Toward A Systems Theory for the Composition of Dynamically Dexterous Robot Behaviors

R. R. Burridge, A. A. Rizzi and D. E. Koditschek
Artificial Intelligence Laboratory and Controls Laboratory
EECS Department, College of Engineering
University of Michigan
1101 Beal Ave, Ann Arbor MI 48109-2110
{burridge,arizzi,kod}@eecs.umich.edu

Abstract

We report on our efforts to develop robot controller composition techniques in the context of dexterous "batting" maneuvers. A robot with a flat paddle is required to strike repeatedly at a falling ball until it is brought to zero velocity at a specified position. The robot's workspace is cluttered with obstacles that disconnect the freespace formed when the ball and paddle remain in contact — the machine is forced to "let go" for a time in order to bring the ball to the desired state. The controller compositions that we create will guarantee that a ball introduced in the "safe workspace" remains there and is ultimately brought to the goal. We believe that the developing systems discipline described here may be extended to build a variety of useful dexterous machines that are similarly single-minded in their pursuit of the user's goal behavior and ability to surmount unanticipated perturbations along the way.

1 Introduction

Presumably, robots are to be programmed by their "higher level masters" using symbols relating to states of the world that the robot is supposed to bring into being. A central problem of robotics concerns the translation of such symbol sequences into actuator and sensor policies that carry out the intended goals. Assume that the symbols are defined unambiguously enough to imply or at least to be consistent with a partition on the state space of the world to be manipulated. In other words, some symbols connote a specific state (or an equivalence class of states that are all equally valid) that should be visited, while others connote states that should be avoided. However fuzzy or approximate their relation to the continuum of world states, let us suppose that no pair of symbols must be interpreted to connote cells that overlap. Under such an assumption, the robot symbol-to-signal problem amounts to the design of controllers capable of guaranteeing passage of the world's states to and from cells of a specified partition

We are pursuing an approach to autonomous robot design that seeks to achieve user goals that are encoded by such state space goal and obstacle sets. Our architecture implements event driven robot policies whose closed loop dynamics drive the coupled robot-environment state toward a goal set and away from obstacles. We strive to develop control algorithms that are sufficiently tractable as to allow correctness guarantees as well. Thus, we have focused theoretical attention on "practicable stability mechanisms" — dynamical systems for which effectively computable local tests provide global conclusions — in first and second order settings. At the same time, we have focused experimental attention on building a distributed computational environment that supports flexibly reconfigurable combinations of sensor and actuator hardware controllers, motion estimation and control algorithms, and event driven reference trajectory generators. The result has been a series of laboratory robots that exhibit indefatigable goal seeking behavior albeit in a very narrow range of tasks. Specifically, we have concentrated on tasks requiring dynamical dexterity — the ability to perform work on the environment by effecting changes in its kinetic as well as potential energy.

From the viewpoint of the symbol-to-signal problem our robot architecture provides tunable families of flows whose deployment is defined by a "control partition" — an assignment of cells to controllers — which induces on the world's state space a composite law of motion. Suppose a robot equipped with such an architecture is presented with the user's commands in the form of these symbol sequences. It becomes the duty of a robot interpreter to design a control partition that suitably refines the command partition so that all command symbol sequences can be realized by the appropriate composition of controller primitives. In this paper we consider the simplest possible command lexicon — goal cells comprised of singletons and obstacle cells of a particularly simple shape — and hand build an interpretation. In so doing, we are becoming convinced that a wide range of behaviors incorporating dynamical dexterity may be forthcoming from our present palette of "control primitives." We hope to see emerge from this work a systems theory for forming their purposeful composition in a theoretically verifiable manner.

2 A Family of Dynamically Dexterous Controllers

We are interested in encoding and achieving user goals taking the form of a desired set of world states, $\mathcal{G} \subset T\mathcal{B}$. A controller, Φ, achieves those goals if $\mathcal{G}$ is an attracting invariant subset, in which case we are interested in rendering its domain of attraction, $\mathcal{D}(\Phi)$ as large as possible.

We have developed over the last five years an approach to robot controller design that accomplishes these goals by essentially re-shaping the total energy exchanged between the robot and its environment. This section reviews some of these prior ideas as background for discussing the compositions introduced in the next section — the central concern of this paper.

2.1 Physical Models

Since we require the robot to bat at falling objects, we are concerned here with a model of the collisions and the kinematics of the robot that controls their occurrence. We shall restrict attention to the collisions of a point with a plane since the falling objects are all balls at present. We shall relate the abstract kinematics to the particular case of the "Buehgler" a three degree of freedom direct drive machine pictured in Figure 1, which we have discussed in a variety of other settings [1, 2].

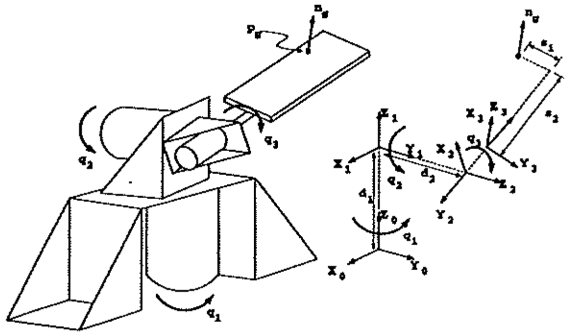

Figure 1: The Buehgler Arm

2.1.1 Flight

It is convenient to decompose the configuration space of the ball, $\mathcal{B} = I\!R^3$ into the direct sum of a two dimensional "horizontal" subspace and one dimensional "vertical" subspace, $\mathcal{B} = \mathcal{H} \oplus \mathcal{V}$, respectively. The ball's dynamics are decoupled in the two subspaces and if $b = (h, v)$ then the Newtonian flight model is given by

$$\ddot{h} = 0; \qquad \ddot{v} = -\gamma. \tag{1}$$

In order to preserve notational consistency, we will find it convenient to denote the states (positions and velocities) of a ball by $Tb := \left(b, \dot{b}\right) \in T\mathcal{B} := \mathcal{B} \times I\!R^3$ — and similarly for the horizontal and vertical components, as well as the robot generalized coordinates, $q, s, \tilde{q}$, to be introduced below. Thus, (1) may be rewritten in first order form as

$$\begin{aligned} \dot{Tb} &= F(Tb); \qquad F(Th, Tv) = \begin{bmatrix} F_{\mathcal{H}}(Th) \\ F_{\mathcal{V}}(Tv) \end{bmatrix} \\ F_{\mathcal{H}}(Th) &= \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} Th \\ F_{\mathcal{V}}(Tv) &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} Tv - \gamma \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{aligned} \tag{2}$$

and integrated [1] as

$$\begin{aligned} F^t(Tb) &= \begin{bmatrix} F_{\mathcal{H}}{}^t(Th) \\ F_{\mathcal{V}}{}^t(Tv) \end{bmatrix} \\ F_{\mathcal{H}}{}^t(Th) &:= \begin{bmatrix} I & It \\ 0 & I \end{bmatrix} Th \\ F_{\mathcal{V}}{}^t(Th) &:= \begin{bmatrix} I & t \\ 0 & I \end{bmatrix} Tv - t\gamma \begin{bmatrix} t/2 \\ 1 \end{bmatrix} \end{aligned} \tag{3}$$

[1] Here and in the sequel, we denote the flow generated by a vector field — in the present instance, $F()$ — by $F^t(\cdot)$, and the pre-image of a set, $\mathcal{S}$, under that flow, by $F^{-}(\mathcal{S}) \triangleq \cup_{t<0} F^t(\mathcal{S})$.

2.1.2 Collisions

As in [3, 4] we will assume that the components of the ball's velocity tangent to the paddle at instant of contact are unchanged, while the normal component is governed by the simplistic (but standard [5]) coefficient of restitution law yielding a change in the ball's velocity after impact

$$\Delta\dot{b} = (1+\alpha)nn^{\mathsf{T}}(\dot{r}-\dot{b}), \tag{4}$$

where n denotes the unit normal vector to the paddle at a hitting point b, and $\dot{r}$ denotes the linear velocity of the paddle at that point. It will be useful to rewrite this in state space form as defining a function of the ball's state after collision, Tb', as a function of the ball and robot's state prior to collision.

$$Tb' = C(Tb, Tr). \tag{5}$$

2.2 Robot Models

Note from (4) that a completely specified collision between a controlling surface and a controlled point mass requires six "dynamical" degrees of freedom — five kinematic degrees of freedom (three in position to make contact at the desired point; two in orientation to achieve the desired normal) and one velocity freedom, $\dot{r}$. In principal, each kinematic degree of freedom adds a velocity freedom as well (away from kinematic singularities). Further, in the physical world, the controlling surface must be at least partially extended in space (as opposed to the mathematical abstraction of an infinitesimal patch) and this effectively adds two additional degrees of freedom, albeit with kinematic range limited by the extent of the surface. From these considerations, a three actuated degree of freedom kinematic chain with a physically extended hitting surface should in principal have sufficient degrees of freedom to bat a point mass in a completely controlled manner.

In practice, velocity magnitudes are limited by actuator constraints, and the robot's accuracy in achieving a dynamical set point — a specified velocity at a specified position — is far worse than its kinematic accuracy. When the physical extent of the hitting surface is smaller than the dynamical set point resolution, the additional kinematic degrees of freedom it might have afforded are lost.

Our Buehgler robot is a three degree of freeom kinematic chain whose dynamical accuracy arguably represents the present state of the art in the field because it has direct drive actuators and a sophisticated adaptive inverse dynamics joint space controller [6]. Moreover, the hitting surface is a rectangular plane that extends roughly eighty by ten centimeters in space. Nevertheless, our lab experience has convinced us that the narrow dimension of the paddle is too small too be useful as a source of additional kinematic freedom (the minimal range of this "extra" joint is negligible with respect to the workspace of the machine). Thus, we work with a four degree of freedom kinematic model (three revolute degrees of joint space freedom and one additional prismatic freedom down the eighty centimeter length of the paddle). Thus there arises the necessity of applying four degrees of kinematic freedom to a five degree of freedom problem.

The kinematic modeling in this section assumes a paddle of infinitesimal width and infinite length. Endowing the robot with an understanding of the length limits of its paddle will be addressed in Section 4 when the notion of a dynamical obstacle is introduced.

2.2.1 The Kinematics of Normals

Let $\bar{p}$ represent (in homogeneous coordinates) a planar transformation taking points in the unit box, $\mathcal{S} \subset [0,1] \times [0,1]$ diffeomorphically onto the hitting surface (in the case of our robot, a plane) expressed with respect to the gripper frame, $\mathcal{F}_g$. For the Buehgler arm that we use, the width of the paddle is sufficiently small relative to the length that we never use it in manipulations, thus $\mathcal{S} \triangleq [0,1]$. Associated with each point on the surface, $\bar{p}(s)$ is the unit normal, $\bar{n}(s)$, again, the homogeneous coordinate representation of the vector with respect to $\mathcal{F}_g$. Denoting by $\mathcal{N}(3) = \mathcal{B} \times S^2$ the bundle of unit normal vectors at each cartesian point, the paddle's "Gauss map" [7] is now parametrized as

$$N : \mathcal{S} \to \mathcal{N}(3) : s \mapsto [\bar{n}(s), \bar{p}(s)] \tag{6}$$

For the Bueghler arm, the surface is a plane and thus $\bar{n}$ is constant.

Denote by $H(q)$ the robot's forward kinematic map taking a configuration, $q \in \mathcal{Q}$, to the homogeneous matrix representation of the gripper frame with respect to the base. The world frame representation of any paddle normal at a point is thus specified by the extended forward kinematic map, $G : \widetilde{\mathcal{Q}} \to \mathcal{N}(3)$

$$G : (\tilde{q}) \mapsto [n(\tilde{q}), p(\tilde{q})] = H(q)N(s) \tag{7}$$

where $\tilde{q} = (q, s) \in \widetilde{\mathcal{Q}} := \mathcal{Q} \times \mathcal{S}$.

The linear velocity of the hit point due the robot's motion may now be written explicitly as

$$\dot{r} = \sum_{i=1}^{dimQ} (D_{q_i} H(q)) \cdot p(s)\dot{q}_i \tag{8}$$

2.2.2 The Inverse Kinematics of Points and Normals

Denote by G_n^{-1} the inverse normal relation defined by G. That is, given a desired normal $n \in \mathbf{S}^2$, $G_n^{-1}(n)$ denotes the set of generalized coordinates, $\tilde{q} \in \tilde{\mathcal{Q}}$ that achieve it. For the four degree of freedom Buehgler arm, $G_n^{-1}(n)$ turns out to be a two dimensional surface which is easily parameterized by the horizontal workspace, $\mathcal{H} \subset \mathcal{B}$. In other words, for each horizontal position and normal vector, there is only one height above the floor at which the paddle achieves the desired normal. When we wish to pin down which horizontal position, $h \in \mathcal{H}$, has been selected we will write $G_{[n|h]}^{-1}(u)$ and this specifies a unique jointspace element, $\tilde{q} \in \tilde{\mathcal{Q}}$.

Denote by G_p^{-1} the inverse point relation defined by G. That is, given a desired spatial position, $b \in \mathcal{B}$, $G_p^{-1}(b)$ denotes the set of generalized coordinates $\tilde{q} \in \tilde{\mathcal{Q}}$ that achieve it. For the Buehgler with its essentially spherical kinematics and radially aligned wrist, $G_p^{-1}(b)$ turns out to include all $q_3 \in \mathbf{S}^1$. In other words, to reach a specified spatial position there remains only a freedom in wrist rotation. When we wish to pin down which wrist angle, $q_3 \in \mathbf{S}^1$, has been selected we will write $G_{[p|q_3]}^{-1}(b)$, and this specifies a unique jointspace element.

Lying in the total configuration space, $\mathcal{B} \times \tilde{\mathcal{Q}}$, is the *contact submanifold*, $\mathcal{C} \triangleq \mathcal{B} \times G_p^{-1}(b)$, of balls that are touching the robot. For the Buehgler, this is a union of all the "wrist" contacts,

$$\mathcal{C} = \bigcup_{q_3 \in \mathbf{S}^1} \mathcal{C}_{q_3} \qquad \mathcal{C}_{q_3} := \mathcal{B} \times G_{[p|q_3]}^{-1}(b). \tag{9}$$

2.3 The Control Architecture

To aid the reader in making sense of the discussion (and notation!) to follow, it seems worth pointing out from the beginning that there will be two different "time-scales" of interest over which control must be asserted and studied. There are the continuous states, (Tq, Tb) governed by the the integrated flight model driving the robot's reaction according to a "mirror law" as dictated by the controller, Φ^i, which will be discussed first. Depending upon the controller, Φ, selected, the ball may experience a flight phase. For each flight there is a ball *apex* that we denote $\overline{Tb}$ — the state of a ball following an impact with the robot at which its vertical velocity vanishes — governed by the closed loop dynamics, f_Φ induced by Φ.

2.3.1 The Sensing, Planning and Actuating Subsystems

Our "Cyclops" stereo vision system delivers a stream of image plane centroid pairs, i, obtained by taking first moments over windows managed in a manner determined by the sensor's "perceptual state," d that we denote $i = s_d(b)$ [8]. The perceptual state, d, evolves according to the dynamics of an estimation scheme, E, and driven by the sensors' centroid reports, i, to arrive at a state estimate for the ball, $\widehat{Tb}$,

$$\begin{aligned} \dot{d} &= E(d, i) \\ \widehat{Tb} &= e_d(i). \end{aligned} \tag{10}$$

Since balls usually go in and out of viewing range or are otherwise occluded, this system is designed to operate for periods of time when only one or possibly neither camera delivers data. The estimator (10) then functions as a predictor, a monocular observer, or a stereo observer according to switching logic specified in a state-based manner [8].

Reference trajectories, $Tq^*(t)$ are tracked by an adaptive inverse dynamics controller

$$\begin{aligned} \dot{Tq} &= C_\theta(Tq, Tq^*) \\ \dot{\theta} &= \Theta(\theta, Tq, Tq^*) \end{aligned} \tag{11}$$

that has been documented at length in [6].

Given a stream of estimator data describing the environment's state, $\hat{b}$, a smooth map,

$$m : T\mathcal{B} \to \mathcal{Q}, \tag{12}$$

from ball states to robot joint positions, generates trajectories of the form, $m \circ \widehat{Tb}(t)$, that (along with their time derivatives) can be fed to the controller (11) as reference signals, Tq^*. Buehler called m a *mirror law* since a robot that successfully tracks b with $\hat{b}$ and Tq^* with Tq behaves as though it were "mirroring" the environment (in some inverted or distorted manner that obviously depends upon the choice of m).[2]

[2]This represents an extreme form of Andersson's [9] policy

2.3.2 Continuous Closed Loop Dynamics, Φ

Given a model of the environment's dynamics, e.g. (2), one may examine the overall "filter" from sensor stream to robot motion as a continuous dynamical system,

$$\Phi \begin{cases} \dot{Tb} &= F(Tb) \\ \dot{d} &= E(d, s_d(Tb)) \\ \dot{Tq} &= C_{\theta^*}(Tq, Tm \circ e_d \circ s_d(b)), \end{cases} \tag{13}$$

on an extended state, space, $\mathcal{X} := T\mathcal{B} \times \mathcal{D} \times T\mathcal{Q}$, $\dot{x} = \Phi(x)$.

It is now convenient to assume that the extended dynamics on $\mathcal{X}$ have a "fast" component that collapses onto $T\mathcal{B}$ [3]. Under these "collapsed" assumptions, we have $Tq = Tm \circ Tb$ so that

$$\Phi^t(x) \approx (F^t, Tm \circ F^t)(Tb) \tag{14}$$

Two examples of closed loop flows relevant to the concerns of this paper now follow. In the first case, a tracking law, m_T, causes the robot to follow the ball without making contact so that Φ_T is defined on all of $\mathcal{X}$. In the second case, a palming law, m_P, actually effects changes in the ball's dynamics when on the contact set, $\mathcal{C}$, so that the interesting closed loop behavior occurs on a restricted submanifold of $\mathcal{X}$.

The Tracker: Φ_T Surely the simplest instance of a useful flow is that induced by the horizontal "tracking" mirror,[4]

$$m_T := G^{-1}_{[p|0]}(\pi_{\mathcal{H}} b), \tag{15}$$

developed by Rizzi to keep the Buehgler's paddle "under" the horizontal projection, of a falling ball. It is immediate from (14) that when the robot tracks this reference, $Tq \to Tq^*$, we have $\pi_{\mathcal{H}} b \equiv \pi_{\mathcal{H}} G(q)$.

of "keeping your eye on the ball" and works admirably well as a means of entraining the robot's actions with the environment it seeks to alter. It has the obvious drawback of requiring unwavering attention from the sensor system (10) that can be ammended to a greater or lesser extent as desired by adding state.

[3] When θ is held constant in (11) then $Tq(t) \to Tq^*(t)$ [6], uniformly. We continue to work on the problem of demonstrating that (10) implies $\widehat{Tb}(t) \to Tb(t)$ uniformly, but the present paper leaves no room for a discussion of the technical issues involved. Laboratory experience indicates that these uniform convergence rates are sufficiently "fast" relative to the ball's dynamics that they can be ignored at the higher level. Establishing conditions on the underlying computational environment that guarantee this situation remains an open problem that we have considered only sketchily to date [2].

[4] The symbol π denotes a projection onto the subspace indicated in the subscript.

The Palmer: Φ_P Consider the "proportional derivative" mirror,

$$m_P(Tb) = G^{-1}_{[n|h]}\left(L_1(h^* - h) - L_2\dot{h}\right) \tag{16}$$

where $L = [L_1, L_2]$ are constant matrices. This map is introduced to correct for deviations from the horizontal setpoint, $Th^* = (h^*, 0)$. A one degree of freedom version of this mirror was introduced by Buehler for the planar arm and generalized by Rizzi for the spatial Buehgler arm. If the ball remains on the paddles and the robot tracks this reference trajectory then the ball's horizontal dynamics (1) are altered as

$$\ddot{h} = \pi_{\mathcal{H}}\left[I - nn^{\mathrm{T}}\right]\begin{bmatrix} 0 \\ 0 \\ -\gamma \end{bmatrix} \tag{17}$$

so that Φ_P implements a classic "ball-on-beam" controller. The robot rolls the ball about on the paddle bringing it eventually to the horizontal position, h^* at zero velocity.

2.3.3 Discrete Closed Loop Dynamics f_Φ

We now add a "batting" mirror law, m_B. Once the ball leaves the paddle the robot will need to constantly revisit it in order to maintain control. Thus, common sense dictates that the most basic criterion of any reasonable trajectory generator (13) is to arrange for such "returns" from all non-goal states so that the robot is at least guaranteed of "getting a crack" at them. Indeed, all of our re-grasping generators to date, whether for dynamical manipulation or quasi-static assembly achieve this property. We are led to consider as state space a "Poincaré Section" in $\mathcal{X}$ that we will term the dynamical workspace. There follows a description of general return maps, f_Φ, induced by controllers, Φ, that guarantee return. This section concludes with a presentation of m_B and the return map, f_{Φ_B} that it generates.

The Dynamical Workspace Denote by $\mathcal{I} = T\mathcal{C} \subset T\mathcal{Q} \times T\mathcal{B}$ the *impact set* — those states (configuration and velocity) where the robot is touching the ball. Denote by τ_Φ the time of first return to $\mathcal{I}$ — that is, the solution for t of the inclusion

$$\Phi^t(x) \in \mathcal{I}$$

The reference generators we build guarantee that $\mathcal{I}$ is always in the domain of τ_c under the dynamics, Φ (13).

Instead of working in the pre- or post- contact states we will find it useful to work in a different transverse "section" along the trajectories of Φ — the *apex coordinates*. Denote by τ_a the time that a rising ball (e.g., just after impact) will reach its apex — that is, the time to achieve zero vertical velocity. Denote by τ_c the time to impact from apex — that is, the first as a function of Tb. Denote an apex state as $\overline{Tb} := F^{\tau_a}(Tb) = (\overline{Th}, \overline{Tv})$, where the velocity component of the vertical component is zero — $\overline{Tv} = (\bar{v}, 0)$.

Of course, the dynamical workspace also has a representation in the apex coordinates, $\overline{\mathcal{I}} = F^{\tau_a}(\mathcal{I})$, and it is on this set that we are interested in computing the return maps.

The General Return Map The effect of the robot on the environment results from integrating forward from a previous post contact state, x_k to the next contact state, applying the impact model at that point and picking off resulting environment state. This induced behavior is denoted

$$f_\Phi(\overline{Tb}) = F^{\tau_a} \circ C(F^{\tau_c}(\overline{Tb}) \ , Tm \circ F^{\tau_c}(\overline{Tb}) \ ). \quad (18)$$

where we have expressed the map in apex coordinates on $\overline{\mathcal{I}}$.

It is now clear that the iterates of (18) govern the future evolution of the environment. The prime example is $f_{\Phi_B^1}$ of such a closed loop map relevant to the concerns of the paper now follow. Note again, while the mirror laws, m, take their domain on the continuous ball flight, Tb, the closed loop maps, f_Φ take their domain on the discrete sequence of apex states, $\overline{Tb}$ induced by the mirror law.

The Vertical Batter: Φ_B Buehler's original construction [3] was a one degree of freedom mirror law,

$$m_B^1(Tv) := \kappa(\overline{Tv})G_p^{-1}(v) \quad (19)$$

for a "gedanken" prismatic robot piston (i.e., a machine whose "kinematics," G, are given by the identity map on $\mathcal{V}$). He showed that this yields a controller, Φ_B^1 , that induces a unimodal closed loop map,

$$f_{\Phi_B^1}(\overline{Tv}) = (\alpha + \kappa(1+\alpha))^2 \overline{Tv} \quad (20)$$

[10, 11], on $\overline{T\mathcal{V}}$ subject to the flight dynamics (2). Since κ in (19) amounts to a "proportional control" with respect to the desired total energy, one obtains a globally asymptotically stable vertical period orbit. Beyond this strong stability mechanism, these maps offer as well a cascade of period doubling bifurcations [4] that can be readily tuned in the mirror law itself, m_B^1 [11]. In other words, $\mathcal{G}(\Phi_B^1)$ can be tuned via gains in the robot controller (13) to consist of a single point (a periodic vertical trajectory passing through the same apex point), two points (a doubly periodic vertical trajectory passing through a high apex and then a lower apex), four points, and so on. Moreover, regardless of the setting, this goal is achieved by all (excepting at most a set of measure zero) initial conditions originating in an arbitrarily large bounded interval of the state space.

3 New Controllers from Old: Compositions

Given two goals, $\mathcal{G}_1, \mathcal{G}_2$, and two flows, Φ_1, Φ_2 whose associated discrete maps, f_{Φ_1}, f_{Φ_2} achieve them, we would like to build composition procedures that allow them both to be achieved by the same robot. Given the architecture described in the previous section, such compositions arise from a suitable combination of the respective mirror laws, m_1, m_2. In this section we introduce three disctinct composition techniques.

The first two techniques represent past work. In practice, we have developed a family of working controllers for juggling, Φ_J [3, 2], that we build up from the simpler "batting," Φ_B, "palming," Φ_P, and "tracking," Φ_T, controllers discussed above. The juggling machines work quite reliably in the laboratory. Both our planar and our Buehgler robots juggle for literally hours at a time, resist severe, nearly adversarial perturbations, and display the predicted response to tuning. We review in this section our present understanding of the theoretical basis for their success.

The final composition technique represents work in progress, and we reserve the next section of the paper for a detailed account of its application (alongside of the previous two techniques whose efficacy is now a matter of record).

3.1 "Interleaving" Compositions: $\wedge$

At the center of our interest in dynamical dexterity is the possibility of achieving several different tasks at the same time with one robot. Thus, the most important composition technique that we have developed addresses the necessity of sharing actuated degrees of freedom between competing tasks.

3.1.1 Interpolation

The common mechanism we have developed for shifting the robot's "attention" between one or the other controller obtains from a partition on the relevant features of the environment, that express the respective "urgency" of the competing tasks. Since the trajectory controller (11) can only track smooth trajectories, we implement this partition via an analytic interpolation function,

$$\Lambda_\sigma(x,y) \triangleq \begin{cases} x & : \sigma = 1 \\ y & : \sigma = 0 \end{cases} \tag{21}$$

where σ is an analytic version of the standard partition of unity [12].

3.1.2 Planar One-Juggle

We have been able to build "jugglers" by interleaving palming and batting controllers,

$$\Phi_J^2 = \Phi_B^1 \wedge \Phi_P^1.$$

Buehler's original mirror, $\tilde{m}_J$, was formed as the sum $m_B^1 + m_P^1$, of the palmer and batter. The associated controller, $\tilde{\Phi}_J$ worked extremely well in the laboratory. Unfortunately, $f_{\tilde{\Phi}_J}$ proved to be analytically intractable because the time of flight,τ_c, associated with this strategy requires the solution of a transcendental implicit function. Nevertheless, intuition held that the periodic vertical impacts introduced an effective sampling interval on the horizontal double integrator (17), imposing a discrete second order linear proportional derivative closed loop

$$f_{\Phi_P^1}(\overline{Th}) := \begin{bmatrix} I & (\tau_c + \tau_a)I \\ 0 & I \end{bmatrix} \overline{Th} + \begin{bmatrix} \tau_a I \\ I \end{bmatrix} n_h u \tag{22}$$

where $n_h := \pi_{\mathcal{H}} n(\tilde{q})$ denotes the component in the $\mathcal{H}$ subspace of the paddle normal at impact and

$$u := n(\tilde{q})^{\mathrm{T}} \left(\dot{b} - \dot{r}\right) \tag{23}$$

denotes the difference in linear velocity at hit point between the paddle and ball arising from (4) and (8). Unfortunately, no closed form expression of the resulting closed loop, $f_{\tilde{\Phi}_J^2}$ has ever been derived. While it is easy to show that $f_{\tilde{\Phi}_B^2}$ has a locally attracting fixed point, the global stability properties of this mapping have never been established. We will instead, develop a new version of juggling.

Rizzi [13] introduced a solvable planar one-juggle via the following mirror law.

$$\begin{aligned} m_J^2(Tb) &= \Lambda_{t/\tau_c}\left(m_B^1, 0\right) \\ &\quad + m_P^1 \circ F^{\tau_c}\left(\overline{Tb}\right) \end{aligned} \tag{24}$$

that achieves the vertical velocity of m_B^1 (19) with paddle postion and attitude at impact specified by m_P^1 (16) by "switching" on the relative time to contact, t/τ_c. In this sense, we consider Φ_J^2 to be the interleaved composistion of Φ_B^1 and Φ_P^1,

The closed loop map is given by

$$\begin{aligned} f_{\Phi_B^2}(\overline{Tv}, \overline{Th}) &= \begin{bmatrix} f_{\Phi_B^1}(\overline{Tv}) + \delta_1(\overline{Th}) \\ f_{\Phi_P^1}(\overline{Th}) + \delta_2(\overline{Tv}) \end{bmatrix} \\ &= f_{\Phi_B^1} \times f_{\Phi_P^1} + \delta \end{aligned} \tag{25}$$

where the δ entries represent higher order coupling terms that vanish at the setpoint $\overline{Tv}^*, \overline{Th}^*$.

3.2 Parallel Compositions, $\times$

When the distinct goals require fewer degrees of freedom than available to the robot, then both should be implemented simultaneously. The parallel composition, $\Phi_1 \times \Phi_2$ is intended to realize this end.

The Spatial One-Juggle To get a three degree of freedom a generalization of the planar juggler controller to the spatial Buehgler arm, we used the two new degrees of freedom to implement the tracker m_T^2

$$m_J^3 := m_J^2 + m_T^2 \tag{26}$$

and thus,

$$\Phi_J^3 = \Phi_T^2 \times \left(\Phi_B^1 \wedge \Phi_P^1\right).$$

3.3 Sequential Compositions, $\bigvee$

Say that controller Φ_1 *prepares* controller Φ_2, denoted $\Phi_1 \succeq \Phi_2$, if its goal lies within the domain of the second $\mathcal{G}(\Phi_1) \subset \mathcal{D}(\Phi_2)$. For any set of controllers, $\mathcal{U} = \{\Phi_1, ..., \Phi_N\}$, this relation induces a directed generally cyclic graph.

Assuming that the overall task goal set, $\mathcal{G}$, coincides with the goal set of at least one controller, $\mathcal{G}(\Phi_i) = \mathcal{G}$, then by starting with Φ_i and recursively tracing the relation backwards through the corresponding graph, one arrives at $\mathcal{U}_\mathcal{G} \subset \mathcal{U}$ — the set of all controllers from whose domains the overall goal might be eventually reached by switching from one to the next: the domain of a properly conceived composite controller, $\Phi = \bigvee \mathcal{U}_\mathcal{G}$, should then be

$$\mathcal{D}\left(\bigvee \mathcal{U}_\mathcal{G}\right) = \bigcup_{\Phi \in \mathcal{U}_\mathcal{G}} \mathcal{D}(\Phi). \qquad (27)$$

Such a sensor based composite results naturally from a partition of $\mathcal{I}$ into cells "governed" by the members of $\mathcal{U}_\mathcal{G}$ as follows. The relation, $\succeq$, induces a partial order (possibly more than one) on $\mathcal{U}_\mathcal{G}$ with a minimal element, Φ_i, as may readily be computed by pruning the original graph into an acyclic graph with root at Φ_i (e.g., perform a breadth-first search back from that node) that we call a *preparation graph* for Φ_i and denote by Γ. This partial order now induces a partition on the dynamical workspace, $\mathcal{I}$, obtained from an arrangement of the domains, $\{\mathcal{D}(\Phi_i)\}_{\Phi_i \in \mathcal{U}_\mathcal{G}}$, of the participating controllers in the obvious manner[5].

We have achieved reasonable experimental success with algorithms of this form where the placement and tuning (hereafter referred to as the *deployment*) of the individual controllers is carefully designed by hand. We report here a specific instance of such a "hand-designed" deployment. Ultimately we expect to build systems where the controller deployment is automatically generated from user-specified task goals.

4 Application: Obstacle Avoidance

Juggling and hopping are examples of dynamical manipulations that incur the repeated need for "regrasping" — letting go of the manipuland while in the course of bringing it to some more desired state. The need for regrasping arises in many robotic manipulation scenarios, assembly being among the most important. We now explore the composition of batting controllers to dodge obstacles that would otherwise have disconnected the configuration space.

4.1 Obstacles and Safety

A barrier for the robot, $\mathcal{O}_\mathcal{R} \subset \mathcal{R}$, or ball, $\mathcal{O}_\mathcal{B} \subset \mathcal{B}$ consists of those states that entail either one penetrating a physical object. A workspace *obstacle*, $\mathcal{O} := \mathcal{O}_\mathcal{B} \cup \pi_\mathcal{B}\left(\mathcal{I} \cap \pi_\mathcal{R}^{-1}\mathcal{O}_\mathcal{R}\right)$, corresponding to a ball barrier comprises the union of the ball barrier with those nearby ball states that the robot cannot reach while itself avoiding a robot barrier. The first return of a ball with respect to a workspace obstacle, is defined to be its *urgency*, $\tau_\mathcal{O} : F^-(\mathcal{O}) \to \mathbb{R}^+$. [6]

Say that a robot policy (13) Φ is *safe with respect to* $\mathcal{O}$ if there can be found an open subset in the free dynamical workspace, $\mathcal{I}$,

$$\mathcal{D}_\mathcal{O}(\Phi) := \left\{b \in F^-(\mathcal{O}) \cap \mathcal{D}(\Phi) \; : \tau_\Phi(b) < \tau_\mathcal{O}(b)\right\}$$

that is left invariant by f_Φ,

$$f_\Phi\left(\mathcal{D}_\mathcal{O}(\Phi)\right) \subset \mathcal{D}_\mathcal{O}(\Phi) \qquad (28)$$

In other words, a safe controller for $\mathcal{O}$ yields a *safe domain* of states, $\mathcal{D}_\mathcal{O}(\Phi)$, from which it can be assured that forthcoming contact will be made before the ball penetrates the obstacle and that this assurance will remain after the next contact. Notice that for any obstacle and controller, a safe domain is invariant during flight, $F^t\left(\mathcal{D}_\mathcal{O}(\Phi)\right) \subset \mathcal{D}_\mathcal{O}(\Phi)$, by its very definition. Thus, once Φ has been selected, only a violation of our models — e.g., a perturbation during flight or impact, a overly large observer error, etc. — can cause a transition out of $\mathcal{D}_\mathcal{O}(\Phi)$.

4.2 Safe Deployments

To achieve obstacle avoidance in the previous section, we have used numerical methods to compute conservative approximations to the set $f_\Phi^-(\mathcal{O})$. Suppose the controller $\Phi = \Phi_P \wedge \Phi_B$ is the composition of controllers that decouple the horizontal and vertical degrees of freedom of the ball,

$$f_\Phi = f_{\Phi_P} \times f_{\Phi_B}.$$

Suppose, moreover, as in this problem, that the obstacle set is cylindrical over $\mathcal{H}$. Then we may use the factor f_{Φ_P} alone as the mechanism of safety — i.e., we will use conservative approximations to the *safe space* defined by Φ_P

$$\mathcal{S}_\mathcal{O}(\Phi_P) := f_{\Phi_P}^-(\mathcal{O})$$

to plan our avoidance maneuvers.

4.3 Juggling Across a Horizontally Disconnected Workspace

We now introduce to the robot's workspace a flat obstacle at the height of the paddle when held flat. Fig-

[5]For example, controller Φ_1 is assigned that portion of its domain, $\mathcal{D}(\Phi_1)$, not contained in that of any "$\succeq$-lesser" controller; controller Φ_{k+1} is assigned the portion of its domain not contained in that of either a $\succeq$-lesser controller one of index $j \leq k$.

[6]The notation of this paragraph was introduced in footnotes 4 and 1

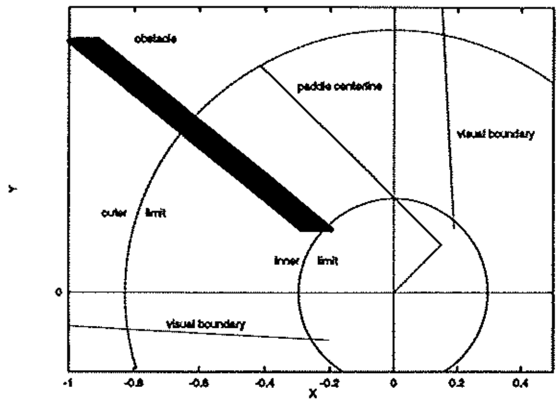

Figure 2: Horizontal configuration space with obstacle and visual boundaries.

ure 2 depicts the obstacle's locus within the planar set defined by these horizontal robot configurations — an annular region owing to the paddle's finite length — intersected with the horizontal projection of the visual workspace. It should be clear from the figure that the pure palming controller, Φ_P, sees a disconnected workspace. Thus, in addressing the problem of how to bring a ball from one to the other connected component, we encounter a simple case of the situations described above wherein a re-grasp maneuver is necessary to the success of the task. This section describes our application of the safe deployment methodology to the re-grasping problem.

When a controller has a goal within a convex domain, then it is not critical what trajectory the state of the ball takes inside the domain as it moves toward the goal. It is enough to know that the domain is invariant, and the goal is attracting. In fact, little changes whether we consider the ball dynamics discrete or continuous. On the other hand, in the presence of a disconnected workspace there must be at least one controller with a domain of attraction to its goal that is disconnected. Since the domain of attraction of an equilibrium state for any continuous dynamical system is homeomorphic to a ball, we require in this application a controller that induces a discrete dynamical system.

As we have described in the previous sections, the juggling controller, Φ_J induces just such a discrete dynamics on the impact set, $\mathcal{I}$. Moreover, when the vertical component is precisely regulated, $Tv = Tv^*$, then the juggling mirror law, Φ_J selects impacts over this horizontal workspace. In this case, we have

$$f_{\Phi_J}(Tb) = f_{\Phi_B^1}(Tv^*) \times f_{\Phi_P^2}(\pi_{\mathcal{H}} Tb)$$

and we may use $f_{\Phi_P^2}$, its "τ_c^*-sampled discrete version" to plan deployments of Φ_P^2. In this paper we have used conservative numerical simulations of the closed loop dynamics to achieve the desired safe deployment.

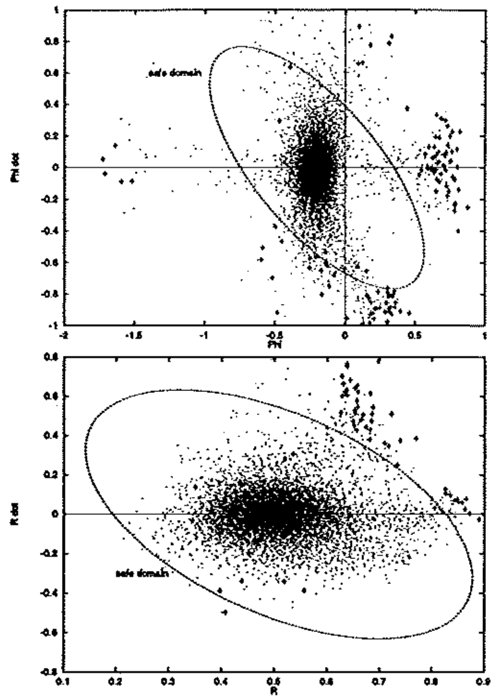

Figure 3: Empirically derived estimate for the horizontally safe juggling domain. The symbol(s) X ($\cdot$) denote the points which failed (succeeded) in remaining within the robot's annular and visual workspace under the action of Φ_J.

4.3.1 A Numerical Approximation to $f_{\Phi_P^2}$

In Figure 3, we display test data used to formulate an approximation of the safe domain for the juggling controller operating within the bounded workspace formed from the robot's paddle and visible region — the region of Figure 2 with no flat obstacle inserted. The "+" symbols represent starting conditions which the robot was unable to regulate safely, and the "." symbols represent starting conditions successfully brought to the set point.

In Figure 4, we display the same test data run through the numerical simulation of the robot operating under Φ_J. Since all but one of the starting conditions that failed on the real robot failed in simulation, we are persuaded that the simulation provides a conservative approximation to the robot's true actions.

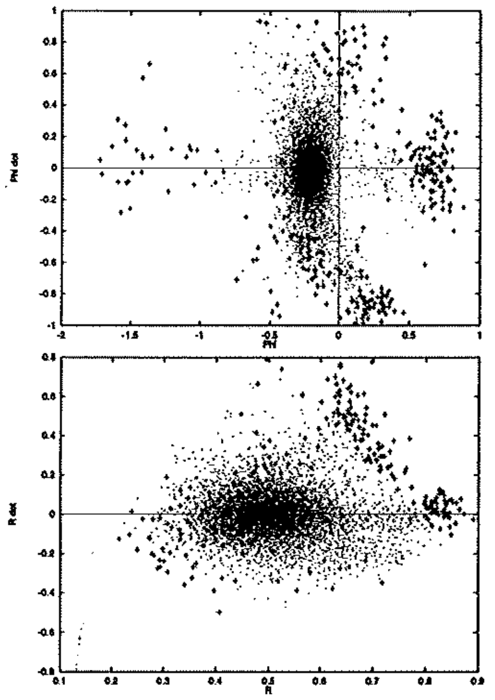

Figure 4: Numerically derived estimate for the horizontally safe juggling domain. The symbol(s) "+" (".") denote(s) the points which failed (succeeded) in remaining within the robot's annular and visual workspace under the action of the numerical simulation of Φ_J.

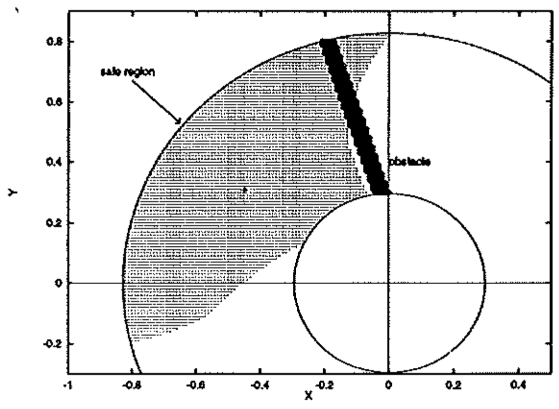

Figure 5: Simulation results: the largest safe subset of the free workspace defined by Φ_J.

Figure 5 depicts the results of a simulation run to establish the largest safe domain of Φ_J with respect to the disconnected obstacle-cluttered free workspace illustrated in Figure 2. The shaded region consists of free points that reach the goal — the spot labelled "+" in the figure — and remain free (that is, they stay clear of the obstacle, stay within the visible workspace, and remain within the reach of the paddle) along the way. This, of course, is merely the zero velocity subset of a four dimensional region that extends in a rather complicated manner into $T\mathcal{H}$.

4.3.2 A Computationally Derived Safe Deployment

At the present time, we have no convenient means of parametrizing the largest safe subset of Φ_J (whose projection onto $\mathcal{H}$ is depicted in Figure 5). Yet to build the sequential compositions laid out in this paper we require just such a parametrization. Thus, we are led to find smaller safe domains whose shape is more readily represented. Ellipses make an obvious candidate shape and have an analytical justification described in the appendix.

In Figure 6, we show four slices through the surface of an invariant four-dimensional ellipsoid, along with the same slices of the forward image of that surface computed by a numerical simulation of Φ_J operating under the mirror law, m_J. Although these pictures do not in themselve suffice to guarantee invariance they provide a useful visulization of what is going on. Indeed, these four projections were used as the primary tool in a search by hand for an invariant ellipsoid using the simulation. When a promising candidate was created by rotating and elongating the four-dimensional ellipsoid, the forward image was tested for inclusion in the original ellipse. After many trials, the ellipse shown in Figure 6 was found and shown to map entirely within itself. [7]

Once a single invariant ellipsoid is found for a particular set point, several of such ellipsoids may be placed within the workspace by making rotationally symmetric copies (the mirror law, m_J, is rotationally symmetric). This allows us to create a complex controller on each side of the obstacle. The one on the goal side draws all states within the copies of the in-

[7]This process will be sped up enormously by access to an analytically tractable return map of a kind introduced in the appendix. Even without analytical results, the search for invariance could be automated by recourse to positive definite programming (convex nonlinear programming problem) on the parameters (in this case, the entries of a positive definite matrix) defining the shape of the invariant set.

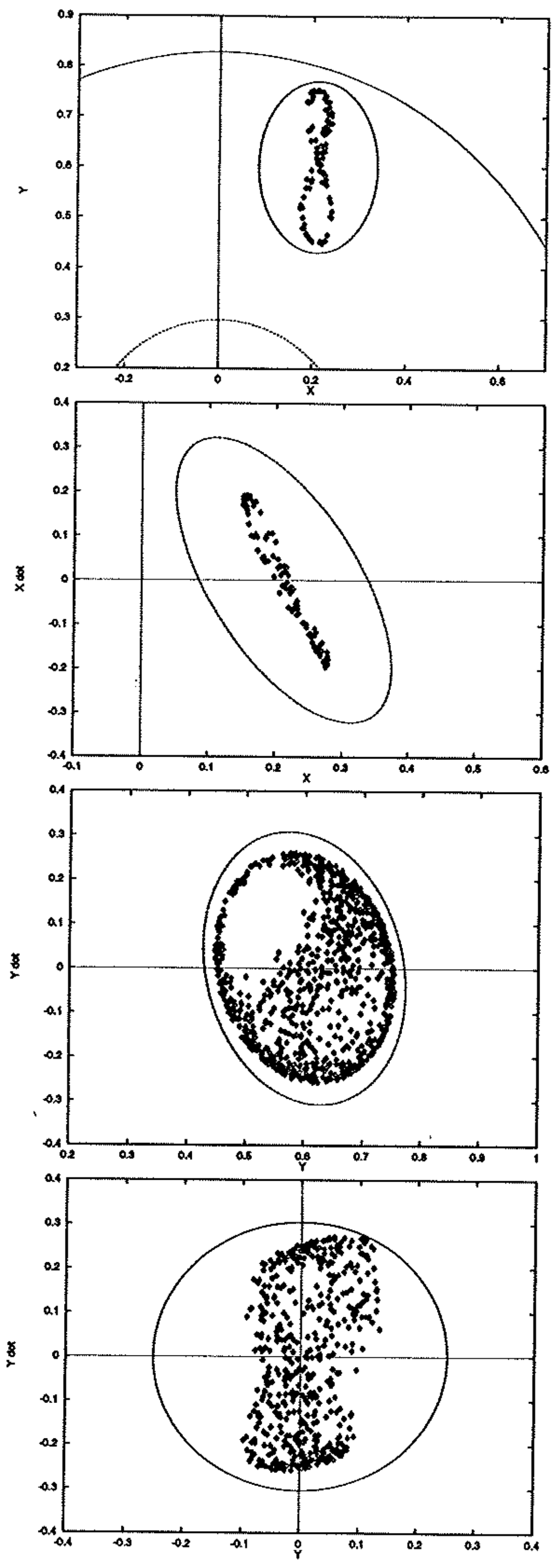

Figure 6: Four 2-D slices of the invariant ellipse.

variant domain toward the task goal point (which is assumed to be the goal of one of the constituent controllers). The one on the disconnected-from-goal side draws a similar union of domains to a goal point near the obstacle. The next task is to develop a controller that will cross the obstacle.

To get across the obstacle, we start with two controllers and domains of interest – the highest priority pre-obstacle controller, and the lowest priority post-obstacle controller, each with its own invariant domain. The latter will form the basis of the jumping, or trans-obstacle controller. Since the pre-obstacle controller will regulate the ball to arbitrary accuracy (at least in simulation), we can choose a small neighborhood around that set point as a "launching pad" for the trans-obstacle controller. Having chosen such a ball, we take the forward image under the mirror law in simulation. The result may no longer be invariant, but we can readily find an ellipsoid that contains it. The new "containing" ellipsoid is now mapped forward and its image is once again contained in some new ellipsoid. Iterating this process, we eventually find an invariant ellipsoid that contains the forward image of the n^{th} "container" ellipsoid *and* lies within the original invariant domain. The union of all these ellipsoids is invariant and safe under the trans-obstacle controller, as the forward image of each member of the union is fully contained within another, and they all lead into the safe invariant domain already found. In Figure 7 we show a union of ellipsoids created by this process projected onto four orthogonal planes.

In Figure 8, we depict the the entire hand-crafted deployment discussed so far by projecting the domain of each onto the horizontal configuration space introduced in Figure 2. There are a total of fifteen ellipsoids for nine different controllers, the union of six of them associated with the trans-obstacle controller.

The deployment of Figure 8 gives rise to a safe controller under the $\bigvee$ composition described previously. Unfortunately, due to the conservative nature of our numerical procedure, the domain of attraction of the composite is not large enough to lend confidence that a ball could easily be introduced into it. Worse, given the inevitable infidelities of the robot and ball with respect to the ideal models (1), (5), (7), a physical implementation will remain roughly faithful to the simulation results, but inevitably depart frequently enough and with significant enough magnitude that some further "safety net" is required to bring balls that start (or, "unexpectedly stray") outside the domain of the composite back into it.

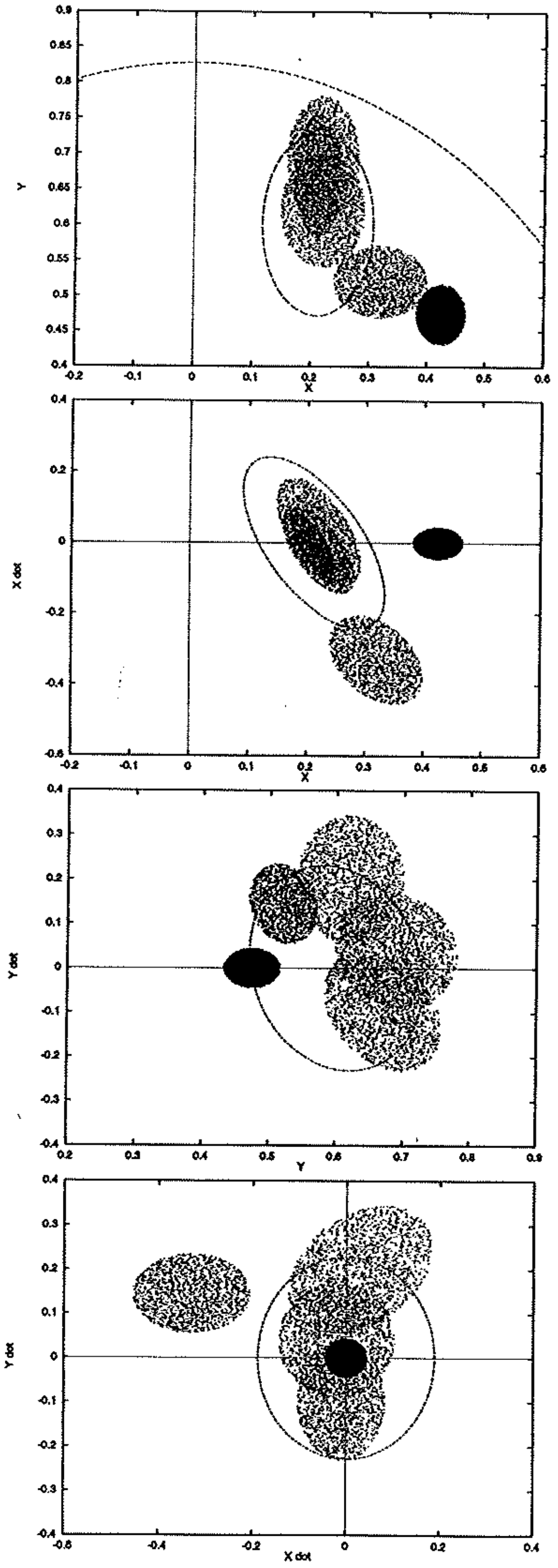
Figure 7: Jump controller invariant union of ellipsoids.

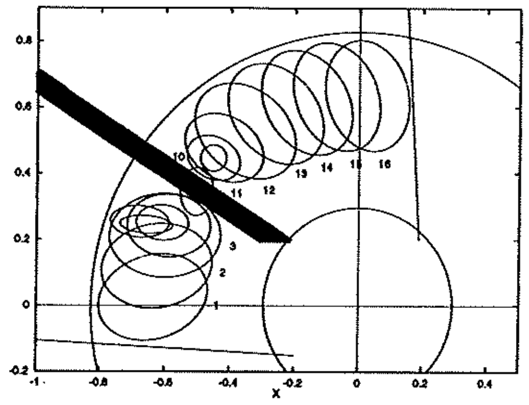
Figure 8: A "theoretical" deployment using the jump controller of Figure 7.

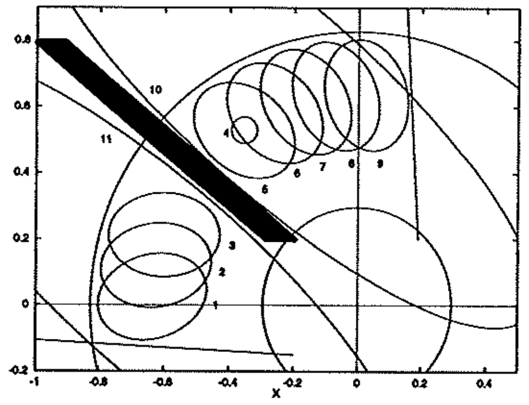
Figure 9: The implemented deployment, adding a "catch-all" (an ellipsoidal approximation to the largest safe domain) to supplement the jump controller.

For purposes of this initial study, we have chosen to use very coarse (and, unfortunately, not necessarily conservative) approximations to the largest safe domain of Φ_J depicted in Figure 5. These take the form of large ellipses that share some of their boundary (at zero velocity) with the disconnecting obstacle as depicted in Figure 9. The robot is assigned the corresponding juggling controller whenever the ball is outside the domain of the sequential composite depicted in Figure 8, yet still within the reachable workspace.

4.3.3 Empirical Results to Date

In figure 10 we show several typical traces of the control mode plotted against time, with the height of the

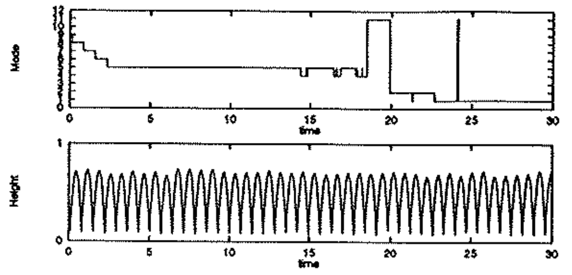

Figure 10: Anecdotal time trace of vertical position and mode number for a typical run.

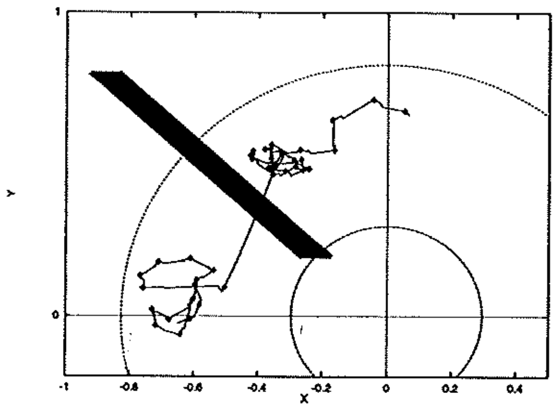

Figure 11: Horizontal projection of the sample trace above.

ball also plotted to provide a sense of time scale. [8]

Figure 11 now plots for this same typical run the horizontal positions at each hit (depicted by a box symbol in the plot) to convey a sense of where the domains for these various controllers reside in the workspace.

References

[1] Alfred Rizzi and Daniel E. Koditschek. Preliminary experiments in robot juggling. In *Proc. Int. Symp. on Experimental Robotics*, Toulouse, France, June 1991. MIT Press.

[2] Alfred A. Rizzi, Louis L. Whitcomb, and D. E. Koditschek. Distributed real-time control of a spatial robot juggler. *IEEE Computer*, 25(5), May 1992.

[3] M. Bühler, D. E. Koditschek, and P.J. Kindlmann. A family of robot control strategies for intermittent dynamical environments. *IEEE Control Systems Magazine*, 10:16–22, Feb 1990.

[4] D. E. Koditschek and M. Bühler. Analysis of a simplified hopping robot. *International Journal of Robotics Research*, 10(6), Dec 1991 .

[5] J. B. Keller. Impact with friction. *ASME J. Appl. Mech.*, 53:1–4, 1986.

[6] Louis L. Whitcomb, Alfred A. Rizzi, and Daniel E. Koditschek. Comparative experiments with a new adaptive contoller for robot arms. *IEEE Transactions on Robotics and Automation*, 9(1):59–70, Feb 1993.

[7] John A. Thorpe. *Elementary Topics in Differential Geometry.* Springer-Verlag, New York, 1979.

[8] A. A. Rizzi and D. E. Koditschek. A dynamical sensor for robot juggling. In Koichi Hashimoto, editor, *Visual Servoing — Automatic Control of Mechanical Systems with Visual Sensors*, pages 229–256. World Scientific, 1993.

[9] R. L. Andersson. Understanding and applying a robot ping-pong player's expert controller. In *Proceedings 1989 International Conference on Robotics and Automation*, pages 1284–1289, Scottsdale, AZ, 1989.

[10] M. Bühler , D. E. Koditschek, and P.J. Kindlmann. A Simple Juggling Robot: Theory and Experimentation. In V. Hayward and O. Khatib, editors, *Experimental Robotics I*, pages 35–73. Springer-Verlag, 1990.

[11] M. Bühler and D. E. Koditschek. From stable to chaotic juggling. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1976–1981, Cincinnati, OH, May 1990.

[12] H. L. Royden. *Real Analysis.* MacMillan, NY, 1968.

[13] A. A. Rizzi and D. E. Koditschek. Further progress in robot juggling: Solvable mirror laws. In *Int. Conf. Rob. and Aut.*, pages 2935–2940, 1994.

[8]Note that the presently working machine doesn't actually respect the painstakingly hand-crafted trans-obstacle mode because the simulation is not accurate enough in the face of system noise. Instead, the ball is captured after a trans-obstacle controller episode by the large "safety net" controllers depicted in Figure 9. .