Preliminary Experiments in Spatial Robot Juggling

A. A. Rizzi and D. E. Koditschek *

Center for Systems Science, Yale University
New Haven, CT 06520-1968

Abstract

In a continuing program of research in robotic control of intermittent dynamical tasks, we have constructed a three degree of freedom robot capable of "juggling" a ball falling freely in the earth's gravitational field. This work is a direct extension of that previously reported in [7, 3, 5, 4]. The present paper offers a comprehensive description of the new experimental apparatus and a brief account of the more general kinematic, dynamical, and computational understanding of the previous work that underlie the operation of this new machine.

1 Introduction

In our continuing research on dynamically dexterous robots we have recently completed the construction of a second generation juggling machine. Its forebear, a mechanically trivial system that used a single motor to rotate a bar parallel to a near-vertical frictionless plane was capable of juggling one or two pucks sensed by a grid of wires into a specified stable periodic motion through repeated batting [7, 3, 6]. In this second generation machine, a three degree of freedom direct drive arm (Figure 1) relies on a field rate stereo vision system to bat an artificially illuminated ping-pong ball into a specified periodic vertical motion. Despite the considerably greater kinematic, dynamical, and computational complexity of the new machine, its principle of operation represents a straightforward generalization of the ideas introduced in the previous planar study. Moreover, its empirical performance reveals strong robust stability properties similar to those predicted and empirically demonstrated in the original machine. The arm will successfully bring a wide diversity of initial conditions to the specified periodic veritical motion through repeated batting. Recovery from significant perturbations introduced by unmodeled external forces applied during the ball's free flight is quite reliable. We typically log thousands and thousands of successive impacts before a random imperfection in the wooden paddle drives the ball out of the robot's workspace.

The work presented here represents the first application of the controllers developed in [3] to a multi-axis robot, and demonstrates the capabilities of the Büghler arm and the Cyclops vision system. Both of these systems have been developed at the Yale University Robotics Laboratory to facilitate our investigations into robot control of intermittent dynamical tasks. Thus, the present paper takes on the strong aspect of displaying the fruits of previous research. We offer a comprehensive description of the components of the new apparatus in Section 2. Section 3

*This work has been supported in part by the Superior Electric Corporation, SGS Thomson-INMOS Corporation and the National Science Foundation under a Presidential Young Investigator Award held by the second author.

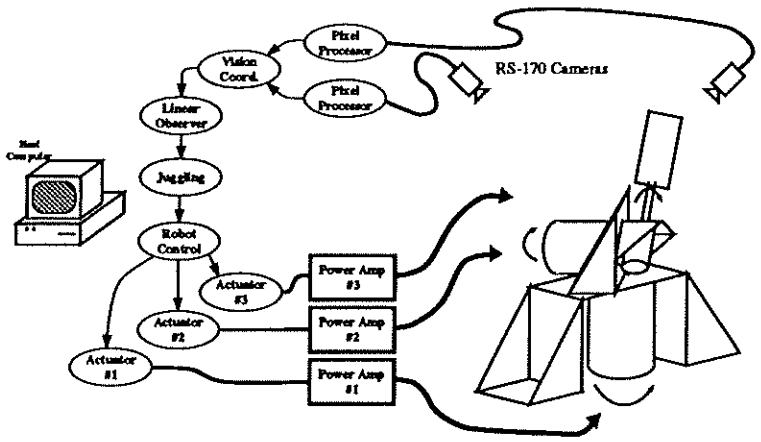

Figure 1: The Yale Spatial Juggling System

briefly reviews the notion of a mirror algorithm, the sole repository of all "juggling intelligence" in our system, and displays its generalization to the present kinematics. Section 4 provides a system level "tour" describing the manner in which the machine's physical and computational architecture is coordinated to realize the desired task. The paper concludes with a brief outline of our near-term future research directions.

2 Juggling Apparatus

This section describes the constituent pieces of our juggling machine. The system, pictured in Figure 1, consists of three major components: an environment (the ball); the robot; and an environmental sensor (the vision system). We now describe in fairly specific terms the hardware underlying each component and propose a (necessarily simplified) mathematical model in each case that describes its properties in isolation.

2.1 Environment: Striking a Ball in Flight

The two properties of the ball relevant to juggling are its flight dynamics (behavior while away from the paddle), and its impact dynamics (how it interacts with the paddle/robot). For simplicity we have chosen to model the ball's flight dynamics as a point mass under the influence of gravity. This gives rise to the flight model

$$\ddot{b} = \tilde{a}, \tag{1}$$

where $b \in \mathcal{B} = \mathbb{R}^3$, and $\tilde{a} = (0, 0, -\gamma)^T$ is the acceleration vector experienced by the ball due to gravity.

Suppose a ball with trajectory $b(t)$ collides with the paddle in robot configuration $q \in \mathcal{Q}$ at some point, p on the paddle which has a linear velocity v. Letting $\mathcal{T} \triangleq \mathcal{B} \times \mathcal{Q}$ denote the total configuration space of the problem, we seek a description of how the ball's phase, $(b, \dot{b}) \in T\mathcal{B}$ is changed by the robot's phase, $(q, \dot{q}) \in T\mathcal{Q}$ at an impact.

As in [7, 6] we will assume that the components of the ball's velocity tangent to the paddle at instant of contact are unchanged, while the normal component is governed by the simplistic (but standard [14]) coefficient of restitution law. For some $\alpha \in [0, 1]$ this impact model can be

expressed as

$$(\dot{b}'_n - v'_n) = -\alpha(\dot{b}_n - v_n), \tag{2}$$

where $\dot{b}'_n$ and v'_n denote the normal components of the ball and paddle velocities immediately after impact, while $\dot{b}_n$ and v_n are the velocities prior to impact. Assuming that the paddle is much more massive than the ball (or that the robot has large torques at its disposal), we conclude that the velocity of the paddle will remain constant throughout the impact ($v' = v$). It follows that the coefficient of restitution law can now be re-written as

$$\dot{b}'_n = \dot{b}_n + (1+\alpha)(v_n - \dot{b}_n). \tag{3}$$

and, hence,

$$\dot{b}' = \dot{b} + (1+\alpha)nn^{\mathrm{T}}(v - \dot{b}), \tag{4}$$

where n denotes the unit normal vector to the paddle.

2.2 Robot Kinematics: An Almost Spherical Arm

At the heart of the juggling system resides a three degree of freedom robot — the Bühgler Arm[1] — equipped at its end effector with a paddle. The revolute joints give rise to the familiar difficulties in computing and analyzing the robot's inverse kinematics. Moreover, as in our earlier work, the presence of revolute kinematics introduces a strongly nonlinear component to the "environmental control system", an abstract discrete dynamical system with respect to which we find it effective to encode the juggling task.

The robot kinematics relevant to the task of batting a ball relates the machine's configuration to the normal vector at a point in its paddle. In order to represent this formally we parametrize the paddle's surface geometry. Let $\bar{p}$ represent (in homogeneous coordinates) a planar transformation taking points in the unit box, $\mathcal{S} \triangleq [0,1] \times [0,1]$ diffeomorphically onto the paddle's (finite) surface area expressed with respect to the gripper frame, $\mathcal{F}_g$. Associated with each point on the paddle's surface, $\bar{p}(s)$ is the unit normal, $\bar{n}(s)$, again, the homogeneous coordinate representation of the vector with respect to $\mathcal{F}_g$. The paddle's "Gauss map" [15] is now parametrized as [2]

$$N : \mathcal{S} \to \mathcal{N}(3) : s \mapsto [\bar{n}(s), \bar{p}(s)]; \qquad \mathcal{N}(3) = \mathbb{R}^3 \times S^2. \tag{5}$$

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

$$G : \tilde{\mathcal{Q}} \to \mathcal{N}(3) : (q,s) \mapsto [n(q,s), p(q,s)] = H(q)N(s); \qquad \tilde{\mathcal{Q}} = \mathcal{Q} \times \mathcal{S}. \tag{6}$$

At the cost of a little more notation, it will prove helpful to define the projections,

$$\pi_{\mathcal{Q}}(q,s) = q; \qquad \pi_{\mathcal{S}}(q,s) = s.$$

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

$$v = \sum_{i=1}^{dim\mathcal{Q}} \dot{q}_i D_{q_i} H\,(q)p(s) = D_q p\,\dot{q} = Dp\,\Pi_{\mathcal{Q}}\dot{q}; \qquad \Pi_{\mathcal{Q}} \triangleq [D\pi_{\mathcal{Q}}]^{\mathrm{T}} = \begin{bmatrix} I_{dim\mathcal{Q}\times dim\mathcal{Q}} \\ 0_{dim\mathcal{S}\times dim\mathcal{Q}} \end{bmatrix}, \tag{7}$$

[1] Pronounced *byoōg'–ler*.

[2] The appearance of n in (4) suggests that it is really a force vector, thus we will define the possible normal vectors at a prescribed spatial point, $\mathcal{N}(3) \triangleq \mathbb{R}^3 \times S^2$ as lying in the space dual to the infinitesimal velocities at the point, $\mathcal{N}(3) \subset T^*\mathcal{B}$.

Additionally lying in the total configuration space is the contact submanifold, $\mathcal{C}$, — the set of ball/robot configurations where the ball is in contact with the paddle — given by

$$\mathcal{C} \triangleq \{(b,q) \in \mathcal{T} : \exists s \in \mathcal{S}, b = p(q,s)\}.$$

which is evidently the only place that the normal appearing in (4) becomes relevant. Since $\bar{p}$ is one-to-one by assumption there is a map $s_c : \mathcal{C} \to \mathcal{S}$ such that

$$b = p(q, s_c(b,q)). \tag{8}$$

Combining (7), and (8) we may now rewrite the impact event (4) in terms of a "collision map" $\mathbf{c} : T\mathcal{C} \to T\mathcal{B}$, as

$$\begin{aligned} \dot{b}' &= \dot{b} + \mathbf{c}(b,\dot{b},q,\dot{q}) \\ \mathbf{c}(b,\dot{b},q,\dot{q}) &\triangleq -(1+\alpha) n(q,s_c(b,q)) n^{\mathrm{T}}(q,s_c(b,q)) \left(\dot{b} - Dp\, \Pi_{\mathcal{Q}} \dot{q}\right). \end{aligned} \tag{9}$$

Choosing a gripper frame, $\mathcal{F}_g$, for the Bühgler Arm depicted in Figure 1 located at the base of the of the paddle (the point of intersection of the second and third joints) whose x-axis is aligned with the paddle's normal and whose z-axis is directed along the paddle's major axis, we have

$$N(s) = \left[\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} d_g \\ s_1 \\ s_2 \\ 1 \end{bmatrix} \right],$$

and we will artificially set $s_1 = 0, s_2 = s \in [\underline{s}, \bar{s}]$ for reasons to be made clear below. The frame transformation, $H(q)$, is developed in [13], and yields a forward kinematic map of the form

$$G(q,s) = [n(q,s), p(q,s)] \tag{10}$$

$$n(q,s) = \begin{bmatrix} \cos(q_1)\cos(q_2)\cos(q_3) - \sin(q_1)\sin(q_3) \\ \cos(q_2)\cos(q_3)\sin(q_1) + \cos(q_1)\sin(q_3) \\ -\cos(q_3)\sin(q_2) \\ 0 \end{bmatrix}$$

$$p(q,s) = \begin{bmatrix} -(\sin(q_1)d_2) + (\cos(q_1)\cos(q_2)\cos(q_3) - \sin(q_1)\sin(q_3))\, d_g + \cos(q_1)\sin(q_2)s_2 \\ \cos(q_1)d_2 + (\cos(q_2)\cos(q_3)\sin(q_1) + \cos(q_1)\sin(q_3))\, d_g + \sin(q_1)\sin(q_2)s_2 \\ -(\cos(q_3)\sin(q_2)d_g) + \cos(q_2)s_2 \\ 1 \end{bmatrix}.$$

Analysis of the jacobian of p shows that it is rank three away from the surface defined by

$$\delta_p(q,s) \triangleq (s_2^2 + \cos^2(q_2)\cos^2(q_3))(\sin^2(q_2) + \cos^2(q_3)) = 0,$$

thus away from $\delta_p = 0$ we can define $Dp^\dagger$, the right inverse of Dp, and the workspace is now given by $\mathcal{W} \triangleq p(\tilde{\mathcal{Q}} - \mathcal{H})$ where

$$\mathcal{H} \triangleq \left\{\tilde{q} \in \tilde{\mathcal{Q}} : \delta_p(\tilde{q}) = 0\right\}$$

Finally the inverse kinematic image of a point $b \in \mathcal{W}$ may be readily computed as

$$p^{-1}(b) = \begin{bmatrix} \mathrm{ArcTan2}(-b_2,-b_1) + \mathrm{ArcSin}\left(\frac{\sin(q_3)d_2}{\sqrt{b_1^2+b_2^2}}\right) \\ -\frac{\pi}{2} + \mathrm{ArcTan2}\left(b_3, \sqrt{b_1^2+b_2^2-\sin(q_3)d_2^2}\right) - \mathrm{ArcSin}\left(\frac{\cos(q_3)d_g}{\sqrt{b^Tb-\sin^2(q_3)d_2^2}}\right) \\ q_3 \\ 0 \\ \sqrt{b^Tb - \sin^2(q_3)d_2^2 - \cos^2(q_3)d_g^2} \end{bmatrix}; \; q_3 \in S^1, \tag{11}$$

with the freely chosen parameter, q_3, describing the one dimensional set of robot configurations capable of reaching the point b. Having simplified the kinematics via the artificial joint constraint, $s_1 \equiv 0$, the paddle contact map may simply be read off the inverse kinematics function,

$$s_c(b,q) = \pi_{\mathcal{S}} \circ p^{-1}(b) = \begin{bmatrix} 0 \\ \sqrt{b^T b - \sin^2(q_3) d_2^2 - \cos^2(q_3) d_g^2} \end{bmatrix}.$$

2.3 Sensors: A Field Rate Stereo Vision System

Two RS-170 CCD television cameras with 1/2000sec. electronic shutters constitute the "eyes" of the juggling system. In order to make this task tractable we have simplified the environment the vision system must interpret. The "world" as seen by the cameras contains only one white ball against a black background. The CYCLOPS vision system, described in Section 2.4, allows the straightforward integration of these cameras into the larger system.

Following Andersson's experience in real-time visual servoing [1] we employ the result of a first order moment computation applied to a small window of a threshold-sampled (that is, binary valued) image of each camera's output. Thresholding, of course, necessitates a visually structured environment, and we presently illuminate white ping-pong balls with halogen lamps while putting black matte cloth cowling on the robot, floor, and curtaining off any background scene.

2.3.1 Triangulation

In order to simplify the construction of a trangulator for this vision system, we have employed a simple projective camera model. Let $\mathcal{F}_c$ be a frame of reference whose origin is at the focal point and whose z-axis is directed toward the image plane of this camera. Let ${}^c p = [p_x, p_y, p_z, 1]^T$ denote the homogeneous representation with respect to this frame of some spatial point. Then the camera, with focal length f, transforms this quantity as

$${}^c u = f \begin{bmatrix} p_x/p_z \\ p_y/p_z \\ 1 \\ 0 \end{bmatrix} \triangleq \mathbf{p}_f({}^c p). \tag{12}$$

Here, ${}^c u \in \mathbb{R}^4$ is the homogeneous vector, with respect to $\mathcal{F}_c$, joining the oragin of $\mathcal{F}_c$ to the image plane coordinates of ${}^c p$. Thus, for a camera whose position and orientation relative to the base frame, $\mathcal{F}_0$ are described by the homogeneous matrix ${}^0 H_c$, the projection of a point, ${}^0 p$ is

$$u = \mathbf{p}_f({}^c H_0 \, {}^0 p).$$

Given two such cameras separated in space, whose frames of reference with respect to $\mathcal{F}_0$ are represented by ${}^0 H_l$ and ${}^0 H_r$, it is straightforward to derive a *triangulation function*, $\mathbf{p}^\dagger$, capable of reconstructing the spatial location of a point, given its projection in both images. In particular if projection onto the *right* and *left* image planes is given by

$${}^r u_r = \mathbf{p}_{\mathbf{f_r}}({}^r \mathbf{H_0} \; {}^0 \mathbf{p}),$$

and

$${}^l u_l = \mathbf{p}_{\mathbf{f_l}}({}^l \mathbf{H_0} \; {}^0 \mathbf{p})$$

respectively, a (by no means unique) triangulation function is given by

$$\mathbf{p}^\dagger({}^r u_r, {}^l u_l) \triangleq \frac{1}{2} \left({}^0 H_r \, (o + t_r \, {}^r u_r) + {}^0 H_l \, (o + t_l \, {}^l u_l) \right), \tag{13}$$

where

$$\begin{bmatrix} t_r \\ t_l \end{bmatrix} \triangleq \left(C^T C\right)^{-1} C^T (o - {}^r H_0 \, {}^0 H_l \, o)$$

and

$$o \triangleq [0 \quad 0 \quad 0 \quad 1]^T ; \; C \triangleq [-{}^r u_r \mid {}^r u_l] ; \; {}^r u_l = {}^r H_0 \, {}^0 H_l \, {}^l u_l .$$

This amounts to finding the midpoint of the biperpendicular line segment joining the two lines defined by ${}^r u_r$ and ${}^r u_l$. Note that there is considerable freedom in the definition of $\mathbf{p}^\dagger$ since it maps a four dimensional space (the two image plane vectors) onto a space of dimension three (ball position).

Finally it is worth noting that although the implementation of a triangulation system of this type is simple, the measurement of the parameters required for its construction is quite difficult. A short description of the automatic method of calibration we have chosen to use in the juggling system can be found in Appendix A.

2.3.2 Signal Processing

In practice it is necessary to associate a signal processing system with the sensor to facilitate interpretation of the data. For the vision system in use here, sensor interpretation consists of estimating the ball's position and velocity, correcting for the latency of the vision system, and improving the data rate out of the sensor system – the 60 Hz of the vision system is far below the bandwidth of the robot control system.

Given reports of the ball's position from the triangulator it is straightforward to build a linear observer for the full state — positions and velocities — since the linear dynamical system defined by (1) is observable. In point of fact, it is not the ball's position, b_n, which is input to the observer, but the result of a series of computations applied to the cameras' image planes, and this "detail" comprises the chief source of difficulty in building cartesian sensors of this nature.

2.4 Controller: A Flexibly Reconfigurable Computational Network

All of the growing number of experimental projects within the the Yale University Robotics Laboratory are controlled by widely various sized networks of Transputers produced by the INMOS division of SGS-Thomson. Pricing and availability of both hardware and software tools make this a natural choice as the building block for what we have come to think of as a computational "patch panel." The recourse to parallel computation considerably boosts the processing power per unit cost that we can bring to bear on any laboratory application. At the same time the serial communication links have facilitated quick network development and modification.

The choice of the INMOS product line represents a strategy which standardizes and places the burden of parallelism — inter-processor communications support, software, and development environment — around a commercial product, while customizing the computational "identity" of particular nodes by recourse to special purpose hardware. We provide here a brief sketch of the XP/DCS family of boards, a line of I/O and memory customized Transputer nodes developed within the Yale Robotics Lab and presently employed in all our control experiments. The backbone of this system is the XP/DCS CPU, providing a transputer and bus extender. By coupling an XP/DCS to an IO/MOD a computational node can be customized for interfacing to moderate bandwidth hardware. Similarly joining up to eight XP/DCSs to individual CYCLOPS frame memory boards, then ganging these together under a single video digitizer forms a programmable field rate monocular vision system.

The XP/DCS processor The XP/DCS (produced by Evergreen Designs) was designed in conjunction with the Yale Robotics Laboratory in 1987 [9] in order to meet both the computational and I/O requirements presented by robotic tasks. The board is based on the INMOS T800 processor, a 32 bit scalar processor capable of 10 MIPS and 1.5 MFLOP (sustained) with four bidirectional 20MHz DMA driven communication links and 4 Kbytes of internal (1 cycle) RAM. The processor is augmented with an additional 1-4 Mbytes of dynamic RAM (3 cycle), and an I/O connector which presents the T800's bus to a daughter board.

IO/MOD The IO/MOD (also produced by Evergreen Designs) allows an XP/DCS to "communicate" with custom hardware in a simple fashion. In order to properly implement the ideal "processing path panel" it is essential that the integration of new sensors and actuators be simple and fast. The IO/MOD augments an XP/DCS by providing a 32 bit latched bidirectional data bus, six 4 bit wide digital output ports, and eight digital input signals, all of which are mapped directly into the memory space of the T800.

CYCLOPS Vision System Much like the IO/MOD the CYCLOPS system has been designed to augment a set of XP/DCS boards for a particular sensing task — vision. In actuality there are three major components to the vision system [8]:

Digitizer: Digitizes an incoming RS-170 video signal and outputs it in digital form over a pixel bus.

Filter: A filter board capable of performing real-time 2D convolution on an image may be placed on the pixel bus.

Frame Memory: In much the same fashion as the IO/MOD the CYCLOPS Memory Board augments an XP/DCS with 128 Kbytes of video memory. By associating up to eight memory boards with a pixel bus it becomes easy to construct a real-time parallel processing vision system.

3 Juggling Algorithm

This section offers a brief review of how the juggling analysis and control methodology originally introduced for the planar system [7] may be extended in a straightforward manner to the present apparatus. After introducing the "environmental control system," an abstract dynamical system formed by composing the free flight and impact models, it becomes possible to encode an elementary dexterous task, the "vertical one juggle," as an equilibrium state — a fixed point. A simple computation reveals that every achievable vertical one juggle can be made a fixed point, and conversely, the only fixed points of the environmental control system are those that encode a vertical one juggle. Leapfrogging the intermediate linearized analysis of our planar work [3], we then immediately follow with a description of a continuous robot reference trajectory generation strategy, the "mirror law," whose implementation gives rise to the juggling behavior.

3.1 Task Encoding

Denote by $\mathcal{V}$ the robot's choices of impact normal velocity for each workspace location. Suppose that the robot strikes the ball in state $w_j = (b_j, \dot{b}_j)$ at time s with a velocity at normal $v_j = (q, \dot{q}) \in \mathcal{V}$ and allows the ball to fly freely until time $s + t_j$. According to (9) derived in the previous section, composition with time of flight yields the "environmental control system"

$$w_{j+1} = f(w_j, v_j, t_j) \triangleq A_{t_j} w_j + a_{t_j} + \begin{bmatrix} t_j \mathbf{c}(w_j, v_j) \\ \mathbf{c}(w_j, v_j) \end{bmatrix}, \tag{14}$$

that we will now be concerned with as a controlled system defined by the dynamics

$$f : T\mathcal{B} \times \mathcal{V} \times \mathbb{R} \to T\mathcal{B},$$

with control inputs in $\mathcal{V} \times \mathbb{R}$ (v_j and t_j).

Probably the simplest systematic behavior of this system imaginable (beyond the ball at rest on the paddle), is a periodic vertical motion of the ball. In particular, we want to be able to specify an arbitrary "apex" point, and from arbitrary initial conditions, force the ball to attain a periodic trajectory which passes through that apex point. This corresponds exactly to the choice of a fixed point, w^*, in (14), of the form

$$w^* = \begin{bmatrix} b^* \\ \dot{b}^* \end{bmatrix}; \quad b^* \in \mathbb{R}^3; \quad \dot{b}^* = \begin{bmatrix} 0 \\ 0 \\ \nu \end{bmatrix}; \quad \nu \in \mathbb{R}^3, \tag{15}$$

denoting a ball state-at-impact occurring at a specified location, with a velocity which implies purely vertical motion and whose magnitude is sufficient to bring it to a pre-specified height during free flight. Denote this four degree of freedom set of vertical one-juggles by the symbol $\mathcal{J}$.

The question remains as to which tasks in $\mathcal{J}$ can be achieved by the robot's actions. In particular we wish to determine which elements of $\mathcal{J}$ can be made fixed points of (14). Analysis of the fixed point conditions imposes the following requirements on w^*:

$$\dot{b}^* = \frac{1}{2}\lambda\tilde{a} \tag{16}$$

and for some $(q, \dot{q}) \in TQ$ and $\lambda \in \mathbb{R}^+$,

$$p(q, s_c(b^*, q)) = b^* \text{ and } \mathbf{c}(b^*, \dot{b}^*, q, \dot{q}) = -\lambda\tilde{a}. \tag{17}$$

Every element of $\mathcal{J}$ satisfies (16), since this simply enforces that the task be a vertical one-juggle. For the Bühgler Arm (17) necessitates that n be aligned with $\tilde{a}$ so as not to impart some horizontal velocity on the ball. From (10) it is clear that this will only be the case when $q \in \mathcal{Q}^*$, where

$$\mathcal{Q}^* \triangleq \{q \in \mathcal{Q} : \cos(q_3)\sin(q_2) = -1\}.$$

Thus, we can conclude that only those elements of $\mathcal{J}$ satisfying the condition $b^* \in p(\mathcal{Q}^*)$ will be fixable. In particular, $\mathcal{Q}^*$ corresponds to the paddle being positioned parallel to the floor, and thus $p(\mathcal{Q}^*)$ is an annulus above the floor, as is intuitively expected.

This simple analysis now furnishes the means of tuning the spatial locus and height of the desired vertical one juggle. The fixed-point input satisfying these conditions, u^*, is given by

$$u^* = \begin{bmatrix} \frac{2}{\gamma}\|\dot{b}^*\| \\ \left(\frac{\alpha-1}{\alpha+1}\right) Dp^{\dagger}\dot{b}^* \end{bmatrix}.$$

3.2 Controlling the Vertical One-Juggle via a Mirror Law

Say that the abstract feedback law for (14), $g : \mathcal{W} \to \mathcal{V} \times \mathbb{R}$, is a verticle one-juggle strategy if it induces a closed loop system,

$$f_g(w) = f(w, g(w)), \tag{18}$$

for which $w^* \in \mathcal{J}$ is asymptotically stable fixed point. For our original planar machine [3] it was shown that the linearization of the analogous system to (14) was controllable around every

vertical one juggle task. A similar analysis has not yet been completed for the Bühgler Arm, although a similar result is expected. Experiments with the planar system revealed that the linearized perspective was inadequate: the domain of attraction resulting from locally stabilizing linear state feedback was smaller than the resolution of the robot's sensors [3].

Instead, in [7] a rather different juggling strategy was proposed that implicitly realized an effective discrete feedback policy, g, by requiring the robot to track a distorted reflection of the ball's continuous trajectory. This policy, the "mirror law," may be represented as a map $m : T\mathcal{B} \to \mathcal{Q}$, so that the robot's reference trajectory is determined by

$$q(t) = m(w(t)).$$

For a one degree of freedom environment it is not hard to show that this policy results in a (essentially) globally asymptotically stable fixed point [5]. For a two degree of freedom environment, we have shown that local asymptotic stability results [3]. The spatial analysis is in progress.

The juggling algorithm used in the present work is a direct extension of this "mirror" law to the spatial juggling problem. In particular begin by using (11) to define the the joint space position of the ball

$$\begin{bmatrix} \phi_b \\ \theta_b \\ \psi_b \\ s_b \end{bmatrix} \triangleq p^{-1}(b). \tag{19}$$

We now seek to express formulaically a robot strategy that causes the paddle to respond to the motions of the ball in four ways:

(i) $q_{d1} = \phi_b$ causes the paddle tracks under the ball at all times.

(ii) The paddle "mirrors" the vertical motion of the ball through the action of θ_b on q_{d2} as expressed by the original planar mirror law [7].

(iii) Radial motion of the ball causes the paddle to raise and lower, resulting in the normal being adjusted to correct for radial deviation in the ball position.

(iv) Lateral motion of the ball causes the paddle to roll, again adjusting the normal so as to correct for lateral position errors.

To this end, define the ball's *vertical energy* and *radial distance* as

$$\eta \triangleq \gamma b_z + \frac{1}{2}\dot{b}_z^2 \quad \text{and,} \quad \rho_b \triangleq \sin(\theta_b)s_b \tag{20}$$

respectively. The complete mirror law combines these two measures with a set point description ($\bar{\eta}$, $\bar{\rho}$, and $\bar{\phi}$) to form the function

$$q_d = m(w) \triangleq \begin{bmatrix} \overbrace{\phi_b}^{\text{(i)}} \\ \underbrace{-\frac{\pi}{2} - (\kappa_0 + \kappa_1(\eta - \bar{\eta}))\left(\theta_b + \frac{\pi}{2}\right)}_{\text{(ii)}} + \underbrace{\kappa_{00}(\rho_b - \bar{\rho}_b) + \kappa_{01}\dot{\rho}_b}_{\text{(iii)}} \\ \underbrace{\kappa_{10}(\phi_b - \bar{\phi}_b) + \kappa_{11}\dot{\phi}_b}_{\text{(iv)}} \end{bmatrix}. \tag{21}$$

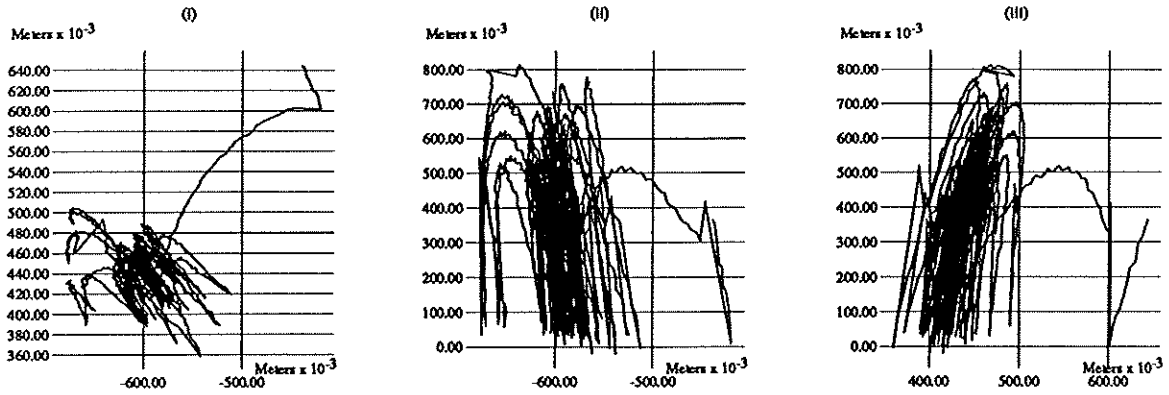

Figure 2: One-Juggle ball trajectory: (i) X-Y projection (ii) X-Z projection and (iii) Y-Z projection.

For implementation, the on-line reference trajectory formed by passing the ball's state trajectory, $w(t)$, through this transformation must be passed to the robot tracking controller. As described in Section 4.4, the high performance inverse dynamics tracking schemes that we presently employ require the availability of a target velocity and acceleration profile as well. By design $m(w)$ is differentiable and the time derivatives of w are known – at least away from an impact event. Thus, denoting by

$$F(w) \triangleq \begin{bmatrix} w_2 \\ \tilde{a} \end{bmatrix}$$

the spatial vector field corresponding to the ball's continuous free flight dynamics (1), we know that $q_d(t) = m(w(t))$ implies

$$\dot{q}_d = Dm\, F$$

and

$$\ddot{q}_d = Dm\, DF\, F + [F \otimes I]^{\mathrm{T}}\, D^2 m\, F$$

In practice, these terms are computed symbolically from (21) and F.

We have succeeded in implementing the one-juggle task as defined above on the Bühgler arm. The overall performance of the constituent pieces of the system – vision module, juggling algorithm, and robot controller – have each been outstanding, allowing for performance that is gratifyingly similar to the impressive robustness and reliability of the planar juggling system. We typically record thousands of impacts (hours of juggling) before random system imperfections (electrical noise, paddle inconsistencies) result in failure. Figure 2 shows the three projections of the ball's trajectory for a typical run. As can be seen the system is capable of containing the ball within roughly 15cm of the target position above the floor and 10cm of the target height of 60cm.

It is worth noting that in the x-z and y-z projections there is evidently spurious acceleration of the ball in both the x and y directions. Tracing this phenomenon through the system confirmed an earlier suspicion; our assumption that gravity is exactly aligned with the axis of rotation of the base motor is indeed erroneous. Correction of this calibration error requires the addition of trivial workspace cues (a plumb bob) to allow the direction of the gravitational force to be calibrated along with the remainder of the system. This correction is now in progress.

4 The Yale Spatial Juggling System

This section describes how we have combined the components of Section 2 to produce a coordinated juggling robot system. An engineering block diagram for this system is depicted in Figure 3. Its implementation in a network of XP/DCS nodes is depicted in Figure 4. The juggling algorithm these diagrams realize is a straightforward application of contemporary robot tracking techniques to the mirror law presented in Section 3 as driven by the output of the vision system. Thus, there is no explicit pre-planning of robot motions. Instead, the ball's natural motion as perceived through the stereo vision system stimulates a "reflex" reaction in the robot that gives rise to intermittent collisions. In turn, these "geometrically programmed" collisions elicit the proper juggling behavior.

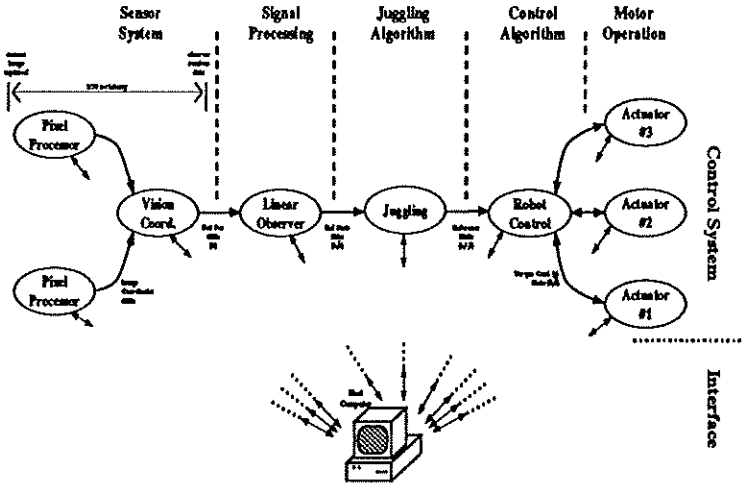

Figure 3: The Juggling System Controller Block Diagram.

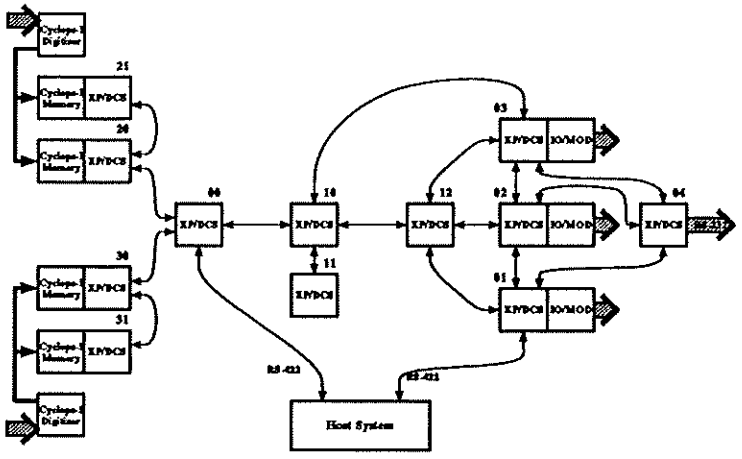

Figure 4: The Juggling System Network Diagram.

4.1 Vision: Environmental Sensing

The vision system must provide sufficient information regarding the state of the environment. In our present implementation we have so structured the visual environment as to make the task conceptually trivial and computationally tractable. In particular, the vision system need only extract the three dimensional position of a single white ball against a black background.

To perform even this apparently trivial task in real time we require two CYCLOPS vision systems — one for each "eye" — introducing a total of four nodes. In both Cyclops systems two memory boards, each with an associated XP/DCS processor, are attached to the digitizer. Computational limitations currently allow the system to process "windows" of less than 3000 pixels out of an image of 131,072 pixels (a 256×512 image).

Figure 5 depicts the flow of events on the five processors used for vision processing during an image cycle. The cycle begins with a field being delivered to one memory board associated with each camera (two of processors 20, 21, 30, 31). Note that these two events are guaranteed to occur simultaneously through the use of synchronized cameras. After the images have been deposited in the memory boards the centroid of the ball is estimated by calculating the first order moments over a window centered around the position of the ball, as determined by the most recent field (depicted by arrows in Figure 5). Upon completion, the image coordinate locations are passed to the neighboring pixel processors, for use in determining window location for the next field, and up to the triangulation process which is located on processor 00 of Figure 4. Once the low-level pixel processors have determined the ball location in a pair of images, stereo triangulation introduced in Section 2.3 locates the position of the ball in space with respect to a fixed reference coordinate system.

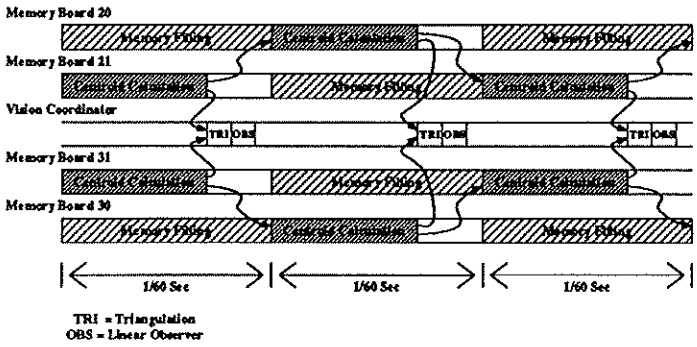

Figure 5: Timing for CYCLOPS vision system

4.2 Signal Processing

The signal processing block must "interpret" the environment and present it in a fashion that is acceptable for use by the remainder of the system. In this simple context, "interpretation" means producing good estimates of the ball's position and velocity at the current time. This is accomplished by connecting the output of the triangulator to a standard linear observer.

The timing diagram in Figure 5 shows that the vision block adds an unavoidable 1/30 sec. delay between the time an image is captured and the time a spatial position measurement has been formed. The ball's flight model presented in Section 2.1 is a sufficiently simple dynamical system that its future can be predicted with reasonable accuracy and, accordingly, a current estimate of the state is formed by integrating that delayed state estimate forward in time one field interval.

The data must now be passed to the interpolator. The task here involves stepping up the unacceptably slow data rate of the vision block (60 Hz): the time constant of the actuators is near 200 Hz. This interpolation stage uses the flight model of the ball, integrating the current state estimate of the ball forward over small time steps allows the data rate to be boosted from 60 Hz to 1 kHz.

This sequence of calculations is carried out on processor 00, (the coincidence with the triangulation process is incidental). The implementation of these signal processing functions is divided into two independent event driven processes. The first of these runs the observer and predictor, which are synchronized with the triangulation system and thereby with the cameras. Thus the sampling rate for the observer is set by the field rate of the cameras. The second process increases the effective data rate by scheduling itself at 1 msec intervals and updates its estimates of the ball state at each interval.

4.3 Juggling

The execution of the jugling algorithms developed in [7, 4] and Section 3 are implemented in this segment of the network. The evaluation of (21) is again carried out on processor 00, where both the high-level vision and signal processing are performed. The implementation consists of a single process which evaluates q_d, $\dot{q}_d$, and $\ddot{q}_d$ whenever new state information is received – yet another example of how we use naturally occuring events within the system to initiate computation. Since the input of this process is connected to the output of the interpolator the reference trajectory fed to the controller will be updated at the same rate as the output of the interpolator (1 kHz).

4.4 Robot Control

The geometric transformation introduced in Section 3, when applied to the joint space coordinate representation of the ball's flight, results in a desired profile of joint locations over time, $q_d(t)$. For the planar juggling robot we have shown that if the robot were to track exactly this "reference signal," then collisions with the ball would occur in such a fashion that the desired periodic motion is asymptotically achieved [7, 3]. We conjecture the same will be true in the present case. It now falls to the robot control block to ensure that the joint angles, $q(t)$, track the reference signal, $q_d(t)$.

We have implemented a large collection of feedback controllers on the Bühgler Arm, as reported in [17]. We find that as the juggling task becomes more complicated – e.g. simultaniously juggling two balls – that it becomes necessary to move to a more capable controller. We have had good succes with an inverse dynamics control law [17] of the form developed in [11],

$$\tau = C(q, \dot{q})\dot{q} + M(q)[\ddot{q}_d] + K_d(\dot{q} - \dot{q}_d) + K_p(q - q_d). \tag{22}$$

At the present time, all experiments have all been run with a robot control block that includes the three nodes (10, 11, and 12 in Figure 4). The model based portion of the control algorithms are implemented on processor 11 with update rate of 400 Hz, while the feedback portion (along with uninteresting housekeeping and logging functions) is implemented on 10 with update rate of 1 KHz, and a variety of message passing and buffering processes run on 12 which is really included in the network only for purposes of increasing the number of Transputer links converging on this most busy intersection of the entire network. There are two motivations for this seemingly lavish expenditure of hardware. First, in the interests of keeping the cross latency of the intrinsic data as low as possible [16], the increased number of links permits direct connectivity of the controller block with each node of the actuator block. Second, in the interests

of maintaining the "orthogonality" of the real-time and logging data flows, we gain a sufficient number of links at the controller to permit the dedicated assignment of logging channels back toward the user interface.

Nonblocking communication between this collection of processors is implemented through the use small input buffer processes. These buffer processes, which are situated at the inputs to each computational process, allow the various elements of the controller to receive data from each other and other elements of the system asynchronously. Thus far we have found that the reduction of effort required for software development and maintenance resulting from the use of this architecture has outweighed the performance costs imposed by the necessarily increased network latencies.

4.5 Actuator Management

The primary task associated with operating a particular actuator is to present a standard simple interface to the remainder of the system, thereby hiding the often unpleasant details of operating a particular motor from the remainder of the system. Since our interest is in developing systems which respect the dynamics of the robot, it follows that the actuator interface must receive torque commands and report state (position and velocity). In addition to this basic task we have found it desirable to include a strong system of safety interlocks at the lowest possible level, so as to ensure safe operation at all times.

5 Conclusion

This paper provides a comprehensive description of the generalization to three space of our earlier juggling work [7, 6]. To those familiar with that previous body of work, it may be apparent that the generalization begins to shed greater light on what went before. In particular, we are now able explicitly to show how the spatial juggling law and its planar predecessor are based on applying the mirroring notion of the "gedanken" line juggler [5] to the inverse kinematic image of the ball's workspace trajectory. In the case of the line juggler we had shown (using the "Singer-Guckenheimer" theory of unimodal return maps [5]) that the global stability mechanism this mirror law induces is a change of coordinates away from that which we had argued [10] stabilizes Raibert's [12] vertical hopping motion. In the case of the planar juggler, we were able to demonstrate that the mirror law results in local asymptotic stability, but, since an analytical expression for the effective closed loop dynamics (14) remained elusive, no global stability analysis has yet been possible.

The absence of an explicit closed form expression is a consequence of the transcendental system of equations that emerge when "PD" terms are added to the simple gedanken robot's mirror law. The same terms appear in the present algorithm (21). Thus, we do not believe that a global stability analysis will be any easier for our current spatial version of this idea. We believe, however, that an empirically transparent modification of these terms that re-expresses them with respect to the joint space coordinate system (as does the application of inverse kinematics to the ball's flight) may result in tractable closed loop dynamics and, thereby, the possibility of a global stability proof for the more interesting kinematics.

A Vision System Calibration

In the course of getting started with spatial juggling, we have been led to re-formulate a very attractive coordinated camera-arm calibration scheme originally proposed by Hollerbach [2]. At

calibration time, one supposes that some point on the robot's gripper (that we willl take to be the origin of the "tool" frame) is marked with a light reflecting material in such a fashion as to produce an unmistakable camera observation — a four vector, $c \in \mathbb{R}^4$ comprised of the two image plane measurements. The problem is to determine the kinematic parameters, $k \in \mathbb{R}^{8+3(m+1)}$, that characterize the chain as well as the relative camera frame relationship and camera focal lengths by comparing measured camera values with the joint space locations that produced them.

The Setting Denote by g_k, the forward kinematic transformation of the kinematic chain that expresses the robot's tip marking with respect to the base frame (that we take to be the frame of the "right" hand camers with no loss of generality). According to the Denavit-Hartenburg convention, the parameter vector, $(k_1, ..., k_{m+1}) \in \mathbb{R}^{3(m+1)}$, that characterizes this function appears in the form

$$g_k(q) = \left(\prod_{i=1}^{m+1} H_i(\theta_i) \right) \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}; \qquad H_i(\theta_i) \triangleq \exp \left\{ \theta_i \sum_{j=1}^{3} J_{ij} k_{ij} \right\},$$

where θ_i is a joint variable and J_{ij} is a constant 4×4 array whose exponent yields the homogeneous matrix representation of the unit screw scaled by parameter k_{ij}. If these $3(m+1)$ parameters were known then g_k would yield for every jointspace location, $q = (\theta_1, ..., \theta_n)^{\mathrm{T}} \in \mathcal{Q}$, the homogeneous representation of the tool frame origin in base frame coordinates.

Now denote by H_0 the homogeneous matrix representation of the screw relating the "left" hand camera frame to the base frame,

$$H_0 = \exp \left\{ \sum_{j=1}^{6} k_{0j} J_{0j} \right\},$$

where J_{0j} constitutes an arbitrary basis for the Lie Algebra corresponding to the group of rigid transformations and $\tilde{k}_0 \in \mathbb{R}^6$ parametrizes the relative camera frame transformation matrix accordingly. The camera transformation is now characterized by the parameters $k_0 = (k_{00}, k'_{00}, \tilde{k}_0) \in \mathbb{R}^8$ that appear in the the stereo projective transformation, $\mathbf{p}_{k_0} : \mathbb{R}^3 \to \mathbb{R}^4$, that for a given camera pair associates with each spatial point a pair of ("left" and "right" camera-) planar points. Specifically, let Π, π denote the projections from $\mathbb{R}^4$ that pick out, respectively, the first two, and the third coordinate, of a homogeneous representation of a point. The camera transformation may be written as

$$\mathbf{p}_{k_0}(w) = \begin{bmatrix} \Pi(w)/k'_{00}\pi(w) \\ \Pi(H_0 w)/k_{00}\pi(H_0 w) \end{bmatrix}.$$

This function admits a family of pseudo-inverses $\mathbf{p}^{\dagger}_{k_0} : \mathbb{R}^4 \to \mathbb{R}^3$, whose effect on the camera image plane, $\mathbf{p}_{k_0}(\mathbb{R}^3) \subset \mathbb{R}^4$, returns the original spatial point — that is $\mathbf{p}^{\dagger}_{k_0} \circ \mathbf{p}_{k_0}$ is the identity transformation of $\mathbb{R}^3$ — and whose effect off the camera image plane is to return the "closest" spatial point to that four-vector with respect to a suitable metric.

A Modified Procedure Hollerbach's proposed procedure tested in simulation of a planar arm, [2], calls for recording some number of joint-space/camera-image pairs, $\mathcal{D} = \{(q_l, c_l)\}_{l=1}^{n}$,

and then performing a Newton-like numerical descent algorithm on the cost function

$$\sum_{l=1}^{n} \|\mathbf{p}_{k_0}^{\dagger}(c_l) - g_k(q_l)\|^2.$$

When we attempted to implement this procedure for the three degree of freedom Bühgler arm, we found that the procedure was extremely sensitive numerically.

Instead, we have had great success with a variant on this idea that substitutes a cost function in the stereo camera image space,

$$\sum_{l=1}^{n} \|c_l - \mathbf{p}_{k_0} \circ g_k(q_l)\|^2,$$

for the previously defined workspace objective. We have been using this procedure on average several times a month (the experimental apparatus is frequently torn down and put back together again to incorporate new hardware, necessitating continual re-calibration) for the last six months with very good results. Starting from eyeball guesses of $k = (k_0, k_1, k_2, k_3, k_4)$, we have been able to achieve parameter estimates that give millimeter accuracy in workspace after two or three hours of gradient descent farmed out on a network of eight 1.5 Mflop microcomputers (Inmos T800 TRAMS). We have experienced similar reliable convergence properties with a variety of algorithms — standard gradient descent; Newton Raphson; Simplex descents — none of which seemed to avail (either singly or in more clever combination) using the original objective function.

References

[1] R. L. Andersson. *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control.* MIT, Cambridge,MA, 1988.

[2] D. J. Bennett, J. M. Hollerbach, and D. Geiger. Autonomous robot calibration for hand-eye coordination. In *International Symposium of Robotics Research*, 1989.

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

[4] M. Bühler , D. E. Koditschek, and P.J. Kindlmann. Planning and Control of Robotic Juggling Tasks. In H. Miura and S. Arimoto, editors, *Fifth International Symposium on Robotics Research*, pages 321–332. MIT Press, 1990.

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

[6] M. Bühler, D. E. Koditschek, and P. J. Kindlmann. Planning and control of a juggling robot. *Int. J. Rob. Research*, (submitted), 1991 .

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

[8] M. Bühler, N. Vlamis, C. J. Taylor, and A. Ganz. The cyclops vision system. In *Proc. North American Transputer Users Group Meeting*, Salt Lake City, UT, APR 1989.

[9] M. Bühler, L. Whitcomb, F. Levin, and D. E. Koditschek. A new distributed real-time controller for robotics applications. In *Proc. 34th IEEE Computer Society International Conference — COMPCON*, pages 63–68, San Francisco, CA, Feb 1989. IEEE Computer Society Press.

[10] D. E. Koditschek and M. Bühler. Analysis of a simplified hopping robot. *Int. J. Rob. Research*, 10(6), Dec 1991 .

[11] Daniel E. Koditschek. Natural motion for robot arms. In *IEEE Proceedings 23rd Conference on Decision and Control*, pages 733–735, Las Vegas, Dec 1984.

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

[13] A. A. Rizzi and D. E. Koditschek. Progress in spatial robot juggling. Technical Report 9113, Yale University, Center for Systems Science, New Haven, Connecticut, USA, October 1991.

[14] J. L. Synge and B. A. Griffith. *Principles of Mechanics.* McGraw Hill, London, 1959.

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

[16] L. L. Whitcomb and D. E. Koditschek. Robot control in a message passing environment. In *Proc. IEEE International Conference on Robotics and Automation*, pages 1198–1203, Cincinnati, OH, May 1990.

[17] Louis L. Whitcomb, Alfred Rizzi, and Daniel E. Koditschek. Comparative experiments with a new adaptive controller for robot arms. In *Proc. IEEE Int. Conf. Rob. and Aut.*, pages 2–7, Sacramento, CA, April 1991. IEEE Computer Society.