Planning and Control of Robotic Juggling and Catching Tasks

M. Buehler
Center for Intelligent Machines
Mechanical Engineering Department
McGill University
Montréal, Québec, Canada H3A 2A7
E-mail: buehler@cim.mcgill.edu

D. E. Koditschek
Electrical Engineering and Computer Science Department
University of Michigan
Ann Arbor, Michigan 48109

P. J. Kindlmann
Center for Systems Science
Department of Electrical Engineering
Yale University
New Haven, Connecticut 06520-1968

Abstract

A new class of control algorithms—the "mirror algorithms"—gives rise to experimentally observed juggling and catching behavior in a planar robotic mechanism. The simplest of these algorithms (on which all the others are founded) is provably correct with respect to a simplified model of the robot and its environment. This article briefly reviews the physical setup and underlying mathematical theory. It discusses two significant extensions of the fundamental algorithm to juggling two objects and catching. We provide data from successful empirical verifications of these control strategies and briefly speculate on 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 distinct circumstances. When in the middle of juggling two pucks, if suddenly one puck is fixed and held in place, the robot will continue to juggle the other. When the first puck is again released, the robot will adjust its hits to restore symmetry between the two pucks' motions. 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 on-line 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, an approach we have come to call "geometric robot programming."[1] It succeeds over a wide range of initial puck locations and recovers gracefully from unexpected perturbations of the puck states during flight.

This article reviews the experimental setup and abstract theory we have developed. It describes the geometric constructs underlying the mathematical formulas that make up 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.

1.1. Geometric Robot Programming

A central theme of this article (and, indeed, of our general program of research in robotics [Koditschek 1986, 1987; Koditschek and Rimon 1990]) is the desirability of translating abstract user-defined goals into phase space

The International Journal of Robotics Research,
Vol. 13, No. 2, April 1994, pp. 101–118,
© 1994 Massachusetts Institute of Technology.

1. There is no relationship with the geometric programming technique for solving algebraic nonlinear programming problems.

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 dynamic systems. Their coupling via functional relationships—in this case, the mirror algorithm—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 by Andersson [1989]). 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 Koditschek and Rimon [1990]), thereby solving an entire range of problems with one controller structure. Although properly modular software is reusable, 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 when compared with 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 Sections 2.4 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. However, 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 biologic systems for inspiration (or who, more radically, treat their robots as plausibility models of biologic organization) will surely not be content with the grip of logic and syntax on their field.

The apparent disadvantage of geometric task encoding relative to syntactic prescriptions is a dramatic reduction in ease of expression. Regardless of whether 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 that follows 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 2.4 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 dynamic environment.

1.2. Intermittent Dynamic Environments

There is a large and important range of robotic problems requiring 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 (1986), whose careful experimental studies verify the correctness of his elegant control strategies for legged locomotion. McGeer (1990) has successfully used local linearized analysis to build a passive (unpowered) walking robot and believes that similarly tractable analysis should suffice for controlling running machines as well (McGeer 1989). Wang (1989a) has proposed using the same local techniques for studying open-loop robot control strategies in intermittent dynamical environments, although his ideas remain to be empirically verified. Research by Aboaf et al. (1989) on juggling 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 once a basically functioning system has been ensured. Thus, increasing numbers of researchers have begun to explore the problems of robotics in intermittent dynamic 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 an analysis similar to that reviewed in this article (Koditschek and Buehler 1991) that a greatly simplified version of Raibert's hopping algorithm (Raibert 1986) 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 later 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 the logical next step. Establishing the formal connection to gait mechanisms will obviously require more work.

Furthermore, we believe that the successful control by a one-degree-of-freedom robot of a two- and a four-degree-of-freedom intermittent dynamic 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 later (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 carefully studied manipulation involving impact with a dynamic environment (Mason 1986; Taylor et al. 1987; Wang and Mason 1987) and have recently begun the study of impacts with intermittent dynamic environments in the absence of sensors as well (Erdmann and Mason 1986; Wang 1989b). There is presumably a clear relationship between these theories: its elucidation would strengthen the applicability of each.

1.3. Organization

This article 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 in other publications, to which we will refer where appropriate. The central contributions of this article 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 and for catching falling objects, all with a one-degree-of-freedom revolute actuator.

2. The Mirror Algorithm for a Vertical One-Juggle

This section summarizes work reported elsewhere (Buehler et al. 1989, 1990) concerning the fundamental mirror algorithm. Section 2.1 presents our experimental setup and a simplified mathematic model. In Section 2.2 we state the task as an "environmental control problem"

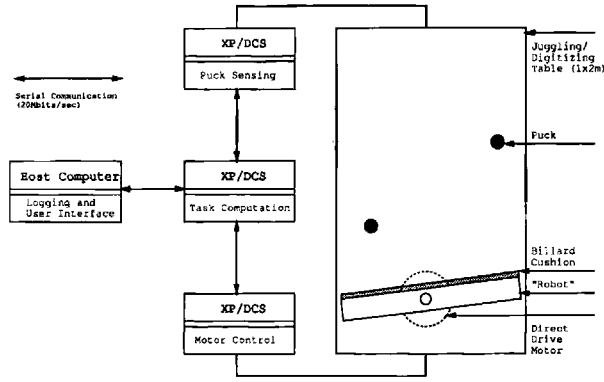


Fig. 1. The planar juggler.

that enables us to pose and solve robot juggling tasks as formal problems in control theory. This analysis, we feel, is central to understanding how robotic tasks in this domain may be planned and effected. In Section 2.3 we first introduce the vertical one-juggle task. We then demonstrate that, assuming the availability of state information at each successive impact, it is achievable via a linear feedback control law, based on the discrete contact state measurements. In Section 2.4 we will develop, for purposes of robot implementation, a vastly superior control strategy that uses continuous free flight information about the body. After a summary of analytical results showing that this new strategy—the "mirror algorithm"—is correct, we present experimental data of working one-juggles.

2.1. Robot and Environment Models

The experimental apparatus consists of one puck (or two pucks, for the two-juggle as described in Section 3.1), which slides on a plane, inclined 30 degrees away from the vertical line, and is batted successively by a simple "robot": a bar with billiard cushion rotating in the juggling plane as depicted in Figure 1. The entire length of the robot bar can interact with the puck. All sensor and controller functions are performed by a four-node distributed computational network formed from the INMOS Transputer-based Yale XP/DCS control node (Buehler et al. 1989). Implementation details describing how the computational resources are mapped to the mechanical hardware can be found in Buehler et al. (1989, 1990).

A puck trajectory with puck-robot collisions is depicted in Figure 2. The full puck state vector is denoted by $w = (b, \dot{b}) \in \mathcal{W}$, where $b \triangleq [b_1, \ b_2]^T$ represents the location of the sliding puck with respect to the inertial reference frame, $\mathcal{F}$, located at the center of the motor. The robot's angular position and velocity are denoted by r and $\dot{r}$, respectively. Throughout the article, puck

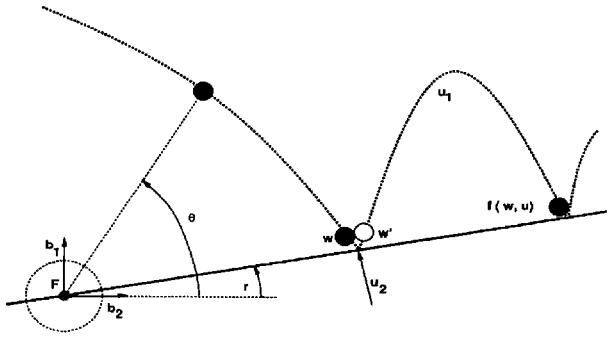


Fig. 2. The impact event. This illustration shows the relevant variables for developing the discrete map between successive robot-puck impacts and the one-juggle mirror law. The robot consists of a motor, located at the inertial frame $\mathcal{F}$, and a hitting bar, depicted by the long solid line. For illustration, several pucks are shown at different positions along a hypothetical trajectory.

or robot states are meant to be either continuous or discrete—evaluated just before impact. If the nature of the states is not obvious from the context, it will be made explicit (i.e., $w(t)$). State measurements taken just after impact will be primed (i.e., $\dot{b}'$).

To obtain an analytically tractable dynamic model that maps the puck states just before one impact into the puck states just before the next impact, we make a variety of simplifying assumptions: First, we assume that the puck-robot impacts occur instantaneously and can be described via a coefficient of restitution law that takes the a priori puck-robot states just before contact, into the new puck velocity vector after contact $\dot{b}'\left((b,r),(\dot{b},\dot{r})\right)$ derived carefully in Buehler et al. (1990) and Buehler (1990). Next, we will idealize the sliding puck to be a point mass and neglect the width of the robot link as well. Finally, friction is neglected. In reality, there is noticeable friction on the sliding plane, and we will compare numeric simulations of the robot control laws in the idealized frictionless environment with the same strategies run in the more realistic simulation model with coulomb friction, as against empirical data.

The future trajectory of the puck 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 model starting with the initial conditions just after impact, $w' = (b', \dot{b}')$, as

$$w(t) = \begin{bmatrix} b + \dot{b}'\left((b,r),(\dot{b},\dot{r})\right)t + \frac{1}{2}at^2 \\ \dot{b}'\left((b,r),(\dot{b},\dot{r})\right) + at \end{bmatrix}. \qquad (1)$$

Here $a = [0, -\gamma]^T$, and $\gamma = \gamma_{\text{grav}} \sin\beta$ denotes the projection of the gravitational constant of acceleration, γ_{grav}, onto the sliding plane inclined away from vertical by the angle β. For a more complete discussion and formal derivation of this model, we refer to Buehler et al. (1989). The vindication of our claim to have retained the essential aspects of a robot and environment whose dynamics are intermittently coupled is only possible by recourse to physical experiments, which will be provided in Section 2.4.2.

2.2. The Environmental Control Problem

The robot can determine where (or, equivalently, after how much elapsed time since the previous impact, t, now denoted u_1) to hit the puck and with what linear velocity ($\|b\|\dot{r}$, now denoted u_2) the impact should occur. In the mean time, the puck's behavior cannot be altered. Thus, on the most fundamental level, the robot's two actions represent the only means of imposing control on its environment. We may now investigate the response of the puck to all logically possible impact events by examining the *environmental control system* whose inputs are the puck states just before impact w, together with the robot's abstracted actions, $u = [u_1, u_2]^T \in \mathcal{U}$, and whose outputs are the puck's state just before the next impact, $f(w,u)$. In other words, we treat the robot as an independent external "agent of control" and consider the various puck behaviors resulting from the robot's actions. The resulting formal discrete dynamic control system

$$w_{j+1} = f(w_j, u_j),$$

is derived by substituting the robot's inputs at impact into (1),

$$f(w,u) = \begin{bmatrix} b + \dot{b}'(w,u_2)t + \frac{1}{2}au_1^2 \\ \dot{b}'(w,u_2) + au_1 \end{bmatrix}. \qquad (2)$$

This represents the underlying physical model with respect to which the robot's behavior must be rationalized. Formally, an *environmental control problem* results from prescribing some desired sequence of puck states,

$$\{w_j^*\}_{j=0}^{\infty},$$

and asking for an impact sequence,

$$\{u_j^*\}_{j=0}^{\infty},$$

which results in asymptotic convergence of w_j to w_j^*.

Since we are interested in sensor-based manipulation, we focus on solving such problems with feedback-based controllers. Although we prefer to avoid time-varying controllers, there is no a priori objection to dynamic controllers. In practice, the memoryless control structure presented here suffices for all the tasks we have encountered to date. Thus, a robot feedback strategy is a map,

$g : \mathcal{W} \rightarrow \mathcal{U}$, from the body's state to the robot's action set, $\mathcal{U}$, resulting in the impact strategy $u_j = g(w_j)$. The robot-environment closed-loop dynamic system is formed from the composition of f (2) with g,

$$w_{j+1} = f_g(w_j); \qquad f_g(w) \triangleq f(w, g(w)). \tag{3}$$

2.3. The Vertical One-Juggle Task

Probably the simplest systematic behavior of this environment imaginable (after the rest position), is a periodic vertical motion of the puck in its plane. Specifically, we want to be able to specify an arbitrary vertical puck impact velocity (and thus an apex position) together with a vertical impact position and, from arbitrary initial puck conditions, force the puck to attain a periodic trajectory that impacts at zero vertical position and passes through that apex point. We call this task the *vertical one-juggle*. Such tasks are exactly represented by the choice of a desired fixed point, w^*, of (2) (for a more formal task definition, see Buehler et al. [1989]). Note that a purely vertical steady-state trajectory along a desired horizontal position b_1^* requires zero horizontal velocity, $\dot{b_1}^* = 0$, throughout the puck's continuous trajectory including the moment of impact. Moreover, a fixed vertical impact velocity, $\dot{b_2}^*$, from a specified impact height, b_2^*, implies a specified apex position by conservation of energy during the free flight phase. Thus, a putative fixed point, w^*, whose horizontal velocity component and vertical position component are set to zero corresponds to a specific vertical one-juggle.

It is clear that not every constant set point, w^*, can be made a fixed point of the environmental control system (3). The allowable vertical one-juggle set points are determined by the fixed points of the discrete impact map, f_g. In this fashion it can be shown (Buehler et al. 1989) that only set points with zero vertical puck position and zero horizontal velocity can be made fixed points. However, it is not enough to merely achieve a vertical one-juggle fixed point. We must also ensure that perturbations away from the desired behavior are dissipated. We show in Buehler et al. (1989) that at any fixed point of (3), the system (2) is controllable. Therefore, it follows from linear control theory that any vertical one-juggle fixed point of (3) can be made locally asymptotically stable by a (discrete) linear affine state feedback law,

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

Here, $(w - w^*)$ denotes the vector of puck state errors at impact, K_Λ a matrix of feedback gains, and u^* the robot control inputs resulting from the fixed point condition. Thus we have shown that the vertical one-juggle is logically achievable.

This discrete analysis confirms the intuition that only state information at impact should be required for a successful juggling algorithm and 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 that stabilizes a desired set point, w^*. Recall that g merely specifies discrete robot control inputs at impact. It is completely up to the designer to solve the robot control problem (i.e., to construct a continuous robot torque command that achieves an approximation to the required impact strategy, g). 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 article (Buehler et al. 1988) we have reported our failure to achieve experimental success with any implementation of a locally stabilizing feedback strategy (4) based on discrete impact state measurements for these reasons.

2.4. The Mirror Algorithm

We now introduce the mirror algorithm, which remedies the shortcomings of the algorithm described in the previous paragraph. In Section 2.4.1 we provide an intuitive motivation for the mirror law and review our analytical results concerning this algorithm. Next, Section 2.4.2 attests to the empirical relevance of our findings.

2.4.1. Intuitive Motivation and Formal Results

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 μ for the robot that 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"

$$\theta = \arctan \frac{b_2}{b_1},$$

to control the puck's vertical motion,

$$\mu(w) \triangleq -\kappa_1(w)\theta + \kappa_2(w), \tag{5}$$

modulo a proportional-derivative (PD) feedback term for stabilization around a fixed desired horizontal position, $(b_1^*, \dot{b}_1^* = 0)$,

$$\kappa_2(w) = \kappa_{21}(b_1 - b_1^*) + \kappa_{22}\dot{b}_1. \tag{6}$$

Both κ_{21} and κ_{22} are constant gains. In the sequel we will assume that the robot r tracks the reference trajectory μ sufficiently accurately and thus will assume that $r = \mu$.

Second, to stabilize the vertical periodic motion, we borrow from Raibert (Raibert 1986; Koditschek and Buehler 1991) the idea of modifying the robot's trajectory by "servoing" on the discrepancy between the puck's currently measured total mechanical energy (constant during flight, as we neglect friction) $\eta(b_2, \dot{b}_2) = \frac{1}{2}\dot{b}_2^* + \gamma b_2$, and its value at the desired steady states $\eta(b_2^*, \dot{b}_2^*) = \eta^*$, $e_\eta \triangleq \eta^* - \eta$,

$$\kappa_1(w) \triangleq \kappa_{10} + \kappa_{11} e_\eta. \tag{7}$$

Here κ_{10} is determined by the fixed point condition, and κ_{11} is again a fixed constant gain. The gain function $\kappa_1(w)$ directly modulates the robot's vertical impact velocity, as can be seen when taking the derivative of (5),

$$\dot{r} = \dot{\mu}(w) = -\kappa_1(w)\dot{\theta}, \tag{8}$$

assuming no horizontal position error and no horizontal motion. Deviations from the desired vertical total energy result in higher or lower robot impact velocities, forcing the puck trajectory onto the desired set point. Similarly, the PD term (6) translates horizontal position and velocity errors into impact angle offsets from the (otherwise) zero impact angle, thus creating restoring horizontal velocity components. The coupling effects between these two stabilizing mechanisms are treated as perturbations. A more complete intuitive account of this mirror algorithm is provided by Buehler et al. (1989, 1990) using the one-degree-of-freedom juggler restricted to the vertical line.

As a consequence of choosing a smooth function for our control law (5), a stability analysis of the resulting four-dimensional nonlinear closed-loop system is possible. It turns out (Buehler et al. 1989, Proposition 5.3) that the robot's "mirroring" motion induces a three-dimensional invariant submanifold of the environmental control system (2). Therefore, (Buehler et al. 1989, Corollary 5.3) the local stability behavior of any valid vertical one-juggle task, w^*, may be adjusted by the appropriate choice of gains in (5).

Although this result obtained from the linearized system provides a formal proof of correctness 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 (Buehler and Koditschek 1990). These results provide us with more insight into the qualitative behavior of the higher dimensional system and might suggest how to achieve the nonlinear analysis.

2.4.2. *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. 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 all the data displayed below, we have chosen[2]

$$w^* = \begin{bmatrix} b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix} = \begin{bmatrix} 0.28 \text{ m} \\ 0 \\ 0 \\ -3.175 \text{ m/s} \end{bmatrix}.$$

The analytical results guarantee that for this fixed point, the vertical one-juggle task is achievable, and they reveal how to choose the specific gain settings to keep all poles inside the unit circle.

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 (Buehler et al. 1990) that w^* is not, in general, globally asymptotically stable. Notice that, even at steady state, the impacts do not occur at zero height, but rather at a fixed offset: this is due to the nonzero puck radius and the robot dimensions, which were ignored in the simplified model but taken into account in the implementation. Despite many 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.

Next, we present data plots of the puck states just before impact. These figures display statistical information (mean and standard deviations) obtained from one single sequence of 20 successive runs (without hand selection). Figure 4 compares the responses of the analytical model with and without friction with the responses of our experimental setup (with friction) for two different initial conditions. These two initial conditions for the puck impact states result from dropping the puck from two

2. These and subsequent numeric set points seem "odd" because they have been converted from nonmetric units; our qualitative results do not depend critically (as some readers might suspect) on these exact numeric values.

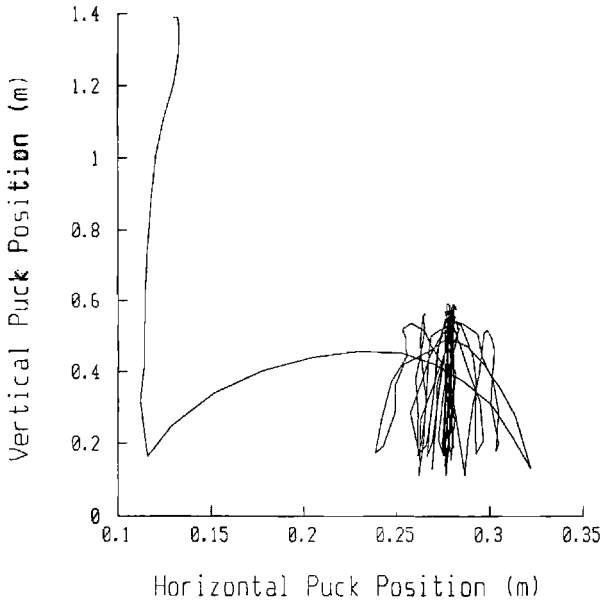


Fig. 3. Sample continuous data. A qualitative continuous "recording" of a vertical one-juggle that illustrates the rapid convergence to the desired periodic motion for even large inital errors.

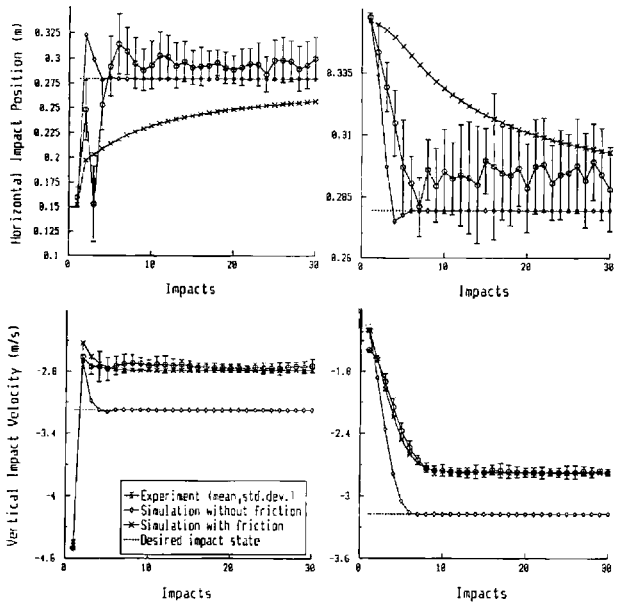


Fig. 4. One-juggle impact data. Here simulations are compared with statistical experimental data of one-juggle runs. We show the mean and standard deviation derived from 20 successive experiments. The upper plots contain the initial transients in horizontal impact positions and the lower plots in the corresponding vertical impact velocities. For the left two plots, the puck was released from a position on a juggling plane to the left and higher compared with the desired apex position. In contrast, for the right two plots, the puck was released to the right and lower compared with the desired apex position.

extremes of the right juggling half-plane, the upper left (initial condition 1) and the lower right corner (initial condition 2). 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 expected, the effect of the unmodeled friction is a steady-state deviation, which, second, is accurately predicted by a simple one-degree-of-freedom model for the vertical motion 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. The main reason for this benign discrepancy is that our apparatus' friction dynamics are more complicated than our computer model. While at the higher vertical velocities the dry friction model is fairly accurate, the actual friction at the lower horizontal velocities is much smaller than modeled. This explains why the transients in horizontal impact positions are closer to the model without friction than to the one with friction.

3. Applications

We now present two informal modifications and extensions of the provably correct vertical one-juggle algorithm described in the previous section. First, the vertical two-juggle task discussed in Section 3.1 introduces new aspects of timing and combinatorial choice to the problems of intermittent dynamic 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 dynamic 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 maintaining two vertical one-juggles. One puck is juggled on each side of the robot actuator in a specified periodic orbit via repeated impacts. A successful two-juggle algorithm must intermittently control the horizontal and vertical impact positions and vertical impact velocities of the two pucks and 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 to achieve the two-juggle. The resulting algorithm maintains the pure geometric features of the original mirror law in that it maps the continuous phase space trajectory of both pucks 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 this 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 are presently pursuing 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 the left and one on the right half of the juggling plane. Here and in the sequel, the subscripts l, and r denote quantities associated with the left and right puck, respectively. As before, the individual one-juggles are encoded via the desired constant sequence of puck impact states just before impact,

$$w_l^* = \begin{bmatrix} -b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix}, \quad w_r^* = \begin{bmatrix} b_1^* \\ 0 \\ 0 \\ \dot{b}_2^* \end{bmatrix}. \tag{9}$$

As the two one-juggles cannot be controlled independently by the one-degree-of-freedom robot, (9) is not yet a complete task definition for the two-juggle. An added "phase angle relationship" between the two one-juggles is needed. For simplicity, we shall specify for this initial implementation that alternative impacts occur maximally separated in time. However, time can be eliminated by the following specification in puck phase space: when one puck impacts, the other puck attains its apex.

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 to keep them away from the emergency situation. We loosely refer to the first as the global phase angle problem and to the second as the local phase angle problem.

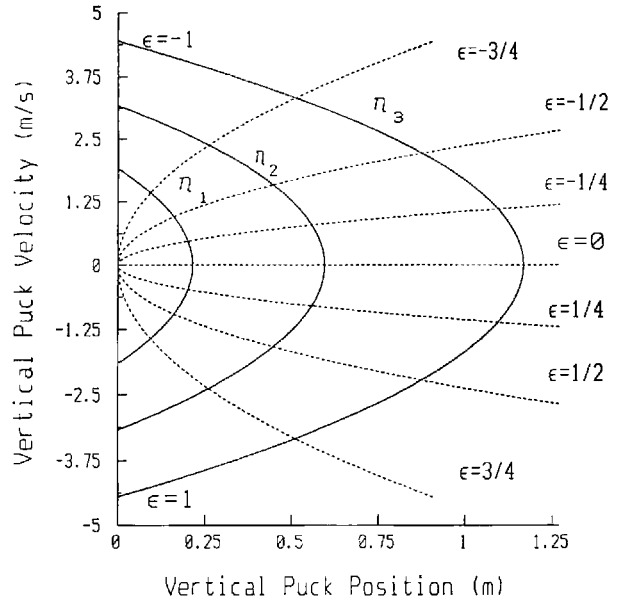


Fig. 5. The phase angle (illustration). The puck flight trajectories for three different total energy levels η_1, η_2, η_3 are shown in the phase plane. The dashed lines indicate lines of equal phase angle ϵ (10). It maps a puck flight trajectory between two impacts onto the interval $(-1, 1)$ in a geometric fashion and independent of the total puck energy.

The Local Phase Angle Control Problem. The puck phase angle is a purely geometric measure (without dynamics and use of time) that indicates how much of the total vertical trajectory between two impacts has been traversed. It will form the basis for the phase angle error used to keep the puck's impacts equally separated in time. To decouple the control of the phase angle, its measure should not depend on horizontal position or total vertical energy η. Those quantities are already controlled by the existing one-juggle mirror law. One such phase angle measure,

$$\epsilon \triangleq -\frac{\dot{b}_2}{\sqrt{2\eta}}, \tag{10}$$

maps any puck trajectory onto the interval $(-1, 1)$ between impacts and evaluates to zero at the apex. Moreover, its time derivative is constant. The phase angle is illustrated in Figure 5, which shows three puck trajectories with different total energy levels η_1, η_2, η_3 and some "equi-phase angle" level lines from just after impact ($\epsilon = -1$), to the apex ($\epsilon = 0$), to just before impact ($\epsilon = 1$). The phase angle takes advantage of the fact that all impacts occur at or close to zero height. The meaning

of the mapping (10) becomes immediately clear when one considers the ideal case without friction. Now the total vertical energy between impacts w_j' (the energy just after the jth impact) and w_{j+1} (the energy just before the $j+1$st impact) is unchanged,

$$\eta(t) = \eta'(t_j) = \frac{1}{2}\dot{b}_{2,j}'^2 + g b_{2,j} = \frac{1}{2}\dot{b}_{2,j}'^2 = \eta(t_{j+1}).$$

Therefore, in the absence of friction,

$$\epsilon = -\frac{\dot{b}_2}{\|\dot{b}_{2,j}'\|}. \tag{11}$$

Here, $\dot{b}_2$ is the continuous puck vertical velocity, and $\dot{b}_{2,j}'$ is the puck velocity just after the previous impact. This representation illustrates more directly than (10) the mapping between the puck trajectory and the $(-1, 1)$ line segment but has the drawback of depending on single discrete measurements of $\dot{b}_2'$. In addition, (10) still maps a puck trajectory onto the complete $(-1, 1)$ interval even in the presence of friction—that is, when the total energy is not preserved.

A geometric version of the *phase angle error* built from this function,

$$e_{ph}(w_l, w_r) = [\epsilon(w_l) - \epsilon(w_r)]^2 - 1, \tag{12}$$

vanishes when 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 this desired phase angle separation, the larger the phase angle error grows, as illustrated in Figure 6, with two pucks' trajectories with significant phase angle error.

We will now describe how the phase angle error quantities can be used to tune the robot impact velocity to achieve the desired phase angle separation. 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, or else to force it back toward zero even at the temporary expense of the accuracy of either puck's vertical one-juggle. The phase angle is changed by tuning the puck's flight time. The flight time, in turn, is modified via the robot impact velocity. This mechanism is used already to stabilize the vertical one-juggle (8). The phase angle error term may now, in the same fashion, be added to the original mirror algorithm gain (7):

$$k_1(w_l, w_r) = k_{10} + k_{11} e_\eta(w) + k_{12} e_{ph}(w_l, w_r),$$

where w are the continuous puck states for the left or the right puck in the mirror law for the left (μ_l) or right puck (μ_r), respectively. The local phase angle control gain k_{12} is determined empirically. The remainder of the

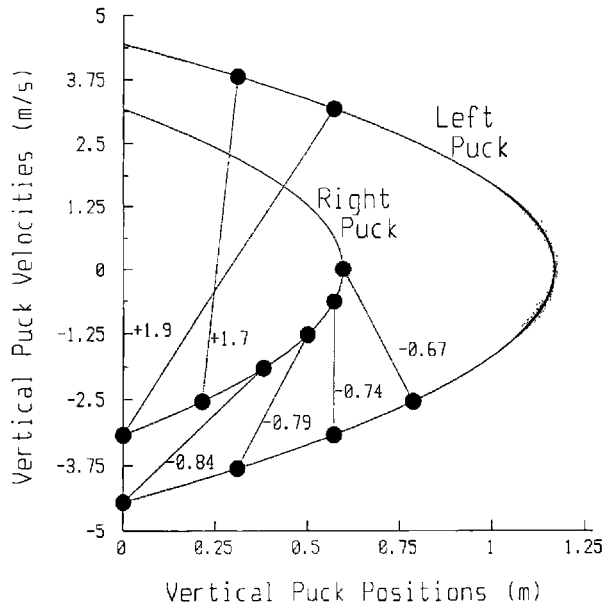


Fig. 6. The phase angle error (illustration). A phase angle ϵ_l, ϵ_r is assigned to the left and the right puck, respectively. Now the phase angle error $e_{ph} = [\epsilon_l - \epsilon_r]^2 - 1$ represents a geometric measure for how far the two pucks' trajectories are away from the desired phase angle relationship: when one puck impacts, the other should be at its apex. The figure shows six successive snapshots of a right and a left puck on trajectories with different total energies and a phase angle error. A pair of a right and a left puck belonging to the same snapshot are linked by a solid line. In addition, the numeric value shows the phase angle evaluated at that time. When the left puck impacts, e_{ph} is negative (−0.84), and when, after two more time steps, the right puck impacts, e_{ph} is positive (+1.9). Thus e_{ph} can be employed in the two-juggle mirror law to modulate the robot's hitting velocity, which affects the flight time, to achieve the desired phase angle.

original one-juggle mirror algorithm (5) is unchanged. Unfortunately, as $\dot{e}_\eta \neq 0$, it follows that $\dot{k}_1 \neq 0$, and thus (for simplicity, along the desired horizontal position) the desired robot velocity just before impact is

$$\dot{\mu} = -k_{12}\dot{e}_{ph}\theta - k_1\dot{\theta}.$$

The first term on the right side seems to affect our ability to control the robot impact velocity solely via k_1 as before. Fortunately, the effect of the first term is θ dependent and thus small for small puck angles θ (that is, close to impact). The implementations confirm the validity of this crude argument.

The Global Phase Angle Control Problem. When both pucks are falling toward the bar nearly at the same time,

it is imperative to service the nearest first, sacrificing any needed one-juggle performance measures to the work of restoring phase angle separation. This intuition can be realized by assigning an urgency measure to each individual one-juggle. While there are many ways to implement this, we will again choose geometric programming and encode this urgency measure as a continuous "weighting function" in puck phase space. A simple weighting function, σ, maps the puck's vertical position and velocity into the unit interval, increasing from zero to one and evaluating to one half at the distance ρ from impact:

$$\sigma(w) = \sigma_1(w)\sigma_2(w),$$

$$\sigma_1(w) = \frac{1}{2} - \frac{1}{\pi}\arctan[k_p(b_2 - \rho)],$$

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

The position weight function σ_1 constitutes the heart of the global phase angle separation. With ρ and k_p, k_v set properly, it evaluates to one in the vicinity of the robot and decreases to zero farther away. Using a simple vertical distance works out well because of the simplifying fact that impacts occur close to $b_2 = 0$, even when the puck's horizontal position and velocity errors are nonzero. While we based the smooth weighting functions on arctan functions, other equally well-suited approaches could employ, for example, the very similar sigmoidal function

$$\frac{1}{1 + e^{-x}}.$$

The velocity weight function σ_2 scales σ_1 in a smooth fashion such that weight is assigned only during the descending part of the puck trajectory. The function s then normalizes the contributions of $\sigma(w_l)$ and $\sigma(w_r)$, mediating between the two reference trajectories from the two puck's independent one-juggle algorithms:

$$s = \frac{\sigma(w_l)}{\sigma(w_l) + \sigma(w_r)}.$$

The Extended Mirror Law. Combining the two approaches to the local and the global phase angle control problem, the mirror law extension for the two-juggle may now be written as

$$\mu_{2jug} = s\,\mu(w_l) + (1 - s)\,\mu(w_r). \tag{13}$$

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 the horizontal position 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, as any small difference between the two pucks' states contributes to the phase separation. In fact, we conjecture that this corresponds to an unstable fixed point of the closed-loop system.

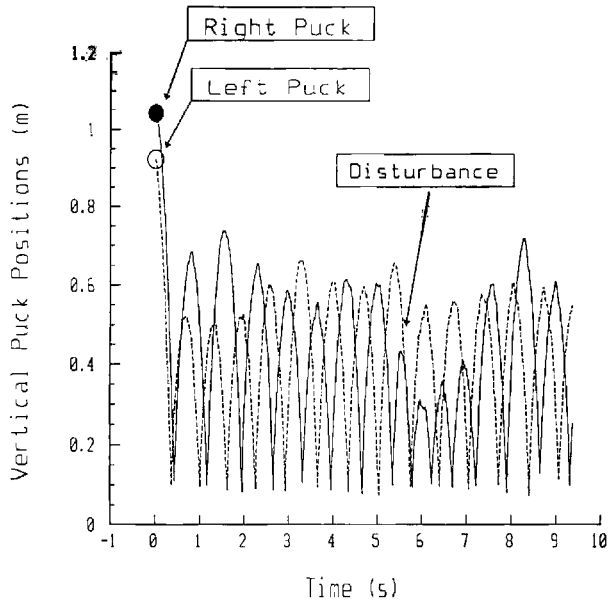


Fig. 7. Continuous vertical positions of the two-juggle. This qualitive continuous "recording" of a two-juggle illustrates its strong stability and robustness properties. During the first 5 s, large initial errors in apex height and phase angle are quickly eliminated. Then the right puck is knocked away by hand before its apex. Again, this experimental data plot shows the recovery from this unexpected external disturbance. Notice that the right puck impacts twice before the next left puck impacts. This is a common occurrence for such large perturbations but presents no problem for the mirror law, as no assumptions about alternating impacts are made.

3.1.2. Discussion of Experimental Data

We implemented the two-juggle algorithm as described earlier on our planar juggling apparatus. To provide better qualitative insight into its convergence and robustness properties, we display in Figure 7 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 apex), but also, as can be clearly seen at the first impact, in almost simultaneous impacts (large initial phase angle error). Subsequent impacts display the recovery from both the error in phase angle and the 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 error (height) 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.

The plots in Figure 8 show the simultaneous convergence properties of three initial errors in horizontal impact positions, vertical impact velocities, and phase angle error. Both pucks were dropped by hand simultaneously from their initial positions ($b_1 = -0.356$ m, $b_2 = 0.838$ m and $b_1 = 0.356$ m, $b_2 = 0.965$ m for the left and right puck, respectively). During the following 100 impacts (50 per puck), each puck's horizontal impact position, b_1, the vertical impact velocity, $\dot{b}_2$, and the phase error e_{ph} were recorded. This was repeated for 32 runs. Two runs were discarded because they failed during the first five impacts. All other runs completed successfully. From the completed runs we show the mean with error bars indicating one standard deviation in both directions. Also, to show the initial error transients more clearly, we display only the first 30 impacts per puck. In Figure 8(1) and 8(2) we see the convergence to steady state within about five impacts for the initial transients in horizontal impact positions. The 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 billiard cushion is compressed). Such modeling errors translate into shifts in horizontal impact positions. Next, Figure 8(3) and 8(4) show both puck's convergence to the impact velocity predicted by a simple one-dimensional model taking friction into account, in a similar fashion to Figure 4. Finally, Figure 8(5) shows that at the same time, the phase angle error between the two pucks is rapidly eliminated. In all the plots, the standard deviations, after some increase during transients, stay fairly constant.

In addition to the displayed convergence properties, this algorithm has additional desirable attributes. It runs consistently for hundreds and hundreds of impacts. Notice that (again, in consequence of the geometric nature of the algorithm) at no point in time does it depend on a single measurement of a discrete event, such as flight time, impact position, impact phase angle error, or apex positions. This results in a strong tolerance to noise and measurement 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, (13), described earlier. There is no additional higher level decision making or conditional branching as found in implementations

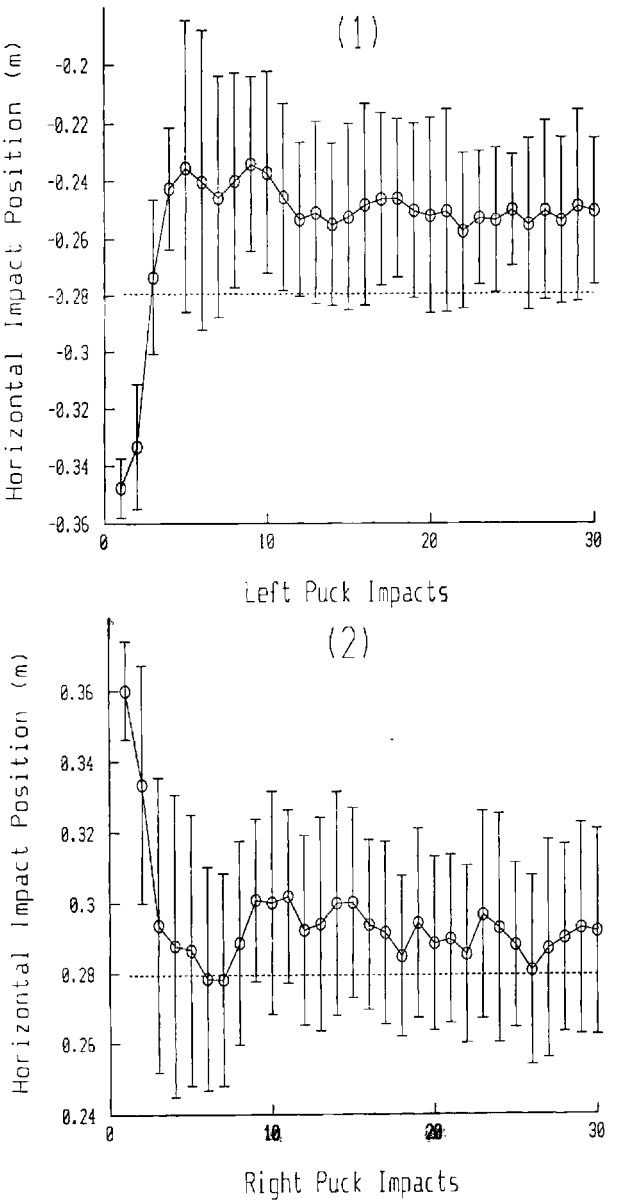
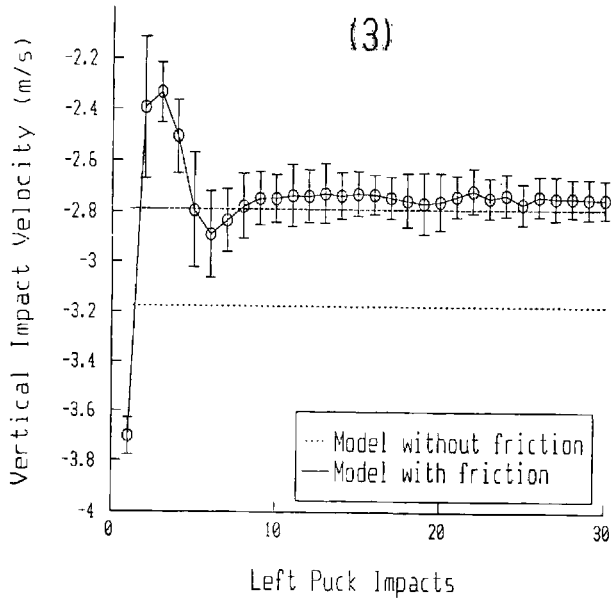


Fig. 8. Two-juggle impact data. The plots (1)–(5) show how simultaneous initial errors in horizontal impact positions and vertical impact velocities of both pucks, as well as an initial large phase angle error between the two pucks, are eliminated. In addition, to demonstrate the consistency of this performance, we show the mean and standard deviation of 30 successive runs.



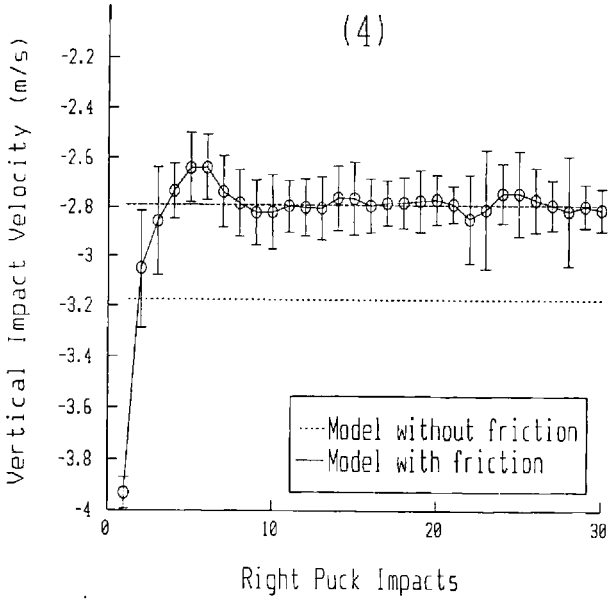


Fig. 8. cont.

purely characterized by a computer program. Therefore, not only simple and robust, this approach lends itself in a similar fashion as the one-juggle (Buehler et al. 1989) to rigorous analysis.

3.1.3. Alternative Extensions

There are many different approaches to the two-juggle that 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 earlier. The global phase angle separation

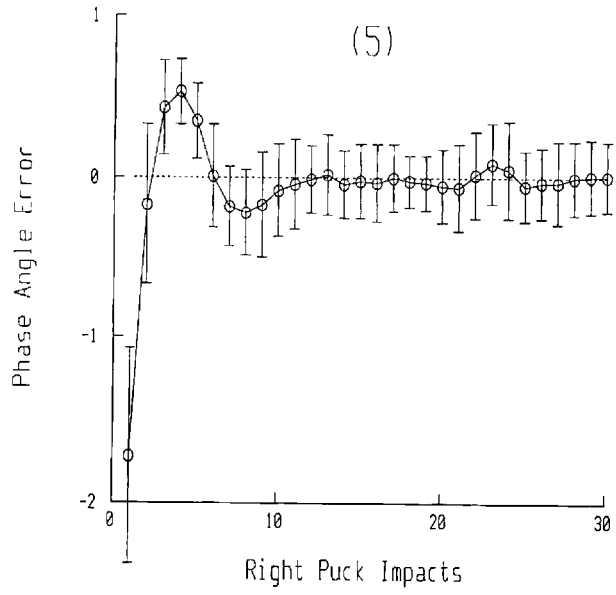


Fig. 8. cont.

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 that 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 earlier. It relies strictly on the expectation that the two pucks will impact alternately. 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 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 large as the ones displayed in Figures 7 and 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

We now provide a second application of mirror algorithms to intermittent dynamic robotic tasks. We will investigate a behavior opposite to juggling: the problem of catching. The same notions of geometric programming discussed in the previous section may be used to devise

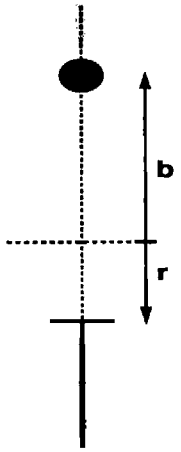


Fig. 9. The line-juggler.

a controller here. In contrast, since the catch is a nonperiodic behavior demanding a "dead-beat convergence" capability, it seems clear that the asymptotic analysis we have used to demonstrate correctness of the one-juggle (Buehler et al. 1989; Buehler and Koditschek 1990) will not be of use. Thus, a theoretical treatment of these ideas lies beyond the scope of the present article.

In Section 3.2.1, we present a one-degree-of-freedom definition of a catch. Section 3.2.2 describes its control via a mirror law. This catching mirror law is then extended in Section 3.2.3 to the two-degree-of-freedom case for the same physical planar juggling apparatus that was used to implement the previous juggling tasks. Experimental evidence of successful catches presented in Section 3.2.4 attests to the validity of the ideas.

3.2.1. Definition of the Catch for the Line-Juggler

Denote by $(b, \dot{b})$ and $(r, \dot{r})$ the puck's and the robot's position and velocity in workspace, respectively, as shown in Figure 9. For simplicity, both puck and robot are assumed to be point objects. A sensible outcome of a catch could be formally characterized as a steady state for both puck and robot contained in the "catch set,"

$$\mathcal{A} \triangleq \left\{ \left((b, \dot{b}), (r, \dot{r}) \right) \; : b = r, \;\; \dot{b} = \dot{r} = 0 \right\}, \qquad (14)$$

the points in puck-robot phase space with the same position and zero velocity.

There exist at least two fundamentally different ways to achieve the goal as stated in (14). First, 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 employed for the analysis of the vertical one-juggle may fail, since a vertical one-juggle set point with zero vertical velocity is contained in a degenerate set whereon it is not completely controllable. Moreover, the vertical one-juggle task is defined in terms of asymptotic convergence to a fixed point. However, in the present setting such 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. Note also that this approach will work only if the puck's velocity vector is aligned with the gravitational acceleration. Thus 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 "dead-beat catch." We then need to refine our previous definition (14) to exclude the asymptotic catch.

The dead-beat catch is characterized by two distinct phases: the pre- and postcontact phases. Proceeding on obvious heuristic ground, we shall define the precatch phase via a velocity matching condition,

$$\mathcal{A}' \triangleq \left\{ \left((b_c, \dot{b}_c), (r_c, \dot{r}_c) \right) \; : b_c = r_c, \;\; \dot{b}_c = \dot{r}_c \right\}, \qquad (15)$$

where all states are evaluated at time of first robot-puck contact, $t = t_c$. Specification of the postcontact phase appears to require a much more comprehensive model of the force interactions between the robot and puck. For example, recall that the juggling environment dynamics (2) assumed a simplistic "coefficient of restitution" impact model that will clearly be inadequate to describe the mechanical impedance our billiard cushion presents the puck's mass during conditions of sustained contact. Thus, any general notion of desirable postcontact behavior seems hostage to the particularity of contact material properties. Even given such a model, the specification and control of postcontact behavior will likely depend significantly on the sensory information (force, slip, puck states). In the present article, we will concentrate solely on achieving the pre-contact behavior—a velocity matching contact $\mathcal{A}'$—and move the puck to a desired postcontact state while maintaining contact by recourse to a simple (properly tuned!) PD controller.

3.2.2. The Dead-Beat Catch for the Line-Juggler

Motivated by the success of geometric programming of a mirror-like robot behavior for juggling, here we use a scaled-down or shadow-like image of the puck trajectory to generate the online robot reference trajectory. Suppose, without loss of generality, that the velocity matching robot-puck contact should occur at $b^* = 0$. Consider the function of puck position via

$$\mu(b) = \kappa_1 \arctan \kappa_2 b, \qquad (16)$$

where κ_1 and κ_2 are constant gains. Forcing the robot to track the trajectory that results from this function when

the puck moves in space gives rise to a contact surface whose geometry may be "programmed" to achieve the desired behavior. Before demonstrating this, observe that (16) enjoys two desirable features. It is smooth (in fact, analytic), which both eases practical tracking requirements and promises to simplify the eventual analysis (which is not provided in this article). Moreover, it is bounded and limits the robot position even in the case of very large puck positions.[3]

Assume, as before, that the robot is able to asymptotically track the desired trajectory, and we will assume that $r(t) = \mu(b(t))$. It is easy to see from (16) that contact between robot and puck occurs whenever the puck passes through $b = b^* = 0$, provided $\kappa_1 \cdot \kappa_2 \leq 1$. Furthermore, the velocity matching condition between robot and puck at contact is satisfied whenever

$$\kappa_1 \cdot \kappa_2 = 1. \tag{17}$$

This can readily be derived from

$$\dot{r} = \dot{\mu} = \kappa_1 \frac{\kappa_2}{1 + \kappa_2^2 b^2} \dot{b},$$

and

$$\dot{r}_c = \dot{r}(b_c = b^* = 0) = \kappa_1 \kappa_2 \dot{b}_c.$$

Thus the catching mirror law (16) together with the velocity matching condition (17) will result in a successful precontact phase $\mathcal{A}'$. The freedom left in satisfying (17) translates into a tradeoff between the maximum robot position for large puck distances versus tracking requirements close to contact.

3.2.3. *Extending the Catch Mirror Law for the Plane-Juggler*

The simple-minded but effective approach to catching described earlier can readily be extended to achieve catching on the plane-juggler in our laboratory, as shown in Figure 1. We now replace the puck position in (16) with the puck angle, $\theta(b) = \arctan b_2/b_1$, to obtain the planar catching algorithm,

$$\mu = \kappa_1 \arctan \kappa_2 \theta. \tag{18}$$

As before, the contact should occur on the horizontal line; thus, if the robot tracks its trajectory correctly, then contact will occur at the specified vertical position. There is no specification of horizontal position, since the robot can catch along its entire link. Thus, we must ensure that the precontact conditions (15) obtain at any contact configuration (that is, in spite of any nonzero horizontal puck velocities). This follows from direct computation,

$$\dot{r}_c = \dot{\mu}_c = \kappa_1 \frac{1}{1 + \kappa_2^2 \theta_c^2} \kappa_2 \cdot \dot{\theta}_c = \kappa_1 \kappa_2 \dot{\theta}_c, \tag{19}$$

which gives the velocity matching condition,

$$\dot{r}_c = \dot{\theta}_c \tag{20}$$

at $\theta_c = 0$, whenever $\kappa_1 \kappa_2 = 1$.

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 dead-beat 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 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 (theoretically, after an infinite amount of time; in practice, after four or five impacts), one commands a dead-beat catch.

3.2.4. *Discussion of Experimental Data*

This section presents experimental catching data from our planar juggler using the catching mirror law (18). Figure 10 shows the vertical puck positions in the course of a typical puck transfer routine as described in the previous paragraph. The circle at the beginning of the data plot represents a realistically scaled puck with a diameter of 7.5 cm. The plot displays the last three impacts produced by the juggle law (5). After the last juggle impact at about 2.6 s, the robot executes the catching algorithm (18). The plot displays an offset of about 12.2 cm because of the finite size of the robot's link and nonzero puck diameter, which we have ignored in the discussion heretofore for ease of exposition. It is worth mentioning that the ripples in the displayed trajectory demonstrate a systematic measurement error introduced by the inductive position sensing method (the actual trajectory is much smoother). The error is periodic in position and is caused by the discrete spacing of inductive loops 2.54 cm apart. This ripple "frequency" can be verified in the plot and is seen best at the lower puck velocities at apex and during the catch. After the contact the close proximity of the metal robot bar to the puck's inductive field creates an additional error. Together they explain the apparent oscillation just after the instant of contact at 3.25 s. This is an artifact. In reality, the puck does not lose contact with the robot.

Next, we drop the puck from different heights and show the resulting catch trajectories without any modification to the catching algorithm. As expected, the

3. We have chosen the arctan function here, but many others, for example, the sigmoidal function $1/(1 + e^{-b})$ are eligible as well.

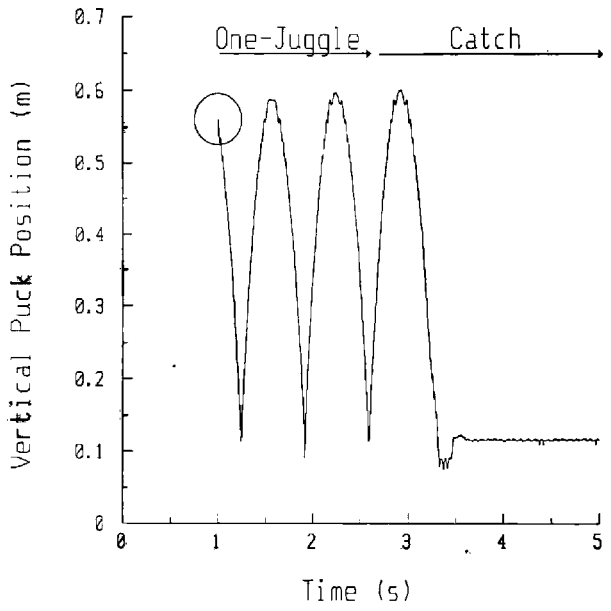


Fig. 10. Continuous vertical puck position of a one-juggle followed by a catch. After the third puck-robot impact, the robot controller switches from the one-juggle to the catch.

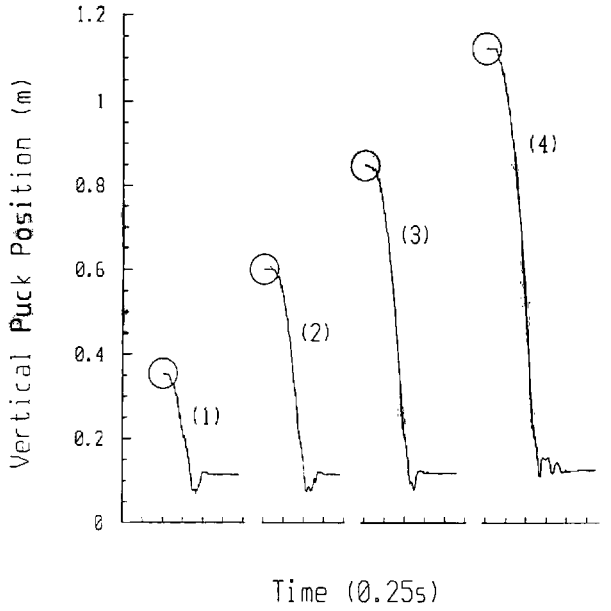


Fig. 11. Smooth catches with different contact velocities. The catching algorithm accomplishes the catching of pucks dropped from increasing heights smoothly, as shown in curves (1)–(3), without any changes. However, in curve (4) the dropping height is so large that the robot cannot track the online "shadow" trajectory sufficiently accurately. This results in a temporary loss of contact during the catch.

catching performance should be independent of the contact velocity of the puck. This is verified in Figure 11 in curves (1)–(3). However, as the initial height—and thus the contact velocity—is gradually increased, the robot will eventually fail to track the reference trajectory defined by (18). Data curve (4) in the same figure depicts such a case: the puck is dropped from about 1 m above the robot bar, and a comparison of sensor estimate and robot state shows a significant error. In consequence, contact occurs earlier than at $\theta = 0$, and there is no velocity match at that contact. Naturally, the tracking error could be reduced by recourse to more sophisticated trajectory tracking control algorithms (for example, feed forward controllers). However, these require a dynamic model of the object to be caught, and our simple PD controller does not. In any case, all physical robots will have torque and velocity limitations, eventually resulting in tracking errors regardless of the controller used. Even for a high-performance system, it would be a challenge to catch a fast-flying baseball in a velocity matching fashion.

As might be expected, the severity of velocity mismatch at contact caused by tracking errors depends strongly on the impact dynamics and the post contact strategy. Figure 11 presents data taken using a very stiff low-loss spring—the billiard cushion chosen to expedite the juggling—and an open-loop robot control strategy for the post contact phase (that is, there is not even a contact sensor, let alone a force transducer). Figure 12 distinguishes the relative importance of catching algorithm and impact dynamics for the excessive dropping height corresponding to curve (4) in the previous figure. Curve (1) shows the trivial asymptotic catch with the robot locked in place at the desired catching angle. Since the coefficient of restitution is less than one, the puck will eventually come to rest after numerous bounces. The next curve (2) shows the puck being dropped from the same height while the robot employs our catching law (18). In consequence of velocity mismatch, two residual bounces occur (the large one-sided oscillation is not a sensor artifact). The deleterious effects of velocity mismatch may be considerably diminished by changing the impact dynamics as depicted in curve (3). Here, the billiard cushion is topped with a 5 mm thick layer of soft foam. We still observe premature contact but eliminate completely any subsequent bounces.

4. Conclusion

Our research aims at developing a framework for the specification, control, and analysis of intermittent dynamic tasks. This class includes, for example, dynamic manipulation, juggling, catching, throwing, as well as dynamic legged locomotion tasks. Such capabilities require (often repeated) interactions with an environment possessed of its own independent dynamics that can be

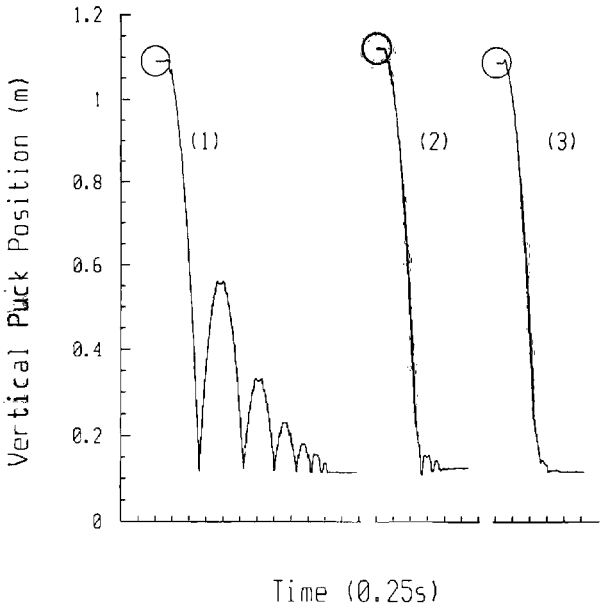


Fig. 12. Improving the catch. Three different catches are shown of pucks dropped from the same height. Curve (1) depicts the trivial asymptotic catch, where the robot bar is stationary. Curve (2) is identical to curve (4) in the previous figure and shows the puck bounces resulting from robot tracking limitations. By increasing the dissipative properties of the contact material (a layer of dissipative materials was added on top of the billiard cushion), these bounces are readily eliminated in curve (4), without any changes to the catching mirror law.

controlled only through intermittent contact with the robot. We have demonstrated here that certain examples of such behavior may be parsimoniously specified in terms of a desired contact condition. In turn, controllers that realize these contact conditions may be synthesized by recourse to "mirror algorithms"—smooth functions that map continuous environment states into desired robot states—reducing the robot's task to a matter of mere tracking. Experiments reveal the effectiveness of this methodology in achieving robust and reliably repeatable instances of the desired behavior even in the face of severe disturbances.[4] Finally, for those tasks requiring repetitive contacts, the correctness of the specification and the resulting controller synthesis may be determined by recourse to standard methods of dynamic systems theory.

4. It is worth remarking here that the robot's level of dexterity is quite advanced by any measure. We are convinced that no human could succeed in keeping two pucks up in the air by swinging a bar around a single revolute joint anywhere nearly as capably.

4.1. Practical Extensions of the Present Work

Perhaps the most obvious application of these ideas is to dexterous manipulation in relatively unstructured environments. For example, notice that in the derivation of the catching algorithm, we have not made any assumption about the dynamics underlying the ball's trajectory. For this reason, we are not limited to catching free flying objects, but might be able to tackle many tasks that require a smooth contact of the robot gripper with a moving (or, of course, stationary) object. Beyond eliminating the "guarded move" that precedes normal pick and place operations, this approach might be used as well to transfer workpieces in a fast and smooth fashion between robots or from and to other conveyors without stopping. Other traditional applications can be imagined as well. Many parts feeders might be replaced in favor of simply dropping parts or tools into the robot's workspace, where they would be caught smoothly. The juggling mirror algorithm is similarly independent of the particular underlying environmental dynamics (even though our knowledge thereof informs the choice of gains). One can imagine the possible role of mirror type algorithms in repetitive but not necessarily periodic environments—for example, in active suspension of wheeled vehicles.

The two-juggle poses a combinatorially complicated problem that our robot "solves" using a very modest switching rule. We presume that similar techniques might be employed to achieve stable and robust gait regulation in legged robots.[5] We also have a growing sense that similarly modest switching rules might provide solutions to more general "planning problems" where there are many tasks that would be better done all at once but for which the robot's limited means necessitate a prioritized sequence.

4.2. Theoretical Questions Arising From the Present Work

The successful experiments reported here, along with the prospects for useful extensions described previously, convince us that a much more systematic theoretical treatment of these ideas is in order. The chief burden of work may be divided into two related categories. Both aim at replacing the present intuitive controller design with an automatic—or at least better guided—synthesis procedure.

The first category concerns the effect of the synthesis. Although the possibility of correctness proofs represents

5. The catch seems to have its application to legged robots as well. For example, consider the problem of landing on the ground after a jump with all feet touching at zero impact velocity. This capability is crucial when moving on slippery ground. It also helps to minimize the energy lost at impact and to reduce mechanical stress in the leg at touchdown.

one of the central virtues of programming in the geometric style displayed here, the gulf between what works experimentally and what we can show formally is still quite large. For a one-degree-of-freedom model of the one-juggle, we are able to propose a formal explanation of why the mirror law works using the language of dynamic systems theory (Buehler and Koditschek 1990). Furthermore, we are able to corroborate experimentally certain strong predictions the theory makes about the juggling behavior (Buehler and Koditschek 1990). However, the extent to which this theory can be extended to more realistic models of the one-juggle is not yet clear. Moreover, while it seems clear that the two-juggle should also be represented and studied using the language of dynamic systems theory, we have not yet understood how to do so. Most fundamentally, we do not presently know what formal framework might be most appropriate for studying such one-shot tasks as the catch.

The second category concerns the origin of the synthesis. We have already acknowledged in Section 2.4.1, and have tried to make plain throughout the article, that the mirror laws we employ for juggling and catching are derived in an ad hoc manner according to intuition. Even in the case of the one-juggle for which a proof of correctness is available, it is not yet clear how to place a priori design constraints on the class of mirror functions to achieve the desired behavior. Clear design principles are even farther removed from the construction of our two-juggle and catching laws, as might be expected in the absence of correctness results for these behaviors.

There is every reason to believe that further informal extension of these ideas to richer and more complex tasks will succeed in practice. More than likely, such extensions will help guide the process of theoretical explanation. We are confident that at some point a formal rendering of these techniques will surpass the limits of intuition and yield general synthesis methods for dynamically dexterous tasks whose complexity defies heuristic approaches.

Acknowledgments

This work has been supported in part by PMI Motion Technologies, INMOS (a member of the SGS-THOMPSON Microelectronics Group), and the National Science Foundation under a Presidential Young Investigator Award held by the second author.

References

Aboaf, E. W., Drucker, S. M., and Atkeson, C. G. 1989. Task-level robot learning: Juggling a tennis ball more accurately. *Proc. IEEE Int. Conf. Robotics and Automation*, Scottsdale, AZ, pp. 1290–1295.

Andersson, R. L. 1989 Understanding and applying a robot ping-pong player's expert controller. *Proc. IEEE Int. Conf. Robotics and Automation*, Scottsdale, AZ, pp. 1284–1289.

Buehler, M. 1990. *Robotic Tasks With Intermittent Dynamics*. Ph.D. thesis, Yale University. UMI (Ann Arbor, MI) Publ. no. 9035329.

Buehler, M., and Koditschek, D. E. 1990. From stable to chaotic juggling: Theory, simulation, and experiments. *Proc. IEEE Int. Conf. Robotics and Automation*, Cincinnati, OH, pp. 1976–1981.

Buehler, M., Koditschek, D. E., and Kindlmann, P. J. 1988. A one degree of freedom juggler in a two degree of freedom environment. *Proc. IEEE/RSJ Conf. Intelligent Systems and Robots*, Tokyo, Japan, pp. 91–97.

Buehler, M., Koditschek, D. E., and Kindlmann, P. J. 1989. A simple juggling robot: Theory and experimentation. In Hayward, V., and Khatib, O. (eds.): *Experimental Robotics I*. New York: Springer-Verlag, pp. 35–73.

Buehler, M., Koditschek, D. E., and Kindlmann, P. J. 1990. A family of robot control strategies for intermittent dynamical environments. *IEEE Control Systems Magazine* 10(2):16–22.

Buehler, M., Whitcomb, L. L., Levin, F., and Koditschek, D. E. 1989. A distributed message passing computational and I/O engine for real-time motion control. *American Control Conference*, Pittsburgh, PA, pp. 478–483.

Erdmann, M., and Mason, M. T. 1986. An exploration of sensorless manipulation. *Proc. IEEE Int. Conf. Robotics and Automation*. San Francisco, CA, pp. 1569–1574.

Koditschek, D. E. 1986. Automatic planning and control of robot natural motion via feedback. In Narendra, K. S. (ed.): *Adaptive and Learning Systems: Theory and Applications*. New York: Plenum, pp. 389–402.

Koditschek, D. E. 1987. Exact robot navigation by means of potential functions: Some topological considerations. *Proc. IEEE Int. Conf. Robotics and Automation*. Raleigh, NC, pp. 1–6.

Koditschek, D. E., and Buehler, M. 1991. Analysis of a simplified hopping robot. *Int. J. Robot. Res.* 10(6):587–605.

Koditschek, D. E., and Rimon, E. 1990. Robot navigation functions on manifolds with boundary. *Adv. Applied Mathematics* 11:412–442.

Mason, M. T. 1986. Mechanics and planning of manipulator pushing operations. *Int. J. Robot. Res.* 5(3):53–71.

McGeer, T. 1989. Passive bipedal running. Technical report IS-TR-89-02. Centre for Systems Science, Simon Fraser University.

McGeer, T. 1990. Passive dynamic walking. *Int. J. Robot. Res.* 9(2):62–82.

Raibert, M. H. 1986. *Legged Robots That Balance.* Cambridge, MA: MIT Press.

Rimon, E., and Koditschek, D. E. 1991. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. *Trans. Am. Math. Soc.* 327(1):71–115.

Taylor, R. H., Mason, M. T., and Goldberg, K. Y. 1987. Sensor-based manipulation planning as a game with nature. In *Int. Symp. Robotics Research.* Cambridge, MA: MIT Press.

Wang, Y. 1989a. *Dynamic Analysis and Simulation of Mechanical Systems with Intermittent Constraints.* Ph.D. thesis, Carnegie-Mellon University, Pittsburgh, PA.

Wang, Y. 1989b. Mechanics and planning of collisions in robotic manipulation. *Proc. IEEE Int. Conf. Robotics and Automation,* Scottsdale, AZ, pp. 478–483.

Wang, Y., and Mason, M. T. 1987. Modeling impact dynamics for robotics operations. *Proc. IEEE Int. Conf. Robotics and Automation,* San Francisco, CA, pp. 678–685.