Autom. Mag.* **1**(2), 14–20 (1994).
61. G. Strang, Wavelets, *Am. Sci.* **82**, 250–255 (1992).
62. R. Sutton, D. Precup and S. Singh, Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning, *Artif. Int.* **112**, 181–211 (1999).
63. J. Sweeney, T. J. Brunette, Y. Yang and R. Grupen, Coordinated teams of reactive mobile platforms, in *IEEE Conf. Robotics and Automation*, Washington, DC, May 2002.
64. F. Takens, Detecting strange attractors in turbulence, in *Dynamical Systems and Turbulence*, eds. D. Rand and L. Young, Lecture Notes in Mathematics, Vol. 898 (Springer-Verlag, 1981), pp. 366–381.
65. A. Witkin, Scale-space filtering, in *Proc. 8th Int. Joint Conf. Artificial Intelligence*, **2**, 1019–1023 (1983).
66. Y. Yang and O. Brock, Viewing motion planning as disassembly: A decomposition-based approach for non-stationary robots, in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Barcelona, Spain, 2005.
67. T. Yoshikawa, Analysis and control of robot manipulators with redundancy, in *Robotics Res.: The Int. Symp.* 1984, pp. 735–747.
68. T. Yoshikawa. Manipulability of robotic mechanisms. *J. Robot. Res.* **4**(2), 3–9 (1985).



Oliver Brock is an Assistant Professor of Computer Science at the University of Massachusetts Amherst. Since 2002, he has been the Co-Director of the Laboratory for Perceptual Robotics. Brock holds a Diploma in Computer Science from the Technical University of Berlin and a Masters and Ph.D. degree from Stanford University. Brock previously served as Research Associate at Rice University and at Stanford University. His primary research focus has been on mobile manipulation in unstructured environments. He has developed approaches to enable the integration of robust control methods with global task constraints in the presence of uncertainty and dynamic change. Brock has authored over 35 publications on these subjects. He is co-founder of the Robotics: Science and Systems Conference and co-founder and Chief Technology Officer of an Internet startup with more than 800 employees.



Andrew H. Fagg holds a B.S. degree (1989) in Applied Math and Computer Science from Carnegie-Mellon University, and a M.S. degree (1991) and a Ph.D. degree (1996) in Computer Science from the University of Southern California. From 1996 to 2004, he worked as a postdoc and a senior research scientist in the Computer Science Department at the University of Massachusetts, where he was affiliated with the Adaptive Networks Laboratory and the Laboratory for Perceptual Robotics. He currently is an Associate Professor in the School of Computer Science at the University of Oklahoma, where he leads the Symbiotic Computing Laboratory. His research interests include robotics, machine learning, computational neuroscience, and wearable computing. He is interested in machine learning techniques for acquiring skills in sensing, reaching, and grasping. He also studies the cortical mechanisms involved in the representation and acquisition of motor skills in primates. In the wearable computing domain, he focuses on new models for human-computer interaction that are specific to the anytime/anywhere mode of the wearable computer form-factor. Andrew Fagg has served on the program committees for *Humanoids*, *ICAR*, *NIPS*, and *ICML*, and regularly reviews for the *Journal of Neurophysiology* and *Autonomous Robotics*.



Roderic Grupen received a B.A. in Physics from Franklin and Marshall College and a B.S. in Mechanical Engineering from Washington University in 1981. He received an M.S. degree in Mechanical Engineering in 1984 from Penn State and a Ph.D. in Computer Science from the University of Utah in 1988. That year, he joined the faculty at UMass and became Co-Director of the Laboratory for Perceptual Robotics (LPR) at UMass. In 1997, he was Principal Investigator on a collaborative project

with learning theorists and Developmental Psychologists sponsored by NSF that led to “Dexter,” the UMass platform for studying bi-manual dexterity. Grupen is now Director of the LPR and a Professor of Computer Science. His research concerns knowledge representations and developmental mechanisms in manipulation systems. He is the author of over 100 publications in the area and will soon publish a book entitled *The Developmental Organization of Robot Behavior*. He is a member of the IEEE and ACM, an associate editor of *Artificial Intelligence in Engineering Design and Manufacture* (AIEDAM), and *Robotics and Autonomous Systems*, and is the chair of the Haptics and Motor Processing Technical Committee for the IEEE Neural Network Society.



Robert Platt Jr. received his M.S. degree in Computer Science from the University of Massachusetts Amherst in 2003. Prior to this, he worked for several years as a software engineer in industry, and earned a B.S. degree in Electrical Engineering from Duke University in 1994. He is currently a Ph.D. candidate at UMass Amherst focusing on general-purpose solutions to multi-objective robot control problems such as grasping and manipulation. Rob’s work focuses on schematic representations of generalizable robot skills for grasping and manipulation.



Michael T. Rosenstein received his B.S. in Applied Physiology (1989) and his M.S. in Biomedical Engineering (1993), both with distinction from Boston University. In 2003, he earned his Ph.D. in Computer Science from University of Massachusetts Amherst. During his tenure as a graduate student he was a National Defense Science and Engineering Graduate fellow (1997–2000) and a NASA Graduate Student Researchers Program fellow (2001–2003). Dr. Rosenstein is currently a Senior Research Scientist in the Department of Computer Science at UMass Amherst. He is a member of AAAI, ACM, and IEEE, and he has co-authored more than 40 peer-reviewed publications in the areas of artificial intelligence, robotics, nonlinear dynamics, biomechanics, and neuromuscular control.



John D. Sweeney received a B.S. in Computer Science from Carnegie Mellon University in 2000. He received a Masters degree in Computer Science in 2005 from the University of Massachusetts Amherst, where he is currently a Ph.D. candidate in the Laboratory for Perceptual Robotics. His research interests include imitation learning in humanoid robots, intention recognition in human–robot interactions, and developmental programming methodologies for robot systems.