Preprint of a Paper Presented at
The Fifth International Symposium on Robotics Research
Tokyo, Japan, 1989

Planning and Control of Robotic Juggling Tasks

M. Bühler, D. E. Koditschek, and P. J. Kindlmann [1]

Center for Systems Science
Yale University, Department of Electrical Engineering

Abstract

A new class of control algorithms — "mirror algorithms" — gives rise to experimentally observed juggling behavior in a simple robotic mechanism. The simplest of these algorithms (upon which all the others are founded) is provably correct with respect to a simplified model of the robot and its environment. This paper reviews the physical setup and underlying mathematical theory, discusses two significant extensions of the fundamental algorithm, provides data from our successful empirical verifications of these control strategies, and briefly speculates upon the larger implications for the field of robotics.

1 Introduction

We have built a one degree of freedom robot capable of juggling two pucks falling freely on a frictionless plane inclined into the earth's gravitational field. The robot responds sensibly to logically distinct circumstances: when it senses the presence of one puck it will attempt to juggle the single body into a prescribed periodic motion; when it senses the presence of a second body it will attempt to adjust the intervals of time between and velocity of impacts with both bodies until they reach the prescribed periodic trajectories with the desired phase angle relationship between them. The juggling algorithm works on the principles of feedback theory and implements what might be called "visual servoing:" the sensor based algorithm translates puck states into an online reference trajectory for the robot controller via a carefully chosen nonlinear function. Thus, the robot is "programmed" using a mathematical formula rather than an expert system or some other "syntactic" means. It succeeds over a wide range of initial puck locations and recovers gracefully from unexpected perturbations of the puck states during flight.

This paper reviews the experimental setup and abstract theory we have developed. It describes the geometric constructs underlying the mathematical formulae that comprise the robot's "program." It presents raw data as well as statistical summaries from extensive experiments attesting to the physical validity of this new class of algorithms that we call "mirror" laws. Beyond the level of simple visceral pleasure afforded by machine juggling, we believe that the experiments and mathematical reasoning presented here offer the rudiments of a general approach to many other classes of robotic tasks. It seems worth pausing to motivate such claims before proceeding with the subject proper.

[1] This work has been supported in part by PMI Motion Technologies, INMOS Corporation and the National Science Foundation under a Presidential Young Investigator Award held by the second author.

1.1 Geometric Robot Programming

A central theme of this paper (and, indeed, our general program of research in robotics [11, 12, 13]) is the desirability of translating abstract user defined goals into phase space geometry for purposes of task encoding and control. A number of advantages arise from the absence of logic implemented in some more or less formal syntax. First, physical robots and the environments within which they must operate are dynamical systems. Their coupling via functional relationships admits some possibility of correctness proofs (as evidenced below) while the recourse to syntactic prescriptions all but eliminates that hope (for example, see the related discussion in [2]). Second, there is good reason to expect that careful attention to the (provable) geometric invariants of a particular task domain will reveal general properties required of any successful controller. These would need merely be "instantiated" by the appropriate change of coordinates (for example, as in [19]), thereby solving an entire range of problems with one controller structure. Although properly modular software is re-usable, one is hard pressed to imagine a careful study of the code itself revealing which modules are essential. Furthermore, we have consistently experienced less brittle modes of failure and decreased sensitivity to modeling errors in experiments using geometrically expressive control algorithms in comparison to experiments with more syntactically expressive laws. The insensitivity to noise and unexpected perturbations and the strong stability properties of our juggling algorithms are apparent from the experimental data presented in Section 2.3 and 3. Finally, the geometry is intrinsic to the problem and does not commit the controller to a particular computational model. Logical statements, in contrast, are intimately wedded to a discrete symbolic model of computation that best fits a digital computer equipped with a computer language. Yet the contemporary hegemony in information processing of digital computers may represent a brief interlude in the history of technology. Moreover, those roboticists who look to biological systems for inspiration (or who, more radically, treat their robots as plausibility models of biological organization) will surely not be content with the grip of logic and syntax upon their field.

The apparent disadvantage of geometric task encoding relative to syntactic prescriptions is a dramatic reduction in ease of expression. Whether or not the robot's and environment's dynamics will "understand," at least we think *we* know what we mean when we write down if-then-else statements in our favorite programming language. Thus, a central aim in the presentation below is the demonstration that even complicated goals involving some combinatorial component (as does the two-juggle in Section 3.1) may be readily expressed via the appropriate geometric formalism. The intuitively generated extensions to the fundamental mirror algorithm of Section 2.3 described and tested in Section 3.1 and 3.2 have as yet no better claim to analytical origins than any old computer program. But, by the same measure, their generation has been no more arcane than writing code in any new computer language.

We do not seriously expect that all robot tasks at any level can or should be forced into the geometric formalism developed here. However, we feel that this approach is particularly suited to robotics in an intermittent dynamical environment.

1.2 Intermittent Dynamical Environments

There is a large and important range of robotic problems requiring systematic interaction with physical objects governed by independent kinematics and dynamics whose characteristics change subject to the robot's actions. The first systematic work in this task domain has been the pioneering research of Raibert whose careful experimental studies verify the correctness of his elegant control strategies for legged locomotion [18]. McGeer has successfully used local linearized analysis to build passive (unpowered) walking robots [17, 16], and feels that similarly tractable analysis should suffice for controlling running machines as well [15]. Wang [21] has proposed to use the same local techniques for studying *open loop* robot control strategies in intermittent dynamical environments, although his ideas remain to be tested. Research by Atkeson et al. on juggling [1] suggests that task level learning methods may relieve dynamics based (or any other parametric) controller synthesis methods of the need to achieve precise performance requirements as long as a basically functioning system has been assured. Thus, increasing numbers of researchers have begun to explore the problems of robotics in intermittent dynamical environments with increasingly successful results.

Our work is principally inspired by Raibert's success in tapping the natural dynamics of the environment to achieve a task. We have previously shown via analysis similar to that reviewed in this paper [10] that (a greatly simplified version of) Raibert's hopping algorithm [18] is correct. Thus convinced of its value, we have borrowed Raibert's idea of servoing around a mechanical energy level to produce a stable limit cycle, and will demonstrate below that this procedure accounts for the success of the fundamental mirror algorithm as well. Its extension to the problem of juggling two bodies simultaneously may, in turn, have significance with respect to problems of gait in legged locomotion. Presumably, our robot "settles down" to a characteristic steady state juggling pattern because that pattern is an attracting periodic orbit of the closed loop robot-environment dynamics. Very likely, similar "natural" control mechanisms would make good candidates for gait regulation. We have proven only that this presumption is correct for the case of a single puck on our juggling plane. The proof of the two puck case is currently in preparation. Establishing the formal connection to gait mechanisms will obviously require more work.

Further, we believe that the successful control by a one degree of freedom robot of a two and a four degree of freedom intermittent dynamical environment has implications for general robot manipulation of objects in the absence of "guarded moves." Prior to the static grasp phase wherein the myriad robot degrees of freedom may be simultaneously engaged to control a (typically) six degree of freedom object there must be a "fumble" phase — a series of controlled collisions involving unpredictable combinations of the robot link surfaces and the surfaces of the object. During a fumble, far fewer robot degrees of freedom may be engaged with the environment, and only intermittently. We show by experiment below (but have not yet formally proven) that a variation of the mirror algorithm used for juggling results in a quick stable "catch." Moreover, by "juggling" the puck into a specified orbit, a catch may be effected at any portion of the robot's link surface. Mason and colleagues have studied carefully manipulation involving impact with a dynamical environment [14, 20, 23] and have recently begun the study of impacts with intermittent dynamical environments in the absence of sensors as well [9, 22]. There is presumably a clear relationship between these theories: its elucidation would strengthen the applicability of each.

1.3 Organization

This paper is organized as follows. Section 2 summarizes our analytical and experimental results concerning the original mirror algorithm that achieves a simple vertical one-juggle. Much of the material has appeared (or will appear) in other publications to which we will refer where appropriate. The central contributions of this paper are made in Section 3 where we describe work in progress generalizing the original algorithm. We present working mirror-like algorithms for juggling two pucks with a one degree of freedom actuator and for catching objects that are falling freely in the gravitational field.

2 The Mirror Algorithm for a Vertical One-Juggle

This section summarizes work reported elsewhere [4, 6, 5, 7] concerning the fundamental mirror algorithm. Section 2.1 presents our experimental setup and a simplified mathematical model. In Section 2.2 we first introduce the vertical one-juggle task. We then demonstrate that, assuming the availability of perfect state information at each successive impact, it is logically achievable. Sensory information concerning the state of the puck at impact being very difficult to obtain in reality, we will develop in Section 2.3 for purposes of robot implementation a control strategy which uses continuous free flight information about the body. After a summary of analytical results showing that this strategy — the "mirror algorithm" — is correct, we present physical demonstrations of working one-juggles.

2.1 Robot and Environment Models

Here we state the "environmental control problem," which enables us to pose and solve robot juggling tasks as formal problems in control theory. We feel this analysis is central to understanding how robotic tasks in this domain may be planned and effected. For tractability, our analysis is based on the discrete puck states just before *impact*, and therefore examines a map between impact states as a function of discrete robot inputs at those impacts.

2.1.1 Experimental Apparatus

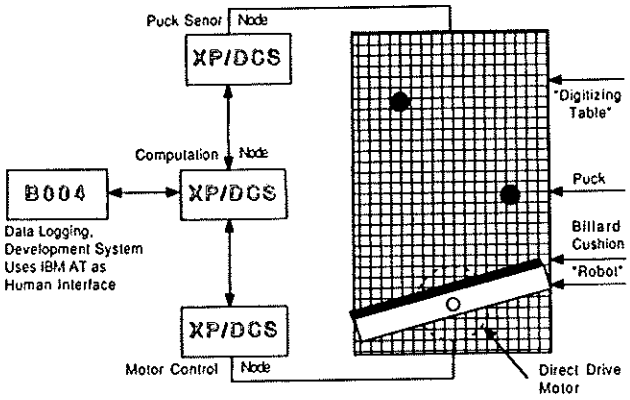


Figure 1: The Yale Juggler

The physical apparatus consists of a puck, which slides on an inclined plane and is batted successively by a simple "robot:" a bar with billiard cushion rotating in the juggling plane as depicted in Figure 1. All intelligent sensor and controller functions are performed by a four node distributed computational network formed from the INMOS transputer based Yale XP/DCS control node [8]. Implementation details describing how the computational resources are mapped to the mechanical hardware can be found in [6, 5].

2.1.2 Mathematical Model

We now motivate and present without further derivation a simplified model of our apparatus which abstracts away as much of the extraneous physics as is possible while retaining the essential aspects of a robot and environment whose dynamics are intermittently coupled. For a more complete discussion and formal derivation of this model we refer to [5]. The vindication of our claim to have retained the essential aspects, of course, is only possible by recourse to physical experiment, and will be provided in Section 2.3.3.

A puck trajectory with a puck-robot collision is depicted in Figure 2. We will use this figure to informally introduce the notation used in the model and the remainder of the paper.

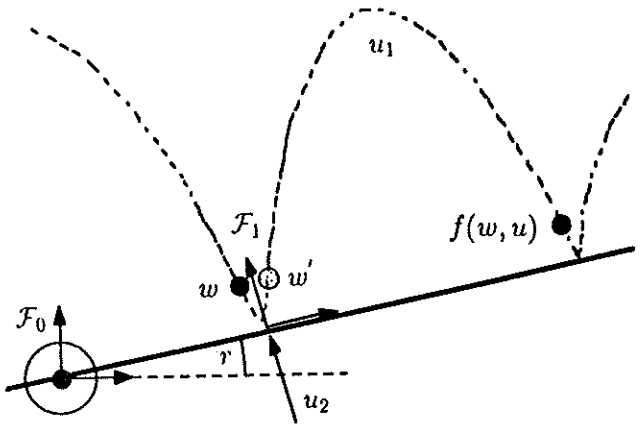


Figure 2: The Impact Event

The configuration space of the entire problem is the cross product, $\mathcal{C} \triangleq \mathcal{B} \times \mathcal{R}$, of the body and the robot configurations. We will model the robot's configuration space as $\mathcal{R} \triangleq [-\pi/2, \pi/2] \subset \mathbb{R}$, which we restrict to a half revolution, since for present purposes, it will suffice to consider only those locations of the bar in the right half of the juggling plane for which the hitting billiard cushion is facing up. We will represent the location of the falling body on the plane $\mathcal{B} = \mathbb{R}^2$ with the coordinates $b \triangleq [b_1 \ b_2]^T$ denoting, respectively, the position of its centroid relative to the horizontal and vertical axes of the reference frame, $\mathcal{F}_0$.

In isolation, the robot's dynamics occur in its phase space, $\mathcal{V} \triangleq \mathcal{R} \times \mathbb{R}$, of angular positions and velocities, and may be modeled by the standard double integrator forced by commanded torque. In isolation, the puck's dynamics occur in its phase space, $\mathcal{W} \triangleq \mathcal{B} \times \mathbb{R}^2$, and may be modeled simply by the equations of free flight in the earth's constant gravitational field. In reality, there is noticeable coulomb friction on the sliding plane, and we have found it interesting to compare numerical simulations of the robot control laws in the idealized frictionless environment with the same strategies run in the more realistic simulation model with friction, as against empirical data. The coupling of robot and puck dynamics is represented by a collision map, c, that takes the à priori puck-robot states at contact, $\left(v = (r, \dot{r}) \in \mathcal{V}, w = (b, \dot{b}) \in \mathcal{W}\right)$, into the new puck velocity vector after contact,

$$\dot{b}' = \mathbf{c}\left((b, r), (\dot{b}, \dot{r})\right),$$

making the simplifying assumption that the nature of energy loss and momentum exchange can be captured via the coefficient of restitution, α. This model is derived carefully in [4, 6, 5].

The future trajectory of the body in $\mathcal{W}$ subsequent to an impact event may now be readily derived (as a function of the puck-robot states just before impact, w, the robot velocity at impact, $\dot{r}$, and the time of flight, t) by integrating the free flight

(i) $w^* \in \mathcal{T}$, *the vertical one-juggle task set, where*

$$\mathcal{T} \triangleq \left\{ w \in \mathcal{W} : b_2 = 0,\ \dot{b}_1 = 0 \right\}.$$

(ii) $g(w^*) = u^* \triangleq \begin{bmatrix} -2/\gamma \\ -\frac{1-\alpha}{1+\alpha} \end{bmatrix} \dot{b}_2^*$.

Notice that $\mathcal{T}$ corresponds exactly to those constant set points which our intuitive thinking leads us to understand would cause a purely vertical periodic puck trajectory which returns to the same apex point again and again. Thus we call any set point $w^* \in \mathcal{T}$ a *vertical one-juggle task* .

But it is not enough to merely achieve a vertical one-juggle fixed point. We must insure as well that perturbations away from the desired behavior are dissipated. We now ask which vertical one-juggle task points, $w^* \in \mathcal{T}$, can be made asymptotically stable fixed points of (3) by an appropriate choice of g. The following result shows the answer to be: all points in the vertical one-juggle task set may be stabilized.

Proposition 2.2 ([5]) *If*

$$w^* \in \mathcal{T}$$

and g fixes w^*, $f_g(w^*) = w^*$. (3) then system (2) is locally controllable at $(w^*, g(w^*))$.

Since the local linearized system around any fixed point $w^* \in \mathcal{T}$ is controllable, it follows from linear control theory, that for any desired set of poles whose complex elements appear in conjugate pairs,

$$\Lambda = \{\lambda_i\}_{i=1}^n \subset \mathbb{C},$$

there exists a matrix, $K_\Lambda \in \mathbb{R}^{2 \times n}$ such that the linear affine state feedback law,

$$g(w) \triangleq u^* + K_\Lambda (w - w^*) \tag{4}$$

places the poles of the linearized closed loop system at Λ. Thus we have shown that the vertical one-juggle is logically achievable.

2.2.3 Missing: A Solution to the Robot Control Problem

This discrete analysis confirms the intuition that only state information at impact should be required for a successful juggling algorithm — that full trajectory information is redundant. Conceptual appeal notwithstanding, in reality state information at or very near the impact event is exceedingly difficult to measure. Moreover, we have said nothing yet concerning the ability of the robot to realize any particular feedback strategy, g, much less one which stabilizes a desired set point, w^*: it is completely up to the designer to solve the robot control problem and achieve an approximation to the required impact strategy. Finally, employing affine feedback (4) guarantees only local asymptotic stability: it is not clear how to enlarge the domain of attraction around the fixed point sufficient for the appropriate equilibrium state to be observed in the physical world. In a previous paper [7] we have reported our failure to achieve experimental success with any implementation of a locally stabilizing feedback strategy (4), essentially for these reasons.

Evidently, missing still from our physical model of the task itself is an account of how the robot dynamics should be coupled to the environment's reaction. The environmental control problem ignores the robot control problem and we must seek some further algorithmic procedure for generating a robot controller capable of inducing effective abstract feedback strategies, g.

2.3 The Mirror Algorithm

We now introduce the mirror algorithm which meets the desiderata of the previous paragraph. In Section 2.3.1 we provide an intuitive motivation for the mirror law. We review our analytical results concerning this algorithm in Section 2.3.2. Finally, the last section attests to the empirical relevance of our findings.

2.3.1 Intuitive Motivation

Intuitively, two different ideas are combined to produce an algorithm that is implemented by recourse to standard trajectory tracking techniques. First, we "reflect" the continuous puck trajectory in $w(t)$ into a "distorted mirror image" reference trajectory for the robot r which is "favorable" to the task at hand. In the specific case of our planar juggler, as depicted in Figure 1, the robot is forced to track the distorted "puck angle," $\theta = atan\frac{b_2}{b_1}$, to control the puck's vertical motion, modulo a PD feedback term for stabilization around a fixed desired horizontal position, $(b_1^*, \dot{b}_1^* = 0)$,

$$r = -\mu(w) \triangleq -\kappa_1(w)\theta + \kappa_{21}(b_1 - b_1^*) + \kappa_{22}\dot{b}_1. \tag{5}$$

Both κ_{21} and κ_{22} are fixed constant gains.

Second, in order to stabilize the vertical periodic motion we borrow from Raibert [18, 3] the idea of modifying the robot's trajectory by "servoing" on the discrepancy between the (constant) total mechanical energy of the puck in its desired steady state, $\eta(b_2^*, \dot{b}_2^*)$, and the (constant during flight, as we neglect friction) currently measured value $\eta(b_2, \dot{b}_2)$:

$$\kappa_1(w, w^*) \triangleq \kappa_{10} + \kappa_{11}[\eta(w^*) - \eta(w)]$$

Here κ_{10} is determined by the fixed point condition, and κ_{11} is again a fixed constant gain. A more complete intuitive account of this mirror algorithm is provided in [5, 6].

2.3.2 Formal Results

In this section we will present a summary of analytical results that are taken from the complete presentation in [5]. When the robot has achieved the reference "mirror" trajectory described above then the puck and robot trajectories lie on a "mirror surface," $\mathcal{M} \subseteq \mathcal{W} \times \mathcal{R}$, in the cross product phase space,

$$\mathcal{M} \triangleq \left\{ \left((b, \dot{b}), (r, \dot{r}) \right) \in \mathcal{W} \times \mathcal{V} : (r, \dot{r}) = \left(\mu(b, \dot{b}), \dot{\mu}(b, \dot{b}) \right) \right\}.$$

Formally, this is specified as the graph of the function $\mu(w)$ (5) and its derivative along the motion of the puck.

Examination of the intersection between $\mathcal{M}$ and the velocities over the contact set, where $\theta = r$, reveals how to choose the gain κ_{10} in (5) to achieve the fixed point conditions of Proposition 2.1 for any $w^* \in \mathcal{T}$. A central result, [5, Proposition 5.2], shows via projection of this intersection onto $\mathcal{W}$, that the robot's "mirroring" motion induces a three dimensional invariant submanifold of the environmental control system (2). In consequence, [5, Corollary 5.3], the local stability behavior of any valid vertical one-juggle task, $w^* \in \mathcal{T}$, may be adjusted by the appropriate choice of gains in (5).

Although this result obtained from the linearized system provides a formal proof of correctnes of the mirror algorithm, it does not furnish a characterization of the domain of attraction. Such information is very important, as local stability by itself does not guarantee a successful practical implementation without a sufficiently large basin of attraction. While a nonlinear analysis of the planar juggler operating under the mirror algorithm is presently incomplete, we have achieved the analogous result for a simplified version of our juggler: a one degree of freedom robot operating in a one degree of freedom environment. These results provide us with more insight into the qualitative behavior of the higher dimensional system, and suggest how to achieve the nonlinear analysis.

2.3.3 Empirical Verification

We now present a direct application of the mirror algorithm for the vertical one-juggle implemented by the revolute robot in a two degree of freedom environment. We will compare experimental results obtained using the apparatus described in Section 2.1.1 with computer simulations of the models developed in Section 2.1.2. First, we review the manner in which the theoretical understanding developed in Section 2.3.2 informs our selection of gains.

The horizontal impact position and the apex point (via the vertical impact velocity) of the desired vertical one-juggle are determined by the selection of an appropriate fixed point. In the data displayed below, we have chosen

$$w^* = \begin{bmatrix} b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix} = \begin{bmatrix} 11 \text{ inches} \\ 0 \\ 0 \\ -125 \text{ inches/sec} \end{bmatrix}.$$

The analytical results guarantee that for this fixed point, the vertical one-juggle task is achievable. Exact tracking of the mirror function (5) induces an environmental feedback strategy, whose closed loop dynamics defines a three dimensional invariant submanifold in the four dimensional phase space of the puck, $\mathcal{W}$, that contains the desired fixed point, w^*. The analysis also reveals how to choose the specific gain settings in order to keep all poles inside the unit circle. We have chosen the following gain settings,

$$\kappa_{11} = 30 \cdot 10^{-6}; \quad \kappa_{21} = 5 \cdot 10^{-3}; \quad \kappa_{22} = \kappa_{21} = 7 \cdot 10^{-3},$$

in the data displayed below. The coefficient of restitution, α, was experimentally found to be 0.7. The resulting linearized system has the poles

$$\hat{\Lambda} \triangleq \{-0.35, \ 0.21, \ 0.18\}.$$

Since these lie within the unit circle of $\mathbb{C}$, w^* is an asymptotically stable fixed point.

Figure 3, a "recording" of a successful vertical one-juggle nicely depicts the rapid convergence for initial conditions (in drop-off position) from a large region within the puck's workspace. Since the global analysis is still incomplete, we have not tested the exact extent of the domain of attraction. It is likely, as in the one degree of freedom case that w^* is not, in general, globally asymptotically stable. Notice, even at steady state, that the impacts do not occur at zero height, but rather at a fixed offset: this is an artifact of the non-zero puck radius and the robot dimensions ignored in the simplified model. Despite these departures from the idealized model and the relatively large sensor noise it may be observed from this and the subsequent plots that our algorithm produces steady reliable juggling performance. We have recorded vertical one-juggle runs with hundreds of impacts without encountering any failures.

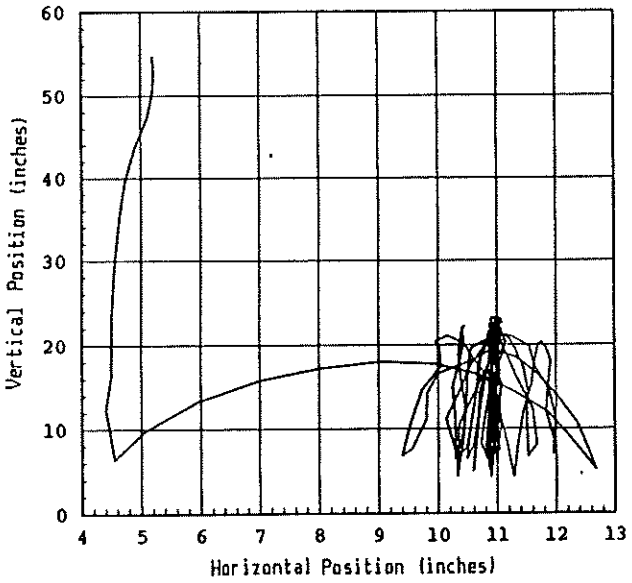
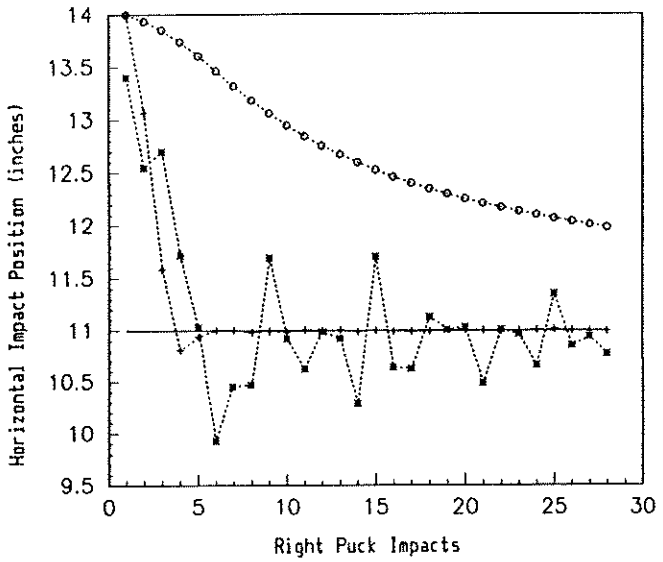


Figure 3: Sample continuous data

Figures 4 and 5 compare the responses of the analytical model with and without friction with the responses of our experimental setup for two different initial conditions. The steady state values in the horizontal position, b_1, are very close around the desired value for all three curves. The plots of the vertical impact velocity, $\dot{b}_2$, exhibits that, first, as we expected the effect of the unmodeled friction is a steady state deviation, which second, is rather accurately predicted by the model augmented with friction. Examining the transients, notice that the experimental transient responses for $\dot{b}_2$ (lower plots) consistently match the responses of the model with friction, as expected. However, for b_1 (upper plots) the experimental transient responses are closer to the much faster transient model responses *without* friction than to those of the model with friction. This latter positive effect is not completely understood at present. We suspect that the unmodeled effect of spin on the impact might be responsible for this benign discrepancy.



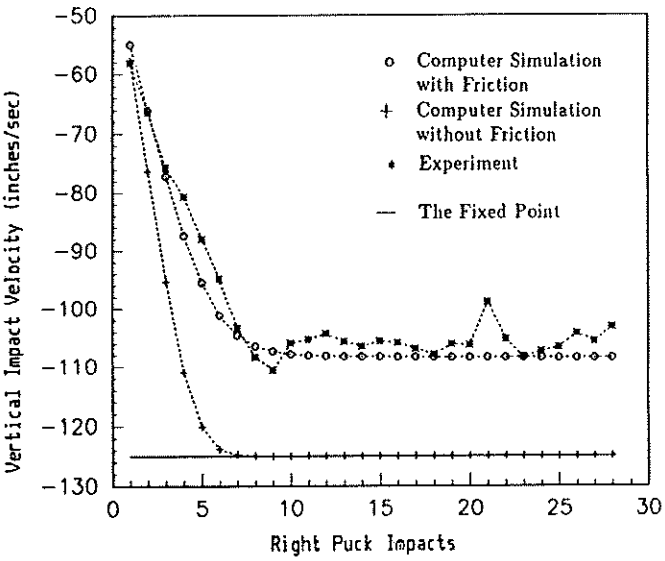
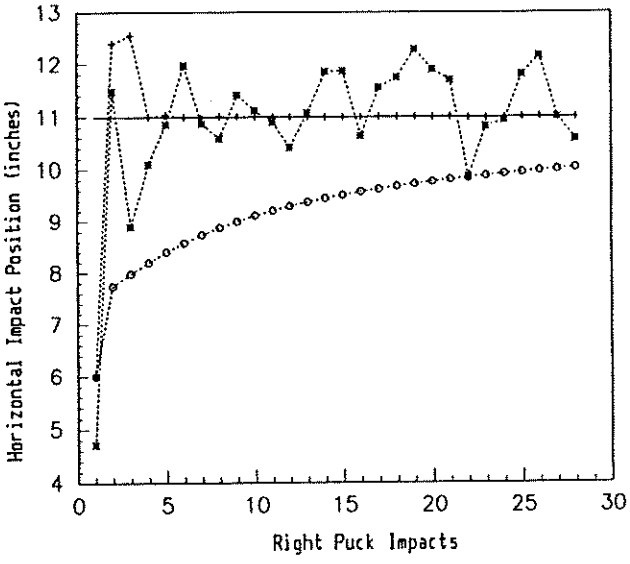


Figure 4: Initial condition: "Right Down"



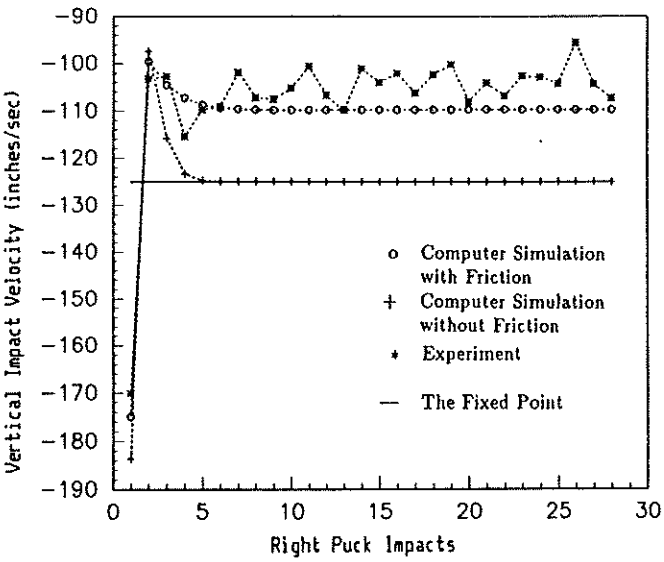


Figure 5: Initial condition: "Left Up"

3 Applications

We now present two extensions of the vertical one-juggle mirror algorithm described in the previous section. We pose two new tasks. The vertical two-juggle task discussed in Section 3.1 introduces new aspects of timing and combinatorial choice to the problems of intermittent dynamical environments that we feel are closely related to issues of gait regulation in legged locomotion. The catch task discussed in Section 3.2 has significance for the "fumbling" stage of robot manipulation. Beyond the intrinsic interest of both extensions, their presentation affords the opportunity for a more detailed examination of the geometric programming methodology implicit in the original mirror law as described in the introduction.

3.1 The Two-Juggle

Our first — and most obvious — generalization of the one-juggle is the "two-juggle:" the task of simultaneously keeping two pucks on either side of the juggling plane in a specified periodic orbit by repeated impacts. A successful two-juggle algorithm must intermittently control the horizontal impact positions and vertical impact velocities of the two pucks, their phase angle separation and, in addition, must resolve the one degree of freedom robot's kinematic restrictions. In this section we will describe how the one-juggle mirror algorithm can be extended in a straightforward fashion to achieve the two-juggle. The resulting algorithm maintains the pure geometric features of the original mirror law in that it maps the continu-

ous phase space trajectory of both pucks simultaneously into a reference trajectory in the robot's phase space. There is no explicit use of time, and there is no logical syntax. The extended algorithm experimentally exhibits global convergence and robustness properties in its four degree of freedom case similar to those of the original mirror algorithm in the two degree of freedom case. We suspect that a very similar analysis will go through as well but have not yet attempted a rigorous stability investigation. Moreover, as stated in the introduction, we anticipate that some of the insights developed here will generalize to the problems of active gait stabilization in legged locomotion. We have not yet pursued these connections.

3.1.1 Extending the Mirror Law

The two-juggle task can be loosely described as follows: Perform two one-juggles, w_l and w_r, one on each side of the juggling plane with

$$w_l^* = \begin{bmatrix} -b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix}, \quad w_r^* = \begin{bmatrix} b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix},$$

while maintaining a specified time between impacts (phase angle relationship) as well. For simplicity, we shall specify for this initial implementation that alternative impacts occur maximally separated in time. There is a provably correct and empirically verified mirror law, $\mu_l \triangleq \mu(w_l), \mu_r \triangleq \mu(w_r)$, (5) for each puck. An obvious approach to the two-juggle problem is to prescribe a weighting rule by means of which the robot can decide which mirror law is more critical at any time. Two distinct issues arise in the determination of a viable weighting rule: what to do in an emergency situation when both pucks are roughly equally needy of attention at once; and what to do when the pucks are reasonably well separated in phase angle in order to keep them away from the emergency situation. We loosely refer to the first as the global phase angle problem and the second as the local phase angle problem.

The Local Phase Angle Control Problem We can readily identify a purely geometric measure of either puck's phase angle, ϵ, that indicates how much of the total trajectory has been traversed, and does not depend on other state variables, like horizontal position, absolute height, or vertical energy $\eta(w) = \frac{1}{2}\dot{b}_2^2 + \gamma b_2$. For

$$\epsilon \triangleq -\frac{\dot{b}_2}{\sqrt{2\eta}},$$

any puck trajectory maps onto the interval $(-1, 1)$ between impacts (assuming they occur close to zero height), and evaluates to zero at the apex. A geometric version of the *phase angle error* built from this function,

$$e_{ph} = [\epsilon(w_l) - \epsilon(w_r)]^2 - 1,$$

vanishes when, for example, one puck is at apex and the other is very near (either just before or after) an impact. The farther away the two pucks are from the desired phase angle separation, the larger the phase angle error grows.

When the pucks are well separated in phase angle then their conflicting mirror laws may be relatively easily satisfied one at a time. Thus, it makes intuitive sense to allow the robot to track the original mirror algorithm only when e_{ph} is close to zero, and to force the phase angle error back toward zero even at the temporary expense of the accuracy of either puck's vertical one-juggle. This intuition is most simply expressed by adding the phase angle error term as a perturbation on the original mirror law (5) as follows,

$$k_{1,r}(w_l, w_r, w^*) = k_{10} + k_{11}[\eta(w^*) - \eta(w_r)] + k_{12}e_{ph}(w_l, w_r),$$

while leaving the remainder of (5) unchanged.

The Global Phase Angle Control Problem In an emergency situation, when both pucks are falling toward the bar nearly at the same time, it is imperative to service the nearest first yet sacrifice any needed one-juggle performance measures to the work of restoring phase angle separation. This intuition can be readily implemented by mediating between the the two reference trajectories from the two puck's independent one-juggle algorithm via a "weighting function",

$$s = \frac{\sigma(w_l)}{\sigma(w_l) + \sigma(w_r)}, \quad \sigma(w) = \sigma_1(w)\sigma_2(w),$$

$$\sigma_1(w) = \frac{1}{2} - \frac{1}{\pi}atan[k_{12}(b_2 - b_{2,0})],$$

$$\sigma_2(w) = \frac{1}{2} + \frac{1}{\pi}atan[k_{13}(-\dot{b}_2)].$$

The function σ_1 constitutes the heart of the global phase angle separation. With $b_{2,0}$ and k_{12} set properly it evaluates to one in the vicinity the robot and decreases to zero further away. Using a simple vertical distance works out well due to the simplifying fact that impacts occur close to $b_2^* = 0$, even when the puck motion is away from the fixed point. The function σ_2 scales the previous one in a smooth fashion such that weight is only assigned during the the descending part of the puck trajectory. The function s then simply normalizes the contributions of σ_l and σ_r.

The Extended Mirror Law The mirror law extension for the two-juggle may now be written as

$$\mu_{2jug} = s\,\mu(w_l) + (1 - s)\,\mu(w_r).$$

In the case of good phase angle separation, each puck gets "full attention" from the robot close to impact. If the phase angle separation is not good, both pucks can be close to impact simultaneously, and thus both $\sigma(w_l)$ and $\sigma(w_r)$ can approach one. In this case both mirror algorithms $\mu(w_l)$ and $\mu(w_r)$ will contribute to the robot reference trajectory in such a fashion that the puck closer to impact receives more weight. If both pucks fall identically in phase angle at once (that is, if they possessed the exact same trajectories, except for a sign reversal in b_1), μ_{2jug} evaluates to zero and the robot does not move at all. Both pucks eventually come to rest. This failure mode is very rarely observed in practice.

3.1.2 Discussion of Experimental Data

We implemented the two-juggle algorithm as described above on our planar juggling apparatus. To provide better qualitative

insight into its convergence and robustness properties, we display in Figure 6 the vertical positions, b_2 of the two pucks versus time (for simplicity, not showing the horizontal positions, b_1). They are dropped simultaneously from slightly different initial heights. This results not only in initial errors in vertical impact velocity (and thus height), but also, as can be clearly seen at the first impact, in very close impacts (large initial phase angle error). Subsequent impacts display the recovery from both the error in phase angle as well as impact velocity. After reaching steady state, the motion is drastically perturbed again. The arrow in the plot marks the instance where the ascending puck was knocked away by hand before its apex. The resulting large velocity (height) error for that puck and the phase angle error (very close impacts) are obvious from the data. Again, steady state is recovered within roughly five impacts per puck.

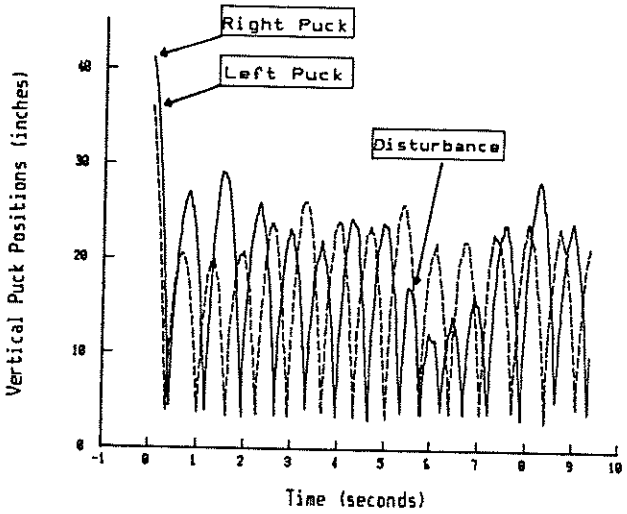


Figure 6: Continuous vertical positions of the two-juggle

Next, we present data plots of the puck states just before impact. These figures display statistical information (mean and standard deviations, or min/max) obtained from a number of successive runs (without handpicking), while also giving the number of failures encountered. We feel that this presentation offers a closer rendering of true performance than one based upon a handpicked best run. Such presentation methods are standard practice in other natural sciences, but seem only slowly to be entering the field of robotics [17].

The following plots of impact states show the simultaneous convergence properties in all three initial errors in this fashion. Both pucks were dropped by hand simultaneously from their initial positions ($b_1 = -14$ inches, $b_2 = 33$ inches and $b_1 = 14$ inches, $b_2 = 38$ inches for the left and right puck respectively). During the following one hundred impacts (fifty per puck) each puck's horizontal impact position, b_1, the vertical impact velocity, $\dot{b}_2$, and the pase error e_{ph} was recorded. This was repeated for 35 runs. Two runs were discarded because they failed during the first five impacts. All other runs completed successfully. From the first 30 completed runs we show the mean with error bars indicating one standard deviation in both directions. In Figure 7 we see the convergence to steady state within five impact for the initial transients in horizontal impact positions. The 1 inch offset for the left puck is readily explained as a consequence of inaccurately modeling the steady state vertical impact position which depends on the amount the elastic billard cushion is compressed. Modeling errors translate into shifts in horizontal impact positions. Following, Figure 8 shows both puck's convergence to the value predicted by the model taking friction into account, in a similar fashion to Figure 4 or 5. Finally, Figure 9 shows the simultaneous rapid elimination of the phase angle error between the two pucks. In all the plots, the standard deviations, after some increase during transients, stay fairly constant.

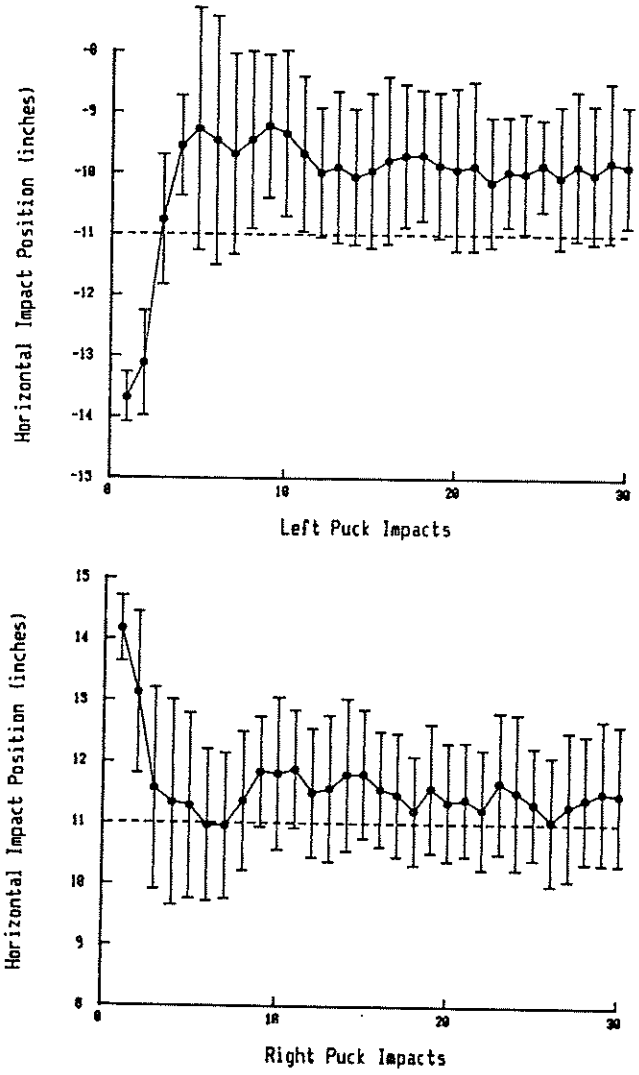


Figure 7: Horizontal position impact data of the two-juggle

In addition to the displayed convergence properties this algorithm has further desirable attributes. It runs consistently for hundreds and hundreds of impacts. Notice (again, in consequence of the geometric nature of the algorithm) at no point in time does it depend on a single measurement, like flight time, impact position, impact phase angle error, apex positions, etc. This means that the system is more tolerant of noise and measurment errors. Furthermore, during the two-juggle, one or even two pucks can be manually halted and released without causing the system to fail. When the first puck is halted, the second one continues its motion; as soon as the first one is released again, the two-juggle continues. All this

is accomplished by no other means than the relatively simple, smooth algorithm, μ_{2jug}, described above. There is no additional higher level decision making or conditional branching as found in implementations purely characterized by a computer program. Therefore, not only simple and robust, this approach lends itself, in a similar fashion as does the one-juggle [5], to rigorous analysis.

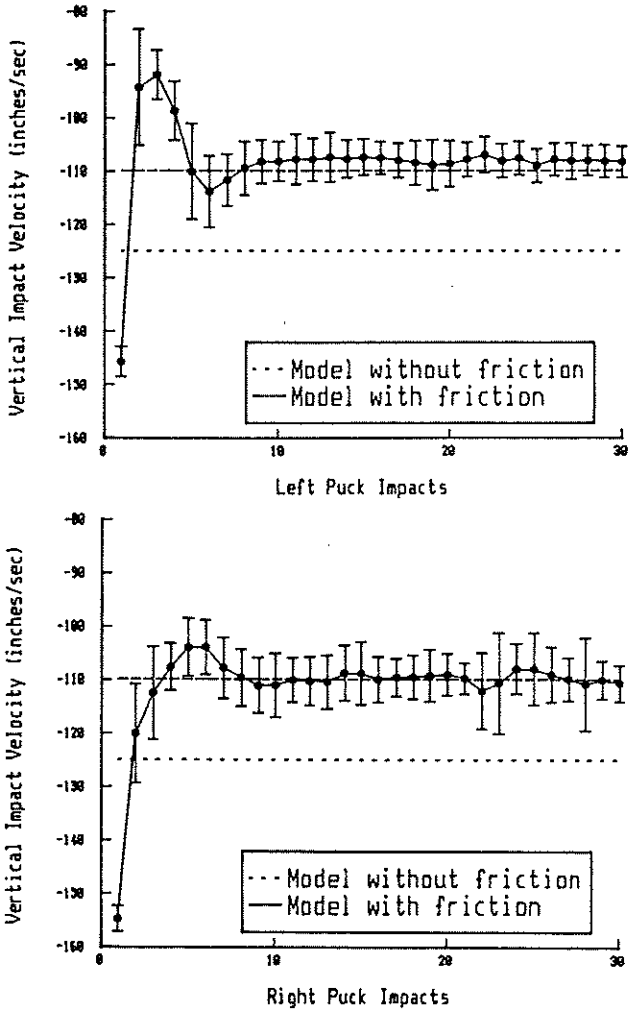


Figure 8: Vertical impact velocity data of the two-juggle

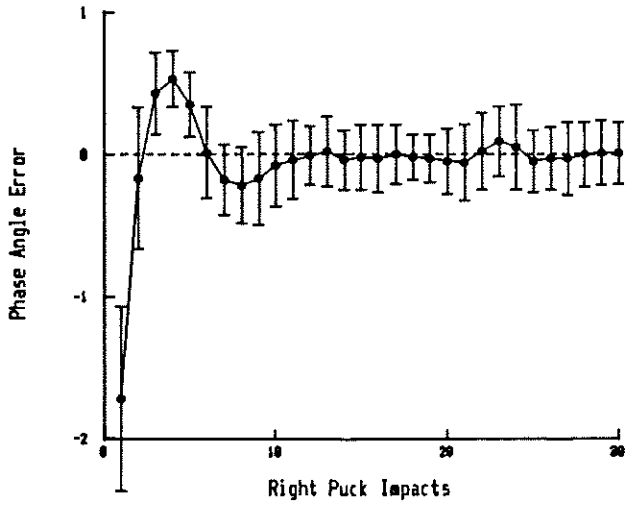


Figure 9: Two-juggle phase angle error

3.1.3 Alternative Extensions

There are many different approaches to the two-juggle which we could have taken. In fact, the first working implementation of the two-juggle servoed around the error in flight time. This error term entered the one-juggle mirror algorithm in the same fashion as the phase angle error term described above. The global phase angle separation was accomplished by simply *switching* s between one and zero at each impact. While this resulted in a working implementation that produced good steady state regulation, it suffered from a number of shortcomings which seem to be typical of algorithms that take time explicitly into account. It had poor global angle phase performance: emergency situations almost always caused a failure. Moreover, the time based two-juggle is much more brittle: it cannot tolerate the removal of one or the other of the pucks during a run as can the algorithm sketched above. It relies strictly on the expectation that the two pucks will impact alternatively. Finally, while the purely geometric extension of the mirror law does not rely on any single event measure, in the time based version we must accurately detect the impact events in order to determine the time between impacts. In the presence of noise and measurement errors this leads to a more difficult and less robust implementation. The scheme of switching the mirror laws at impacts worked well at steady state, with both pucks' phase angle well separated. It failed with perturbations of magnitudes as displayed in Figures 6 to 8. The time between the two close impacts is much too short for the robot to track the reference trajectory sufficiently well to maintain the two-juggle.

3.2 Catching

In this section we continue our informal application and extension of the theory of Section 2 to a rather different task domain — "catching" — which we feel sets our research off in the direction of dynamical manipulation through the investigation of fumbles described in the introduction.

Define a catch to be characterized by steady state points in the "catch set,"

$$\mathcal{A} \triangleq \left\{ \left((b, \dot{b}), (r, \dot{r}) \right) : b_1 = r, b_2 = 0, \dot{b}_1 = \dot{b}_2 = \dot{r} = 0 \right\}, \quad (6)$$

the points in puck-robot phase space with the same position and zero velocity. It would be appealing to treat a catch as a particular instance of a vertical one-juggle whose apex point happens to be at zero height. There are, however, two distinct problems with this point of view. Operationally, the theory as set out in Section 2.3.2 may fail since $\mathcal{A}$ is contained in the degenerate set $\mathcal{T}$ whereon (2) is not completely controllable according to Proposition 2.2. Moreover, the vertical one-juggle task is defined in terms of asymptotic convergence to a fixed point. Yet in the present setting an asymptotic catch (at least to an unspecified horizontal position on the bar) obtains trivially from any initial condition (by the strategy of commanding the robot gripper remain stationary) since the coefficient of restitution is less than one. We are led to distinguish asymptotic convergence to $\mathcal{A}$ from convergence in finite time: borrowing from discrete time linear control theory, let us call the latter a "deadbeat catch."

We now present in an informal fashion a variation on the theme of the mirror algorithm which produces a deadbeat catch in one step. We will display below experimental data attesting to its efficacy (in the two degree of freedom environment) and defer to a future paper for its theoretical justification. For ease of exposition, consider first a one degree of freedom robot in a one degree of freedom environment, where r and b represent, respectively, the robot's and puck's vertical position. Consider the variant mirror law,

$$\mu_{catch}(w) = \kappa_{10} atan\left(\kappa_{11}\theta(b)\right). \qquad (7)$$

Notice, first, that contact between robot and puck will occur if and only if $\theta(b) = b = r = 0$. Taking the derivative of (7) and evaluating at $b = 0$ results in

$$\dot{r}\,|_{b=r=0} = \kappa_{10}\kappa_{11}\dot{b}.$$

By setting $\kappa_{10}\kappa_{11} = 1$, we not only have satisfied the catch conditions, but also accomplish them at $r = b = 0$. The extension to our physical revolute robot in the two degree of freedom environment is readily accomplished: now r denotes the robot *angle* and θ, the body angle as defined previously. With this change of notation, the same insights apply as above.

As mentioned above, a deadbeat catch is possible in the two degree of freedom environment only to the horizontal position on the robot's bar corresponding to that of an initial drop state with zero horizontal velocity. With a mind toward eventual applications to general robot manipulation, we now point out that the two task capabilities — the vertical one-juggle and the deadbeat catch — offer a general means of rapidly transferring the body from one rest position (or, for that matter, any initial state on the juggling plane) to any other rest position on the robot's bar. Namely, given a desired steady state in $\mathcal{A}$, one commands a vertical one-juggle to any point in $\mathcal{T}$ whose horizontal component matches that of the desired catch point. After the puck settles down to its steady state trajectory corresponding to the task point in $\mathcal{T}$ (theoretically after an infinite amount of time, in practice, after four or five impacts), one commands a deadbeat catch.

We will now present the measured data from an implementation of the deadbeat catch algorithm for our physical revolute robot in the two degree of freedom environment. Figure 10 shows the continuous vertical puck position data starting from the dropping condition. The horizontal components we presume to be correct at the initial conditions, thus, the deadbeat catch could be implemented at the very first impact. Instead, to underscore the comments of the previous paragraph, we first use the mirror algorithm (5) to achieve a good vertical one-juggle above the desired catch point in $\mathcal{A}$. Following the fifth impact, the control law is switched to the variant catch algorithm (6), and the puck is "caught" at the correct horizontal position on the robot's bar.

A final comment concerning data interpretation is now in order. Although it is not clear from the plot (for the reasons listed in [5] concerning measurement noise) we emphasize that the puck remains in contact with the robot bar after the catch. The subsequent jitter in the figure is not a result of repeated impacts, but a consequence of the motion of the robot bar. This motion, generally not energetic enough to cause visually verifiable loss of contact, is partially the result of improperly tuned constants in (6), and more likely a consequence of the difficulty we have in sensing puck events in the vicinity of the robot bar (as explained above). A theoretically informed choice of μ_{catch} would presumably enable us shape the local transients more effectively, but the robot cannot match the ball velocity *exactly* anyway, and will not come to rest immediately no matter how carefully (6) is adjusted. In a real application, one might imagine adding a dead zone or some other more sophisticated open loop strategy for taming the final phase of the catch.

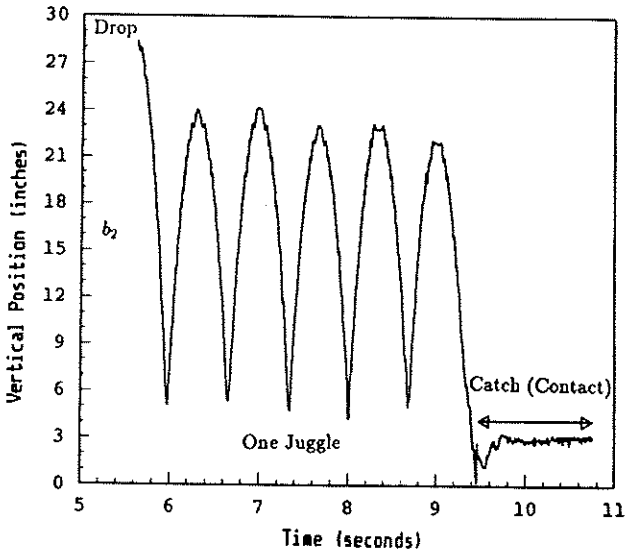


Figure 10: Continuous vertical positions of the catch

References

[1] E. Aboaf, S. Drucker, and C. Atkeson. Task-level robot learning: juggling a tennis ball more accurately. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1290–1295, Scottsdale, AZ, May 1989.

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

[3] M. Bühler and D. E. Koditschek. Analysis of a simplified hopping robot. In *IEEE International Conference on Robotics and Automation*, pages 817–819, Philadelphia, PA, Apr 1988.

[4] M. Bühler and D. E. Koditschek. A prelude to juggling. In *26th IEEE Conference on Decision and Control*, pages (paper available from authors — not in proceedings), Los Angeles, CA, Dec 1987.

[5] M. Bühler, D. E. Koditschek, and P. J. Kindlmann. A Simple Juggling Robot: Theory and Experimentation. In V. Hayward and O. Khatib, editors, *International Symposium on Experimental Robots*, page (to appear), Springer-Verlag, Montréal, Canada, 1989.

[6] M. Bühler, D. E. Koditschek, and P. J. Kindlmann. A family of robot control strategies for intermittent dynamical environments. In *IEEE International Conference on Robotics and Automation*, pages 1296–1301, Scottsdale, AZ, May 1989.

[7] M. Bühler, D. E. Koditschek, and P. J. Kindlmann. A one degree of freedom juggler in a two degree of freedom environment. In *Proc. IEEE Conference on Intelligent Systems and Robots*, pages 91–97, Tokyo, Japan, Oct 1988.

[8] M. Bühler, L. Whitcomb, F. Levin, and D. E. Koditschek. A distributed message passing computational and i/o engine for real-time motion control. In *Proc. American Control Conference*, pages 478–483, Pittsburgh, PA, Jun 1989.

[9] Michael Erdmann and Matthew T. Mason. An exploration of sensorless manipulation. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1569–1574, IEEE, San Francisco, CA, Apr 1986.

[10] D. E. Koditschek and M. Bühler. Analysis of a simplified hopping robot. *The International Journal of Robotics Research*, (to appear).

[11] Daniel E. Koditschek. Automatic planning and control of robot natural motion via feedback. In Kumpati S. Narendra, editor, *Adaptive and Learning Systems: Theory and Applications*, pages 389–402, Plenum, 1986.

[12] Daniel E. Koditschek. Exact robot navigation by means of potential functions: some topological considerations. In *IEEE International Conference on Robotics and Automation*, pages 1–6, Raleigh, NC, Mar 1987.

[13] Daniel E. Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, (to appear).

[14] Matthew T. Mason. Mechanics and planning of manipulator pushing operations. *International Journal of Robotics Research*, 5(3):53–71, Fall 1986.

[15] T. McGeer. *Passive Bipedal Running.* Technical Report IS-TR-89-02, Simon Fraser University, Centre for Systems Science, Apr 1989.

[16] T. McGeer. Powered flight, child's play, silly wheels and walking machines. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1592–1597, May 1989.

[17] T. McGeer. *Stability and Control of Two-Dimensional Biped Walking.* Technical Report IS-TR-88-01, Simon Fraser University, Centre for Systems Science, Sep 1988.

[18] Marc H. Raibert. *Legged Robots That Balance.* MIT Press, Cambridge, MA, 1986.

[19] E. Rimon and D. E. Koditschek. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. (under review) *Transactions of the American Mathematical Society*, 1989.

[20] Russell H. Taylor, Matthew T. Mason, and Kenneth Y. Goldberg. Sensor-based manipulation planning as a game with nature. In *Fourth International Symposium on Robotics Research*, MIT Press, 1987.

[21] Yu Wang. *Dynamic Analysis and Simulation of Mechanical Systems with Intermittent Constraints.* PhD thesis, Carnegie-Mellon University, 1989.

[22] Yu Wang. Mechanics and planning of collisions in robotic manipulation. In *Proc. IEEE International Conference on Robotics and Automation*, pages 478–483, Scottsdale, AZ, May 1989.

[23] Yu Wang and Matthew T. Mason. Modeling impact dynamics for robotics operations. In *Proc. IEEE International Conference on Robotics and Automation*, pages 678–685, San Francisco, CA, Apr 1987.