Real-Time Vision for Robot Swat-Juggling

Sanjiv Singh

OtfU-RI-TR-90-27

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

December 10,1990

Table of Contents

List of Figures

Abstract

This paper describes the implementation of a vision system developed to track moving objects in real-time- In the implementation described, a "puck" sliding on an inclined plane is tracked so that its position and velocity may be used to "swat-juggle" it to a constant height at a specified lateral position. Raw centroid calculations of the puck based on images from a CCD camera are filtered using an "augmented" linear observer and an interpolator to produce puck state estimates. Experimental data and results are presented and compared to the previous method of puck state sensing.

L Introduction

Computer vision is a natural means of measuring positions and velocity of moving objects as in the case of a planar juggler developed at Center for System Science, Yale University that "swat-juggles"[1] upto two pucks falling freely on an almost frictionless plane inclined into the earth's gravitational field.

To be able to "swat-juggle," the robot must have access to position of the pucks. It is usually not possible to measure velocity directly, and hence a variety of methods maybe used to estimate velocity from position measurements. This paper will describe the methodology used to obtain position and velocity estimates at a high rate using computer vision.

Other researchers have reported similar work- in two cases a ball was tracked in real-time using special purpose vision hardware [Andersson, Atkeson]. Atkeson's juggling robot used a pair of cameras positioned orthogonally, to measure the position of a ball by calculating centroids of the image of the ball in a scene- There are two major differences between his work and that described in this report. Atkeson assumed an orthographic (as opposed a perspective) projection for each camera. This required both cameras to be positioned exactly orthogonal to coordinate frame of the robot. Under this assumption, camera calibration is a relatively simple process, but suffers in accuracy and the inability to be able deal with arbitrarily positioned cameras. We have implemented a method that does not stipulate the position of the camera(s). Secondly, position and velocity estimates were obtained at frame rate (30Hz). More precisely, every time a frame of vision data was obtained, a linear least square fit was performed to obtain the position of the ball at impact. Such an approach is purely geometric- there is no consideration of dynamic considerations like gravity and friction. In contrast, the work described in this report is motivated by the need to accurately determine the position of the "puck" being tracked at a high rate (1 Khz). Such a high data rate is obtained by filtering position estimates obtained at 60 Hz using an augmented linear observer that explicitly encodes gravitational and frictional forces. An interpolator that also encodes puck dynamics, is then used to obtain position and velocity estimates between position measurements.

Andersson's ping-pong playing robot also tracked a moving ball in real-time. His approach is similar to ours in that dynamics were used to correct position measurements. Andersson found it necessary to account for gravity and air-drag. These were compensated for by performing a local quadratic fit to position data at every measurement. Velocity and acceleration estimated from this fit were then used to compute higher order correction terms using a dynamic model which in turn were used to "pre-correct" the sums for the quadratic fit of following position measurements.

---

[1] Swat-juggling refers to the action of "swatting" an object (usually a puck or a ball) to a specified height without grasping the object

Buehler demonstrated a variety of juggles on the Yale Juggler as examples of a representative class of tasks which involve repeated robot-environment interactions [Buehler89, Buehler90a, Buehler90c]. Position and velocity estimation was accomplished by the use of an oscillator inside each puck in conjunction with an inductive grid. A single measurement of a puck using this scheme was not very accurate (+/- lcm) but was significantly improved through a high sampling rate and a linear observer that filtered the measurements. A problem with this scheme is that it is difficult to scale to the case where more than two objects must be tracked. Ideally, we would like a sensing mode that doesn't limit the number of objects that can be tracked. Further, for a system that juggles in three space, it will not be possible to use an inductive grid, whereas measurements based on vision are not similarly limited.

We were motivated by the need for a more general and extensible means of measuring puck states. The system implemented can be extended to track multiple objects as well as objects moving in three space in a straightforward manner. This report describes the integration of a real-time vision system with the planar juggler such that complete state estimation of the pucks is done using off the shelf CCD cameras. Once a snap shot of the puck is obtained from the camera an observer that encodes a Newtonian model of puck dynamics is used to filter position data and to estimate velocities. An interpolator is used to estimate puck states at small time intervals to match the output rate of the method that uses the inductive grid. Our work has essentially followed the same methodology used by Buehler except that we have have replaced the inductive sensing scheme with passive vision.

In the following section, we first discuss the apparatus used. The camera calibration scheme used is briefly described in Section HI, and the computing architecture for the entire juggler is detailed in Section IV. Section V discusses the design of the tracking system and Section VI discusses experimental results.

DL The Planar Juggler

An existing apparatus was used in the experimentation [Buehler89]. The apparatus consists of one or two pucks sliding on an inclined plane that are batted successively by a bar covered with a billiard cushion rotating in the juggling plane.

There are three parts to this system:

1* Puck State Sensing: to dose a feedback loop at a high rate, it is necessary to be able to access positions and velocities of the pucks at a high rate (approximately IKhz). Previously, puck sensing was accomplished by placing an oscillator inside each puck and burying a grid inside the juggling plane, thus imitating a digitizing tablet. State estimation was accomplished by measuring grid voltages induced in the grid by the pucks and filtering the raw data using puck dynamics. A sensing module (processor and multiplexing hardware) was dedicated to this task.

A new sensing module has been designed that uses vision as the mode of measuring puck positions and velocities.

2. Tuggling Algorithm Computation: a separate module is designated to compute the reference trajectory (angle and angular velocity) of the juggling bar given the robot state (from a shaft encoder on the juggling arm motor) and puck states.

3. Motor Servo Control: this module is dedicated to commanding a high torque DC servo actuator at a rate of approximately 1.5 Khz using a PD algorithm. Since the control cycle is faster than the sensing cycle, sometimes, new sensed data is not available. In this case, it suffices to use data that is late by one sensing cycle since the control loop is closed at a very short time interval.

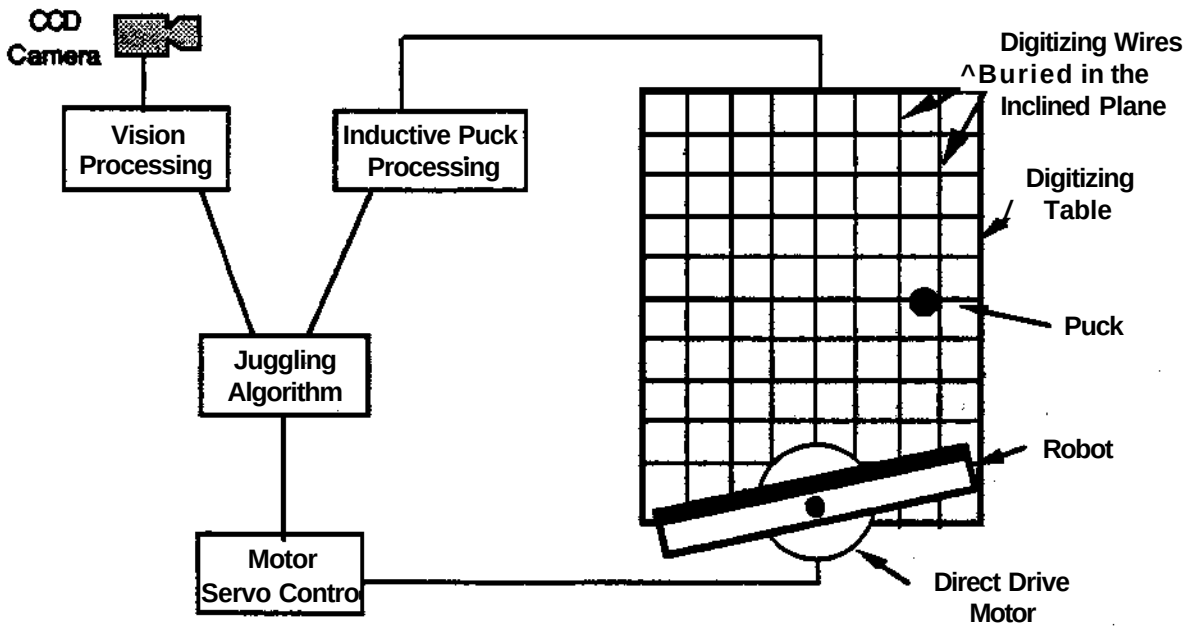


Fig. 1: Configuration of the Planar Juggler

Fig. 1 shows the configuration of the juggler with the added vision system. Both puck state sensing modalities were retained so as to compare data from the grid sensor and the vision sensor, though juggling was accomplished only through the use of vision data-

III Camera Calibration

Figure 2 shows the physical setup of a CCD camera mounted on the ceiling looking at the juggler. Note, that the camera has been rotated about the optical axis by 90

degrees so as to provide greater sensing resolution in the vertical direction (the camera has more columns than rows).

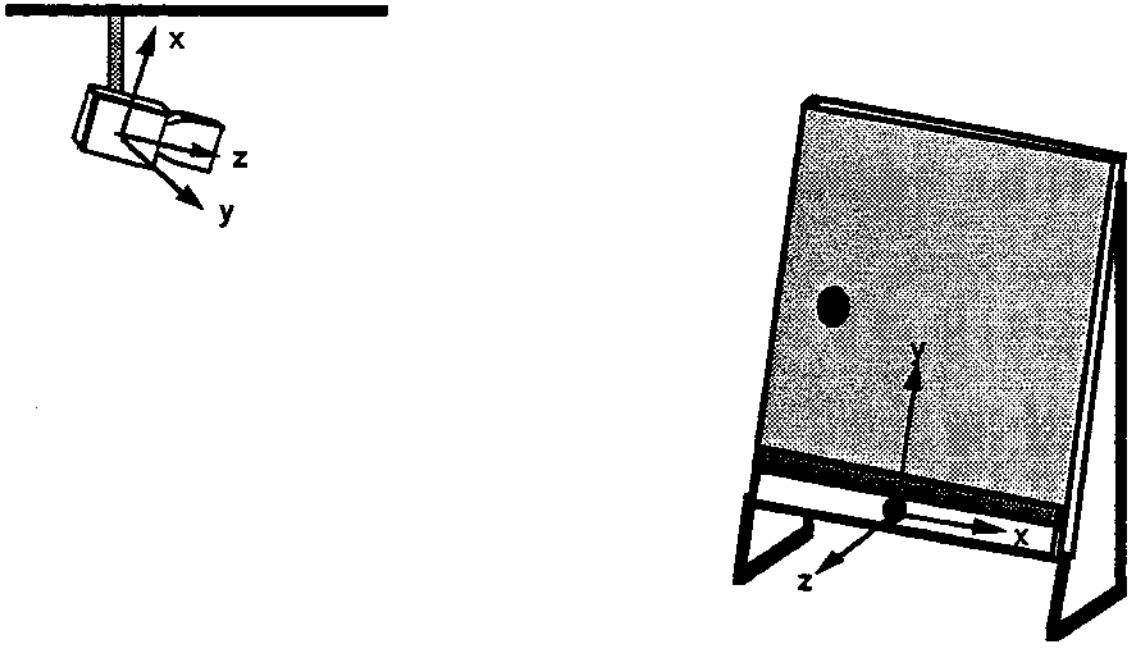


Fig. 2: Camera setup for position sensing

To be able to sense puck position, it is necessary to go through a calibration phase which relates pixel locations in the image to physical locations on the juggling plane. The calibration scheme [Tsai], computes the following parameters:

/: focal length of the camera
s: a fudge factor to compensate for disparate digitization timings between the camera and the frame grabber.
kl%2: lens distortion parameters
R: A 3X3 rotation matrix that describes the transformation from the camera frame to the world frame
T: A 1X3 translation matrix that denotes the translation vector between the above two frames

It is necessaiy to provide input to this algorithm in the form of a set of training points for which both spatial coordinates in world frame and image coordinates in the image space are known. For objects that lie in a plane, it is only necessary to provide teaming points that all lie in a plane. However, for the general case in which objects that may lie in three space, it is necessary to provide training points that lie in several planes. This is accomplished by using an image of a calibration grid located on the juggler as shown in figure 3(a). Figure 3(b) shows an actual image of the calibration grid obtained from the CCD camera.

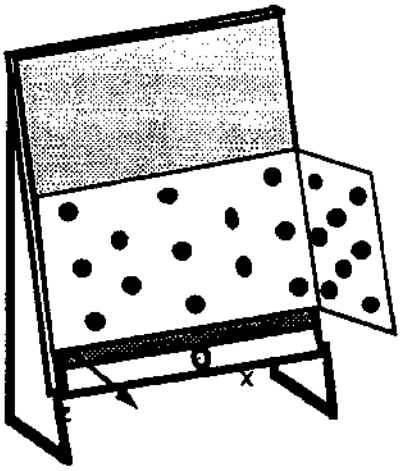


Fig. 3(a): The CaHbration Grid Located on the Inclined Plane.

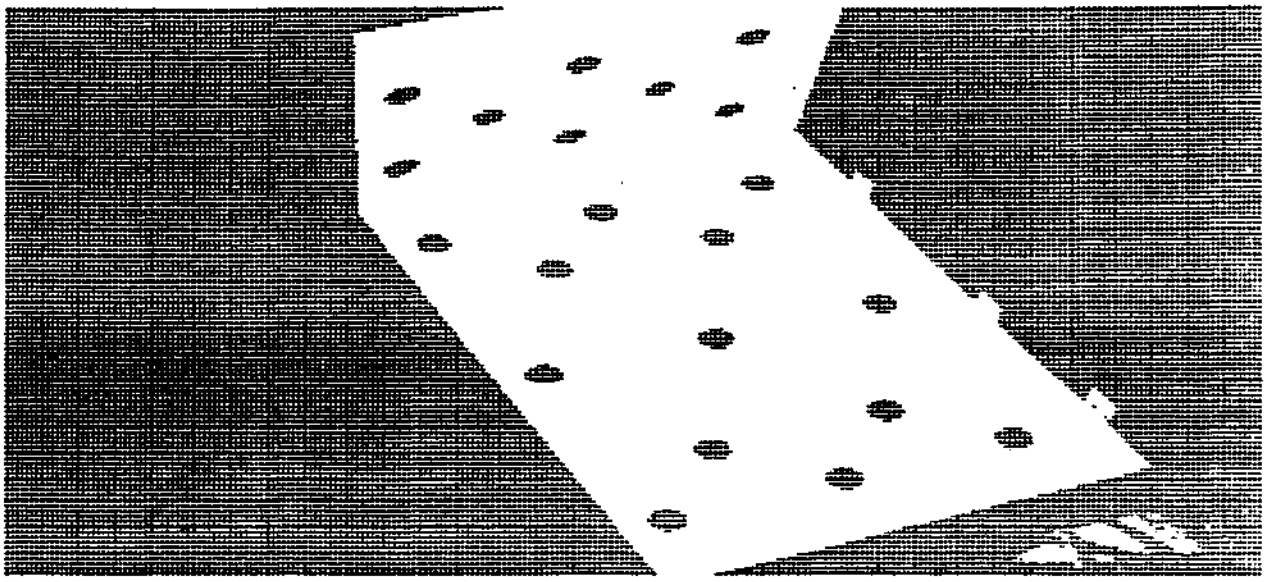


Fig.3(b): An Image of the Calibration Grid

The location of the centroid of each of the circles on the grid is known a priori through careful measurement, while the corresponding centroids of the circles in image space must be computed. At present, the correspondence problem of deciding which circle in three space corresponds to which circle in the image space is solved by hand. The result of this process is a set of coordinates for every circle on the calibration grid:

row, col: the centroid in image space
x,y,z : the centroid in world space

The calibration scheme uses these training points as input to a nonlinear minimization scheme to estimate the parameters *(ffjcljd*, JR,T). Once these are known, given an (i,j) pair in image space, the corresponding point in world space can be solved by Eqns. 8(a) and 8(b) in Tsai's report. Essentially, two relationships- f(x,z), f(y,z) are obtained.

If the object(s) being tracked move in a plane then the z parameter is known and it suffices to simultaneously solve for x and y in the two equations above. However, if the object moves in 3-space, two cameras are needed. For each camera there are two such ftinctions, giving rise to an over constrained set of equations: fl(x,z), fl(y,z), f2(x,z), f2(y,z). These four equations can be solved using a variety of minimization methods.

In our case the pucks move in a plane so a single camera suffices.

IV. System Architecture

The Cyclops vision system [Cyclops] was used to track the pucks. The Cyclops system consists of three components:

- **digitizer: the digitizer digitizes the RS-170 signal from the CCD camera and outputs the image on a video bus. The RS-170 signal is interlaced, so each of the half-frames is broadcast on the video bus, alternately.**

- **memory modules: An arbitrary number of memory modules can listen to the video bus. These are configured to listen load one of the two half frames broadcast by the digitizer.**

- **frame-processors: Each memory module has a frame-processor attached to it that operates on the data loaded into the memory module. Each frame processor is able to communicate with other processors in the system via messages.**

The configuration is shown in figure 4.

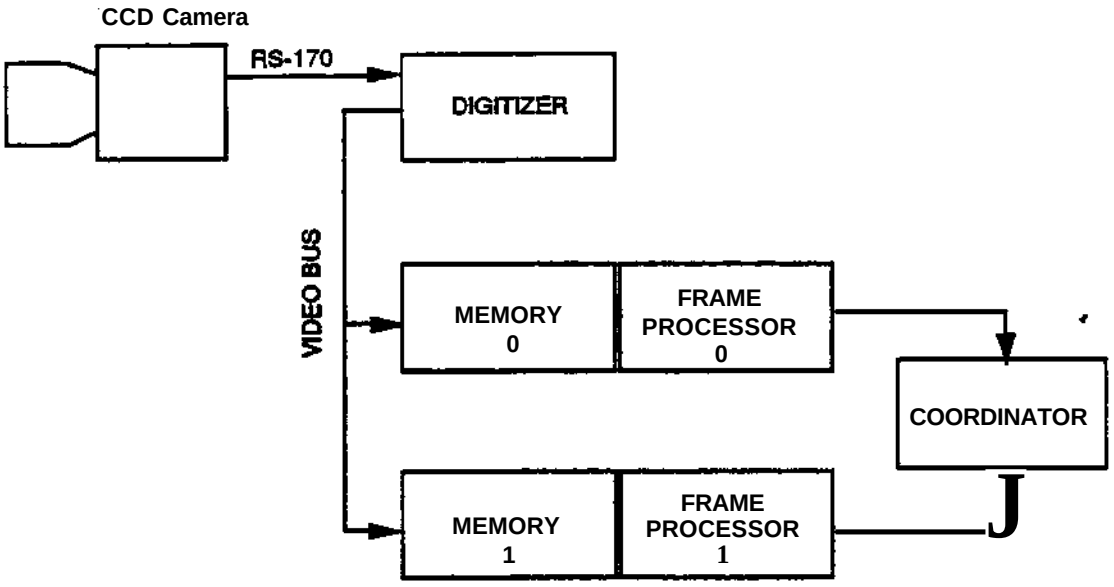


Fig. 4: The Cydops Vision System

In this case, each frame-processor is dedicated to a window on each half frame for the sake of efficiency. This window is moved as the object being tracked moves. We used a window of 30X60 pixels and were able to do all the processing in the window within 7ms. The window size was large enough that once tracking was initiated, the puck never escaped the window under normal experimentation.

Within this window, a binary thresholding operation is done and the centroid (first order moment) of the bright pixels is found. Since the puck is a bright object against a dark background, the operations of thresholding and finding centroids are straight forward- all pixels within the window that have pixel values above the threshold are averaged in their *x* and y coordinates to obtain the centroid. Each frame-processor then converts the centroid information to world coordinates using the calibration equations and these are sent to a "Coordinator" module.

Since each frame-processor alternately, gets a frame at 30Hz, it is possible to combine the data from both processors to obtain state estimates at 60 Hz. The Coordinator serves exactly this function. Its duties are two fold:

- **Updating the window: As each frame-processor sends centroid information to the Coordinator, the new position of the puck in the image is sent to the other frame-processor. This allows the other frame processor to use an updated estimate of the window in which to find the puck in the next sensing cycle.**

- **Filtering raw position data: Data from vision system is noisy on a per sample basis but it can be filtered using a linear observer that uses puck dynamics to smooth raw position data obtained from the vision system. Figure 5 shows (x) raw position data obtained from the vision sensor. The**

observer is also used to estimate puck velocities. The filtering process is discussed in detail below.

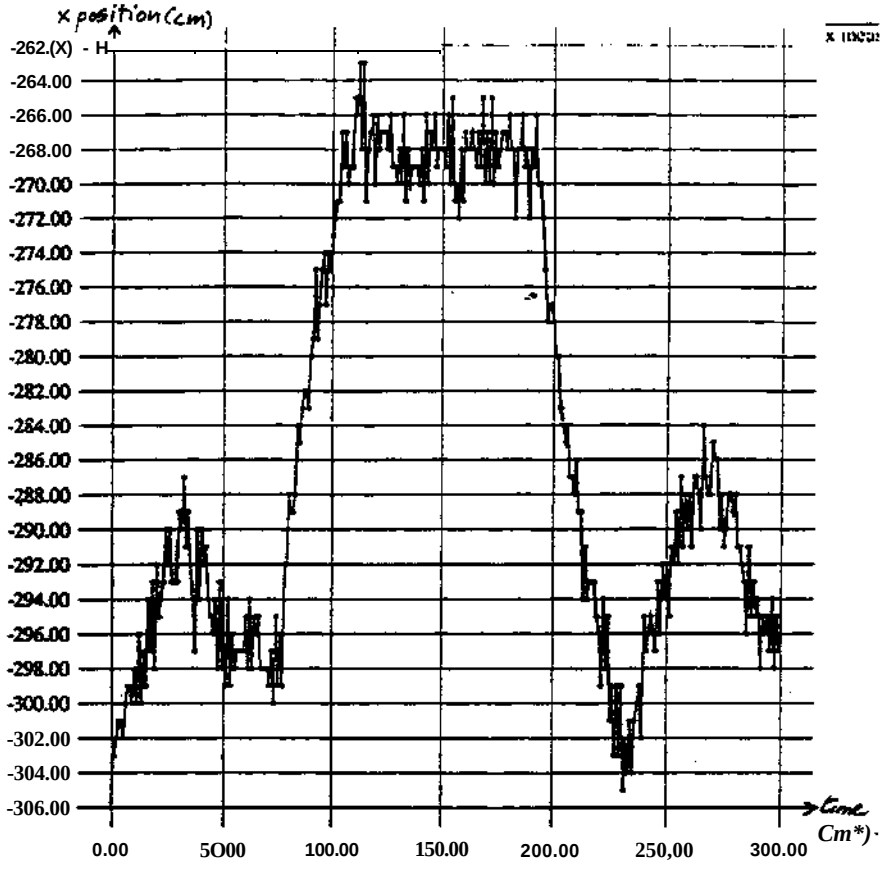


Fig. 5:Raw (x) Position Data from the Vision Sensor

The previous incarnation of the juggler used 3 T-800 transputers to accomplish juggling. One transputer was dedicated to each of juggling algorithm computation, puck sensing and motor servo control. Additionally a T-400 transputer was used as a host interface to a PC AT, primarily for the purposes of compilation and data logging. The current work has introduced 4 additional processors. 2 T-800 transputers are used as frame-processors for low level image calculations. Their output is sent to another T-800 transputer that runs the "coordinator" tasks. A fourth transputer (BOO-7) is used as a graphics processor to display the located centroid of the puck in the image. The whole system is configured as in Fig 6.

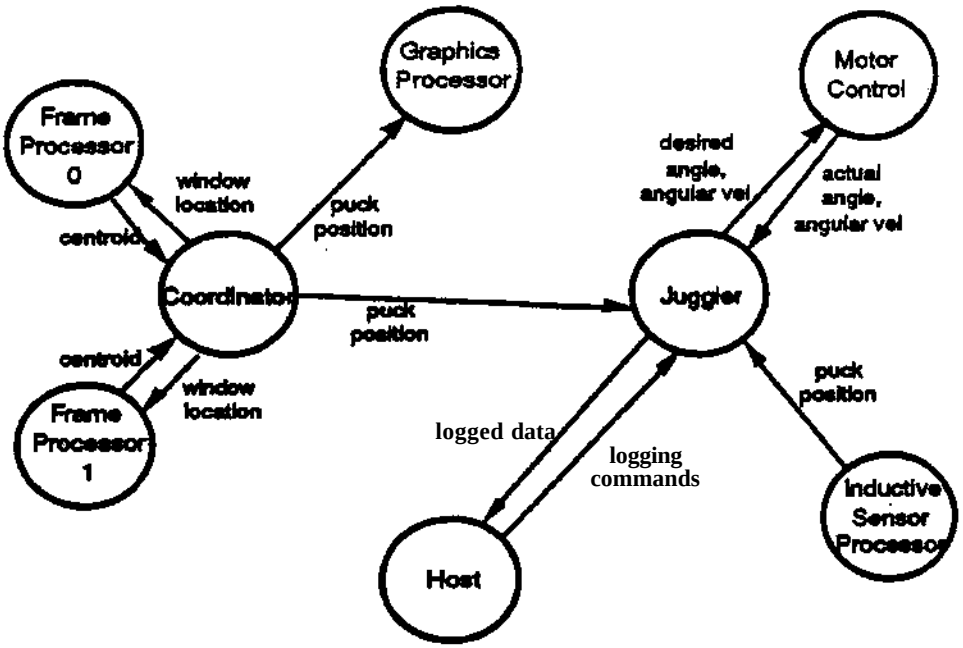


Fig. 6: TheMIMD Architecture for the Juggler

V. Tracking the Puck

A *Puck Dynamics*

A **complete model of puck dynamics incorporates gravitational forces and friction forces experienced by die puck. Friction forces can be characterized by two different kinds of effects- dry friction and viscous friction. The former is the force experienced during constant sliding, whereas viscous friction is the force required to move a stopped object. Since the puck is in motion constantly, it was sufficient to only use estimates of dry friction. Hence a simplified model of puck motion can be described by:**

$$\ddot{y} = -g - \dot{y} f_y \quad (1)$$

$$\ddot{x} = -\dot{x} f_x \quad (2)$$

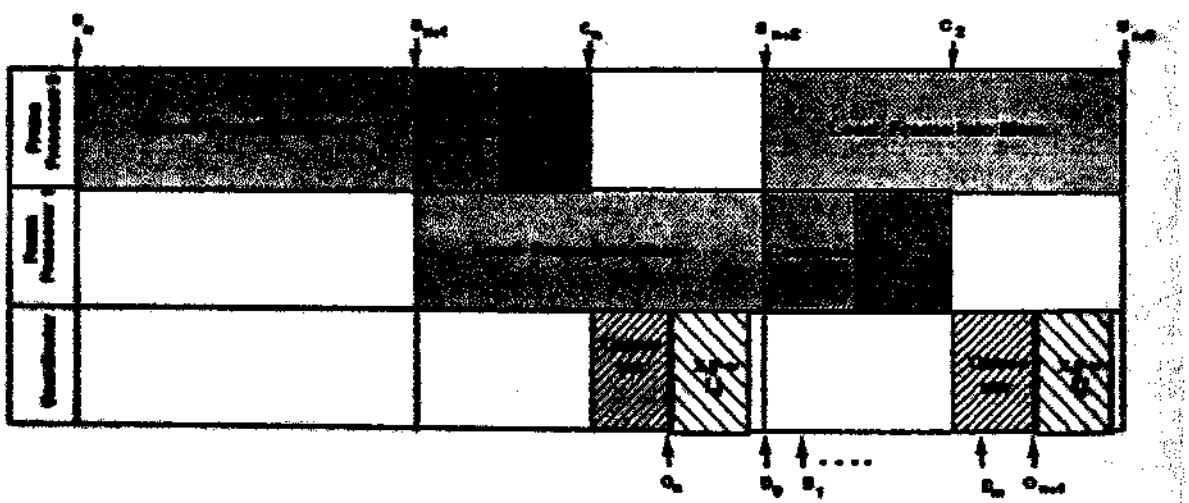
Here x, y **denote the puck position,** g **is the acceleration due to gravity,** f_x **and** f_y **are the dry friction terms in the** x **and y directions, respectively. Dry friction has been experimentally determined to be 0.16 [Buehler90b]. However, this estimate was good for high velocities, which in our task, are in the y direction Velocities in the** x **direction the juggling task described are almost negligible, and hence it we have found that is possible to ignore the friction force in the** x **direction.**

Puck dynamics are much harder to encode at the impact. A restitution model is used in the control of the juggling arm [Buehler90b]. For the purposes of tracking the puck, we have used an ad hoc scheme that resets some of the observer states when impact is detected.

B. *System Timing*

Figure 7 shows the time separation of events. The CCD camera shutter opens at ^ S2,.., Sn (two shutter events are separated by l/60th sec). At time S_{n+}i, the first half-frame is completely loaded into frame-processor 0. Immediately, frame-processor 1 starts loading the next half frame of the RS-170 video signal. At time Cn, frame-processor completes the centroki computation and conversion of the centroid to from *ij to* x,y coordinates. This information is sent to the Coordinator which uses an observer to filter the position data. The centroid data is converted back into Image coordinates and sent to frame-processor 1 such that as soon as the the second half feune has been loaded, the new position of the puck can be used in the new centxoid operation*

Notice that if the observer is written in standard form (new estimates of the state at time n are based on state estimates from time *n-l* and measurements from time step »), liim the output of the observer at time O_n is an estimate of the states at time S* Sinct w© CM only reliably complete the state estimation by time Sn+2, we would like the estimation scheme to output state estimates for the puck at time S_{n+}2. Further, the next measurement (shutter exposure) happens at Sa+i- We solve this problem in two steps. H » i we write the observer in such a way that its output Sn+2 is aa estimate of the states at S^+i- Next we use a staple predictor to carry the motion of the puck forward in time by the time $(S_{n+1} - S_n)$.



fljf 7*: System Tfanfaog

There is one further issue. We would Mice the observer to output at high rate (I Khz), but the output of the observer is only once every l/60th of a sec. To achitvt, this high rate between observer outputs, we use a simple interpolator that carrkt the equations of motion in time forward by a delta (= 0.001s) at each output *fa*, D*. (At tune D_{3J} the ftest full state estimate (i and y positions and velocities! is available *J* To actually achieve this, the processor that the coordinator runs on,

must multitask between running the observer and running the interpolator at fixed time intervals. The interpolator thus is encoded as an interrupt task, that is, it is executed with a high priority at every lms.

The **data flow in the entire** process is shown in figure 8.

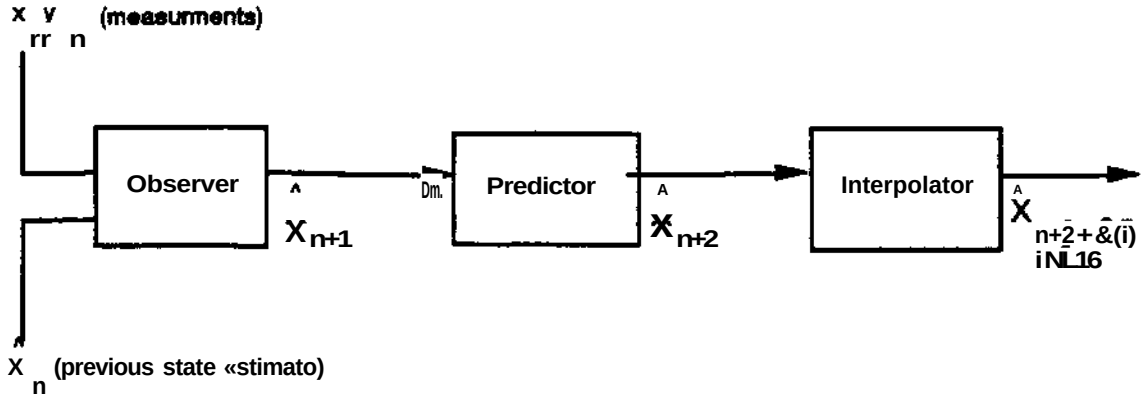


Fig. 8: Data Flow

Note that the interpolator produces an output every ms and every 16 msa **new measurement is obtained and the interpolator is re-initialized.**

C. Observer design

Writing the dynamics of the puck in state space form gives:

$$\dot{X} = Aoc + B.u \quad (3)$$

$$Y = Cx \quad (4)$$

or more explicitly:

$$\begin{bmatrix} \dot{x}1 \\ x2 \\ x3 \\ Lx4J \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -f_x & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & O & -fyj \end{bmatrix} \begin{bmatrix} x1' \\ x2 \\ x3 \\ Lx4J \end{bmatrix} + \begin{bmatrix} 0" \\ 0 \\ 0 \\ L-IJ \end{bmatrix} \quad (5)$$

$$Y = [1\ 0\ 0\ 0] \begin{bmatrix} x1 \\ x2 \\ x3 \\ x4J \end{bmatrix} \quad (6)$$

where *xl, xl* are the position and velocity states in the *x* direction and *x3, x4* are the position and velocity states in the y direction.

Given a 60 Hz sampling rate, the above can be written as an equivalent difference equation where the A and B matrices have been appropriately transformed to **<I> and r.**

$$x(n+1) = <\&.\ x(n) + T.\ u(n) \tag{7}$$

For values of fx = 0.0, fy = 0.16, this is given by Eqn 8.

$$\begin{bmatrix} xl(n_{+}l)l \\ x2(n+l) \\ x3(n+l) \\ Lx4(n_{+}l)J \end{bmatrix} = \begin{bmatrix} rj & 00162 & 0 & 0\text{-}i \\ o & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.0161 \\ L_o & o & l & a9974J \end{bmatrix} \begin{bmatrix} |"xl(n) \\ *2(n) \\ x3(n) \\ [x4(n) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -0.001 \\ -0.016. \end{bmatrix} \cdot g \tag{8}$$

We write the observer in a form such that the output of the observer is a prediction of the states at time step n+1 given measurements at time step n:

$$\hat{x}Xn+1) = <\&\hat{x}(n) + r.u(n) + L(y(n) - C\hat{x}(n)) \tag{9}$$

$\hat{x}$ denotes an estimate of the state vector while y is denotes the vector of actual measurements. L is the observer gain matrix and is computed by pole placement (poles of the observer are at 0.95, 0.955 for the *x* system and 0.4, 0.399 for the y system). The resulting L matrix is:

$$L = \begin{bmatrix} 0.095 & 0.000 \\ 0.135 & 0.000 \\ 0.000 & 1.198 \\ L0.000 & 21.473J \end{bmatrix} \tag{10}$$

The observer gain can be thought of as weights on the error between measured states and estimated states. Essentially pole placement is a way of encoding how much the measurements can be trusted in comparison to the dynamic model. In the former case, a very large weight is placed on the error between measured and estimated states ("fast" poles- dose to, but less than unity). Alternatively, if there are large errors in measurement, "slower" poles (closer to, but greater than zero) are used to weigh the errors less and rely more on past data. The price paid for having fast response, is sensitivity to noise. In our application, we have ignored any dynamics in the *x* direction and thus want to pay relatively more attention to the measurements that the perfect integrator model. Additionally the velocities in the x direction are small and measurement errors are relatively minor. Hence the observer poles for the *x* system are quite fast. Alternatively, the y system has significant dynamics effects which are well modeled along with high velocities. Correspondingly, the observer poles for the y system are slower.

D. The Predictor

As mentioned above, the task of the predictor is to compensate for the latency between title observer output and the actual puck states. This effect is achieved

simply by integrating forward in time using Eqn. 8 after every estimate of the observer. Thus, at O_n we have an estimate of puck states at S_n+2-

E. *The Interpolator*

Since we would like to get state estimates at a high rate, we can repeatedly use the same method that the predictor uses, only this time integrating over a much smaller time interval. Eqn. 11 gives the system dynamics as a difference equation discretized at a time interval of 0.001 s.

$$\begin{bmatrix} x1(n+1) \\ x2(n+l) \\ x3(n+l) \\ x4(n+l) \end{bmatrix} = \begin{bmatrix} 1 & 0.0010 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.0009 \\ 0 & 0 & 1 & 0.9998 \end{bmatrix} \cdot \begin{bmatrix} xl(n) \\ x2(n) \\ x3(n) \\ x4(n) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -0.000000499 \\ -0.000999920 \end{bmatrix} \cdot g \qquad (11)$$

This allows for successive state estimates at Di (i = 0.. 16)

F. Dealing with Impacts

Since the observer designed assumes a linear system, the impact, which is highly non-linear, is treated as a disturbance. Thus, left to itself, the observer produces erroneous output for a short period after the impact, until the error between its estimates and measured positions becomes large enough to significantly effect the state estimates. This effect can be seen in figure 9 in which the output of the y position as estimated by the vision system (after being filtered by the linear observer) is compared with the output of the same state by the grid sensor. In this example, it has taken 3 sensing cycles in which the direction of the puck is reversed, before the velocity estimates go from negative to positive. On its own, the linear observer, treats the switch in the direction of the puck as a disturbance until repeated measurements cumulatively make the y velocity positive.

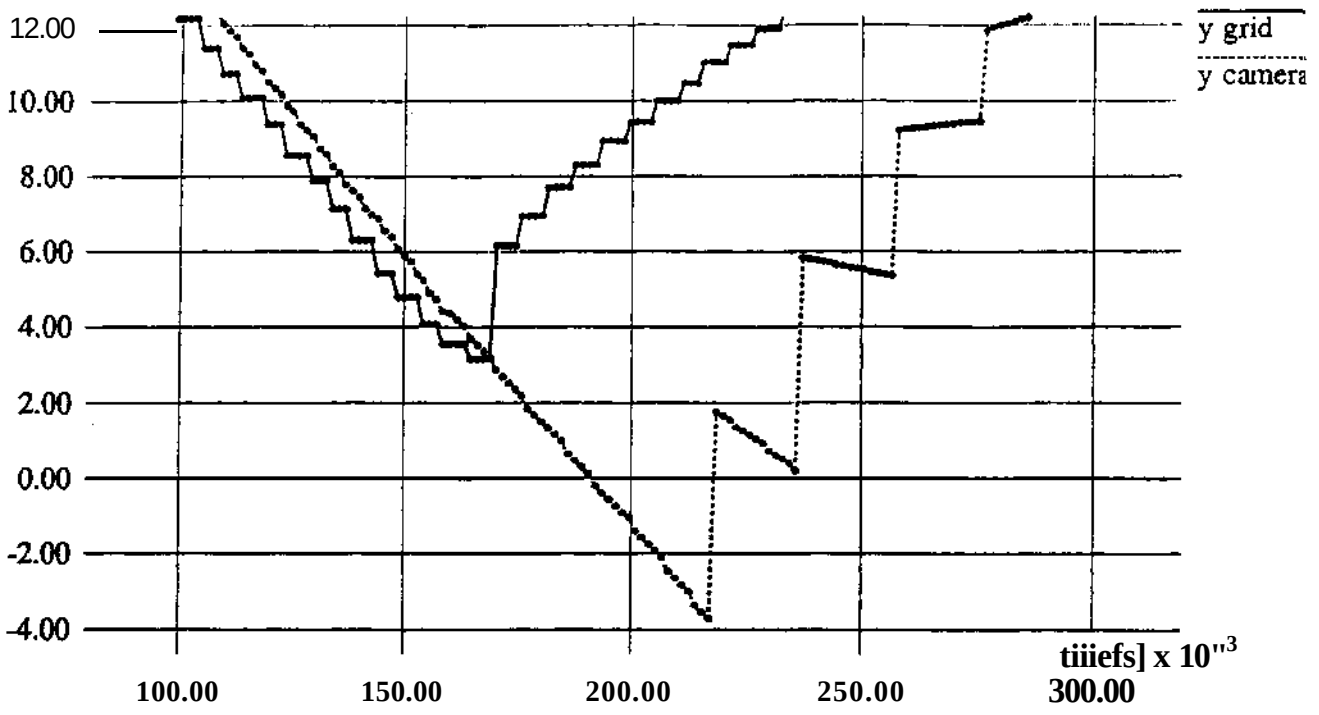


Fig. 9: Output (y position) of tlte linear Observer

Figure 9 shows the output of the y position by a linear observer. The solid line denotes the output of the same state by the grid sensor. It turns out that for juggling it is most important to have good state information just before the impact whereas the error due to the linear observer does not start showing until just after the impact. Still, we would like to augment the linear observer with a method that would improve the state estimates just after the impact. A simple heuristic suffices:

if

$$(y<e) \text{ and } (\dot{y}<0)$$

then

$$\dot{y} = -\dot{y}$$

where £ is a small distance above the juggling bar at zero angle.

Figure 10 shows the improvement in the observed states right after the impact

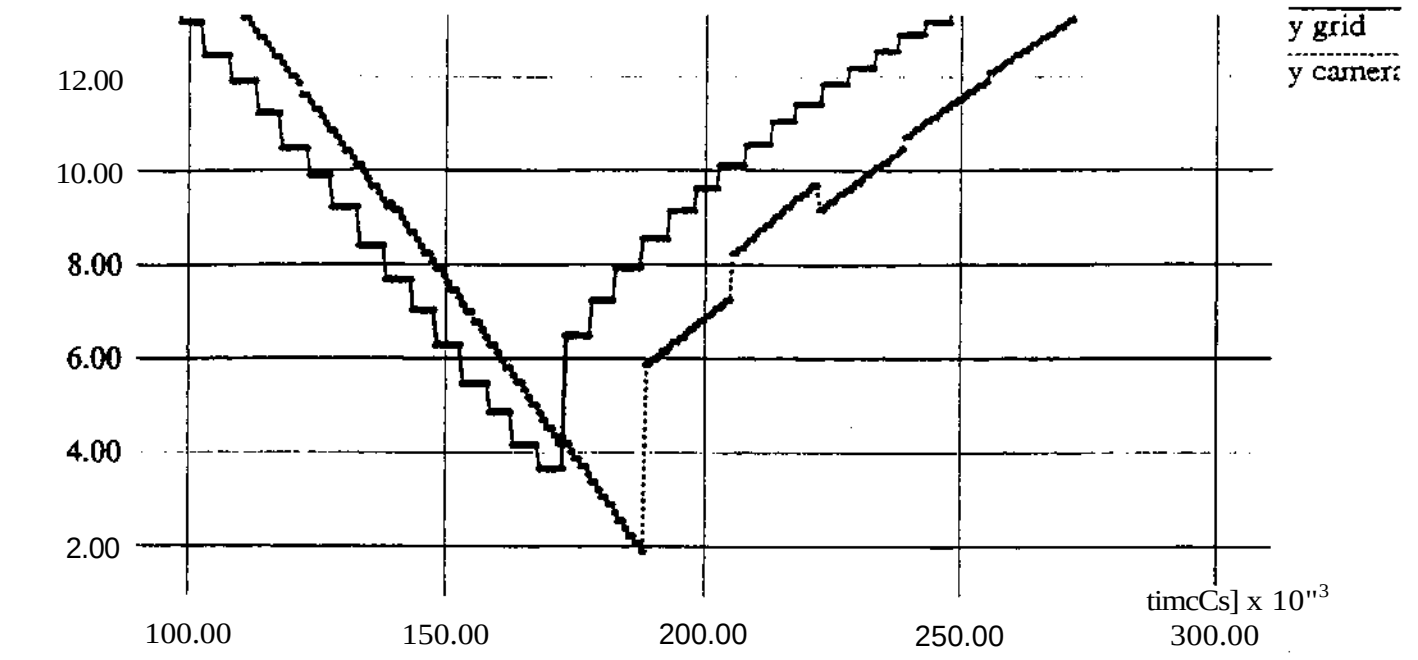


Fig. 10: Output of Linear Observer (7 position) with added heuristic

VL Results

Even though there is a large disparity in sampling rates and accuracy between the grid sensor and the vision sensor, it is possible to qualitatively match the performance of the grid sensor with the vision sensor. Since it is difficult to compare each sensing mode to an absolute reference frame- i.e to determine the errors of each sensing mode in an absolute sense, the discussion of results is restricted to a comparison between sensing modes. In all the experiments discussed, the vision sensor was used to obtain position as well as juggle. The grid sensor was run concurrently to collect data for purposes of comparison.

The results shown in Fig. 10 are typical. The best results (as evidenced by the smallest difference between the two sensing modes) are obtained exactly when state estimates are the most important- just before impact. In this case, the steady state difference is around 1 inch (2.5 cm). After impact, it takes a while for the observer attached to the vision sensor to produce estimates that match the grid sensor. This is chiefly due to the large difference in the sampling rates (60 Hz for the vision sensor vs. 1000 Hz for the grid sensor). Hence, right after impact, the difference in the two sensing modes can be as much as 6 inches (15cm). This difference is made up quickly and the typical difference as the puck moves up towards the peak is around 2 inches (4.5 cm).

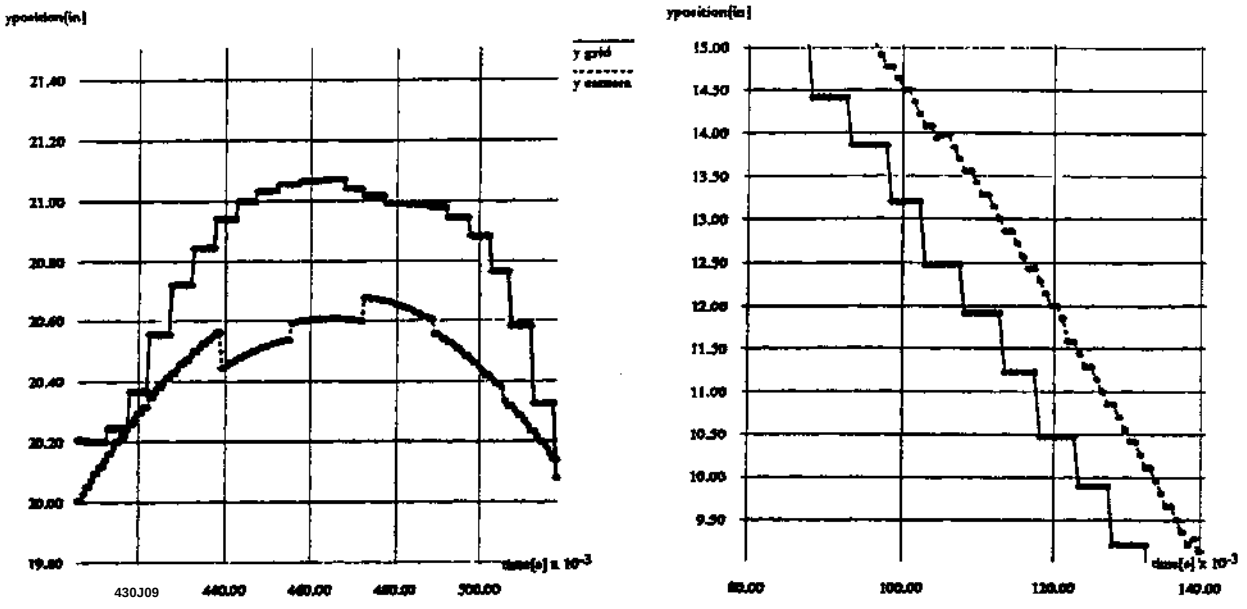


Fig. 11: Comparison of Position Data at Different Parts of the Trajectory (a) at the peak (b) during downward flight

In some cases, as at the peaks of y motion, the vision sensor provides a qualitatively better state estimate. By this, we mean that the the vision sensor shows a more "natural" peak than does the grid sensor (Fig. ll(a)). Notice that the vision sensor produces discontinuities every so often. These correspond to outputs of the observer/predictor when a measurement from vision data is obtained. The following points until the next discontinuity come from interpolation where the puck dynamics are used instead of measurements. Use of the grid sensor does not involve any interpolation, rather its output depends on running a high bandwith of measurements through the observer. Fig. ll(b) compares data from the two sensors when the puck is falling, when it is most important for the sake of juggling to have good data. Figure 12 compares the y velocity estimates by the two sensing modes.

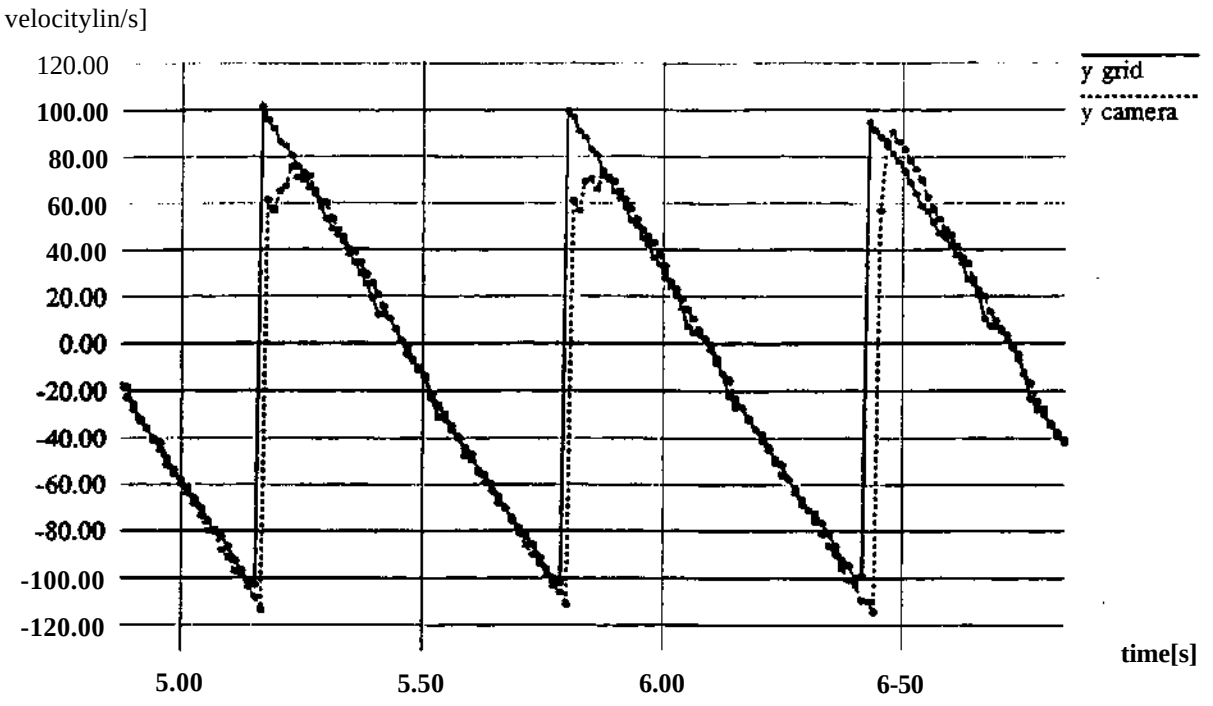
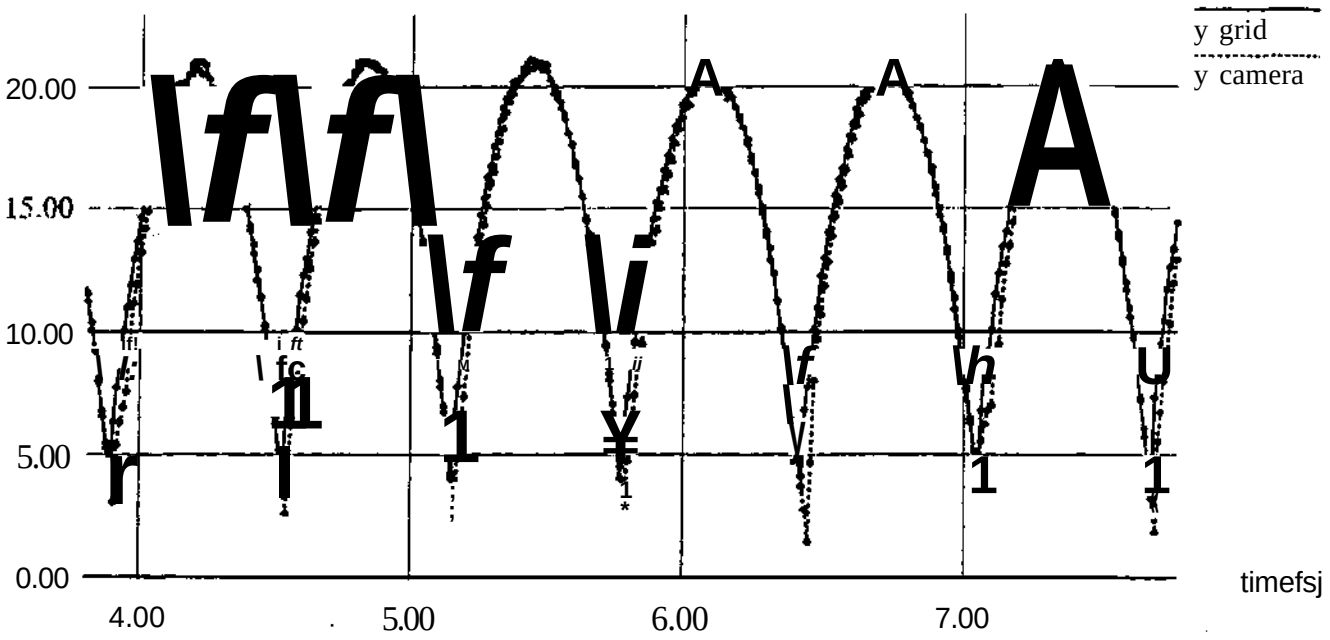


Fig. 12: (y) Velocity Estimates

It can be seen that right after impact, there is a large difference in the velocity estimates by the two sensing modes. This is once a again due to the difference in the sampling rate- the grid sensor detects the impact much sooner than the vision sensor.

Finally, figure 13 compares the y position estimates for repeated juggles by the two sensing modes.



Hg. 13: Comparison of Steady State Performance (y position) between Vision and Inductive Sensing

Finally, the proof of the pudding is in how well the robot is able to juggle under each sensing mode. Our experiments with both sensing modes have shown that the robot can juggle ad infinitum (our longest experiment with the vision sensor lasted 5 minutes without any degradation of performance)*

VIL Conclusions

A sensing scheme using machine vision has been demonstrated that tracks a moving object in real time. This scheme was tailored to produce full state estimates (positions and velocities) of a puck falling on an inclined plane such that it was possible to replicate previously demonstrated swat-juggling. The advantage of this scheme is that it is not as contrived as the previous scheme in which it was necessary to have active electronic circuits inside each of the pucks. Most importantly, this scheme scales nicely for tracking multiple objects- it is only necessary to replicate part of the hardware- each additional module is dedicated to tracking a single object. Hie scheme described can also be extended in a straight forward manner to tracking objects moving in 3-space.

Acknowledgements

Acknowledgement is due to Martin Buehler for his pioneering work in robot juggling. A big thank you to Al Rizzi and Louis Whitcomb for their untiring support with debugging the system. The author would like to acknowledge Dan Koditschek for the opportunity to do this work and the stimulating discussions that put the work in perspective.

References

[Andersson] **R. Andersson, *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control,* MTT Press,1988.**

[Atkeson] **E. Aboaf, S. M. Drucker, C. G. Atkeson, *Task Level Robot Learning: Juggling a Tennis ball More Accurately,* In *Proc. IEEE International Conference on Robotics and Automation,* Cincinnati, Ohio, May 1989.**

[Buehler89] **M Buehler, D. E Koditschek, P. J. Kindlmann, *A Simple Juggling Robot: Theory and Experimentation,* In *Proc. First International Symposium on Experimental Robotics,* Montreal, Canada, June , 1989.**

[Buehler90a] **M. Buehler, D. E. Koditschek, *From Stable to Chaotic Juggling: Theory, Simulation and Experiments,* In *Proc. IEEE International Conference on Robotics and Automation,* Cincinnati, Ohio, May 1989.**

[Buehler90b] **M. Buehler, *Robotic Tasks with Intermittent Dynamics,* , Phd Thesis, Yale University, 1990. University Microfilm, Public #9035329, Ann Arbor, ML**

[Buehler90c] **M. Buehler, D. E. Koditschek, PJ. Kindlmann, A Family of Robot Control Strategies for Intermittent Dynamical Environments, IEEE Control Systems magazine 10(2): 16-22, Feb 1990**

[Cyclops] **M Buehler, N. Vlamis, C. J. Taylor, A. Ganz, *The Cyclops Vision System, In Proc. North American Transputer Users Group Meeting,* Salt Lake City, UT, Apr 1989.**

[Tsai] **R. Tsai, *A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off the Shelf TV Cameras and Lenses,* IBM Research Report, RC 11413, 1985**