

# SENSING AND CONTROL FOR AUTONOMOUS GRASPING IN DYNAMIC ENVIRONMENTS.

M. Zhang, J. Damianakis and M. Buehler

Centre for Intelligent Machines  
McGill University, Department of Mechanical Engineering  
Montréal, QC, CANADA H3A 2A7

## Abstract

Autonomous grasping in dynamic environments where the object, the robot or both are moving is an increasingly common task in manufacturing. Since the environment is partially or completely known, this information can be exploited during sensing. We report on work in progress towards the development of a proximity sensing network, located in a robot's multi-fingered gripper. This network will form an integral part of a multistage sensing system with eye-in-hand vision and tactile sensor pads. Sensing information is passed on to an online trajectory planner – a geometric controller – which evaluates a nonlinear memoryless function to map the current object position and velocity into a desired robot pose. If the robot tracks these set-points, it is guaranteed to match the object's velocity and acceleration on a specified grasp surface. A planar simulation demonstrates that this paradigm performs favorably when compared with the traditional planning approach. Since it does not depend on future state predictions, no object model is required. Without trajectory prediction, the computational effort is drastically reduced, allowing for higher controller speed and tracking feedback gains. At the same time this approach provides a framework for general sensor based control of robotic tasks.

**Keywords:** Dynamic Grasping, Sensor-Based Control, Geometric Control, Proximity Sensing Network, Robotics.

## 1 Introduction

Robotics is advancing rapidly into applications which require increasing dexterity and dynamic response. This is heightening the demand for tightly integrated sensing and control, “sensor-based control,” to support autonomous or supervisory operation. Full master-slave operation is often too slow, inaccurate or impossible due to time-delays between master and slave. In manufacturing setups, much or all of the environment is known, and in this scenario, the critical pre-contact proximity information can be acquired by a number of small Infra-Red (IR) sensor pairs, embedded in the robot fingers. Such a sensor is inexpensive, robust and, when integrated with machine vision and tactile sensing, provides a suitable solution particularly for dynamic grasping tasks, where a smooth robot-object contact is required.

We define dynamic grasping as the approaching of a moving object and final velocity and acceleration matching at a desired grasp location. It is a fundamental task that precedes many manipulation operations and is by itself the main constituent of a large class of robotic tasks. In manufacturing, robots pick up objects from conveyor belts and J-hooks, and will in the near future have to grasp objects with a priori unknown trajectories handed to them from humans or other robots. In space, this problem occurs when capturing satellites or when containing and docking space assemblies. In other tasks, for example robotic live line maintenance of live power lines, both the robot on its boom as well as the power lines are moving.

The use of electro-optical proximity sensing has a long history, beginning with the work of Ernst at MIT [9] who used them as inputs to his event based programming language. Johnston at JPL [13] carried out a mapping experiment where the proximity sensor was used to follow the profile of a surface and to guide an end-effector into position for grasping. Bejczy [4] incorporated the IR sensors with a vision system to provide acoustic feedback in telerobotic operations. In [20] IR sensors aid telerobotic grasping when the manipulator is within the region of the object. Balek and Kelley used gripper mounted proximity sensors for robot control [3] via a hierarchical control scheme. In [17], Marszalec describes optical *fibre* based proximity sensor characteristics and their incorporation on a robot gripper. A fibre optic proximity sensor that measures object-sensor distance via a simple inverse square relationship of the magnitude of the received signal is discussed by Li [14]. Masuda [18] uses the phase shift of the received signal to measure distance, angle or orientation depending on the mode of operation. The sensor consists of six LED's in a cross shaped pattern with the photo transistor in the center. Goldenberg et al. [19] show how this sensor's design parameters affect its performance and propose an optimal sensor design.

One of the first accounts of dynamic grasping in 2D was given by Andersson [2] where rolling ping pong balls on a slanted surface were caught. Lin, Zeman and Patel [15] proposed a heuristic coarse/fine tuning procedure to match the object’s trajectory at contact. Houshangi [11] addressed explicitly the time delay introduced by image processing in visual feedback controlled robots by predicting the object trajectory via an auto-regressive discrete-time model. Hove and Slotine [12] implemented spatial robot catching based on a trajectory matching algorithm that combines an observer with a varying-strength filter, an error estimator, and an initial motion algorithm. Allen et al. [1] grasp a moving train with a PUMA robot under vision guidance based on straight line planning in cartesian space between the robot’s end effector and the target. Gosselin’s work on global inverse robot kinematics [10] includes a path planning scheme to catch moving objects: The last object state measurements are employed to predict its path as a parabola, assuming a free falling object. The most recent version of the classical sense-plan-replan paradigm is found in [8] where an object moving in the plane is caught based on a predicted path assuming constant acceleration. The problem of blending trajectory segments in real time subject to sensory inputs and tracking constraints is solved by Lloyd and Hayward [16].

The paper consists of two parts. In Section 2 we describe an amplitude based electro-optical proximity sensor suitable for known environments and which meets our particular specifications. Section 3 summarizes a sensor based approach for dynamic grasping based on geometric controllers.

## 2 Towards Robust multistage sensing

### 2.1 Requirements

The specifications for a local proximity sensor were derived by studying manipulation tasks in maintenance of live power lines and dynamic grasping of a hand tool, to be implemented on our SARCOS Dextrous Arm, a 7 dof robot arm with a three finger hand. The two most stringed requirements arise from the combination of a working range down to  $5mm$  and a size suitable for embedding in a human-size finger pad, limited to  $5mm$  diameter. By recessing the sensor in the finger pad, an effective range down to  $0mm$  can actually be achieved. Such an arrangement is also preferable for mechanical protection. Based on compliant finger pads, we estimate that an accuracy of  $1mm$  just before contact is sufficient, with decreasing accuracy requirements as distance increases. In order to support low level sensor-based control, high sampling rates of at least  $1kHz$  are required.

Working Range	5-100 mm
Size	5mm Diam., 5mm Height
Accuracy	1mm (@ 5mm) – 10mm (@100mm)
Sampling Rate	1kHz

### 2.2 Options

For short range proximity sensing in robotic application, electro-optical sensors are at present the most appropriate. There are three basic approaches – triangulation, phase and amplitude based systems. While triangulation and phase based systems are more independent from ambient light, surface material properties and surface orientation, they typically require a base separation between transmitter and receiver larger than  $5mm$  which makes them ineligible for our purposes. When the *magnitude* of the reflected energy is correlated to proximity, the transmitter and receiver pair can be placed as close together as physically possible, and we have built a prototype with less than  $3mm$  diameter. In our approach, we are going to employ several sensors (one sensor pair per finger pad), rely on signal processing to reduce ambient light dependence and exploit the knowledge of the object geometries to achieve the accuracies required.

### 2.3 Object Localization Strategy

We calibrate the sensors to obtain the sensor output as a strictly monotonic function of the distance  $d$  and the surface angle  $\alpha$ ,

$$v = f(d, \alpha).$$

From the simultaneous measurements of two sensors which are a distance  $d_0$  apart, we obtain two equations with four unknowns,

$$v_1 = f(d_1, \alpha_1)$$

$$v_2 = f(d_2, \alpha_1).$$

In the case of partially known environment, two additional equations can be added from the object geometry and we can solve the resulting system of equations. The parameters  $d_1$ ,  $d_2$ ,  $\alpha_1$  and  $\alpha_2$  describe the location and

orientation of the object with respect to the sensor pairs. Without restriction of generality, we assume here that the two sensor pair lie in a plane. In general, this is not the case, for example, when each one is located in a separate finger pad. We now derive the three simple geometries for flat, rectangular and round object shapes.

**Locally Flat Surface:**

$$\alpha_1 = \alpha_2 = \alpha$$

$$\tan\alpha = \frac{d_2 - d_1}{d_0}.$$

From two measurements  $v_1, v_2$  and  $d_0$  (Figure 1) we can solve for  $d_1$  and  $d_2$ .

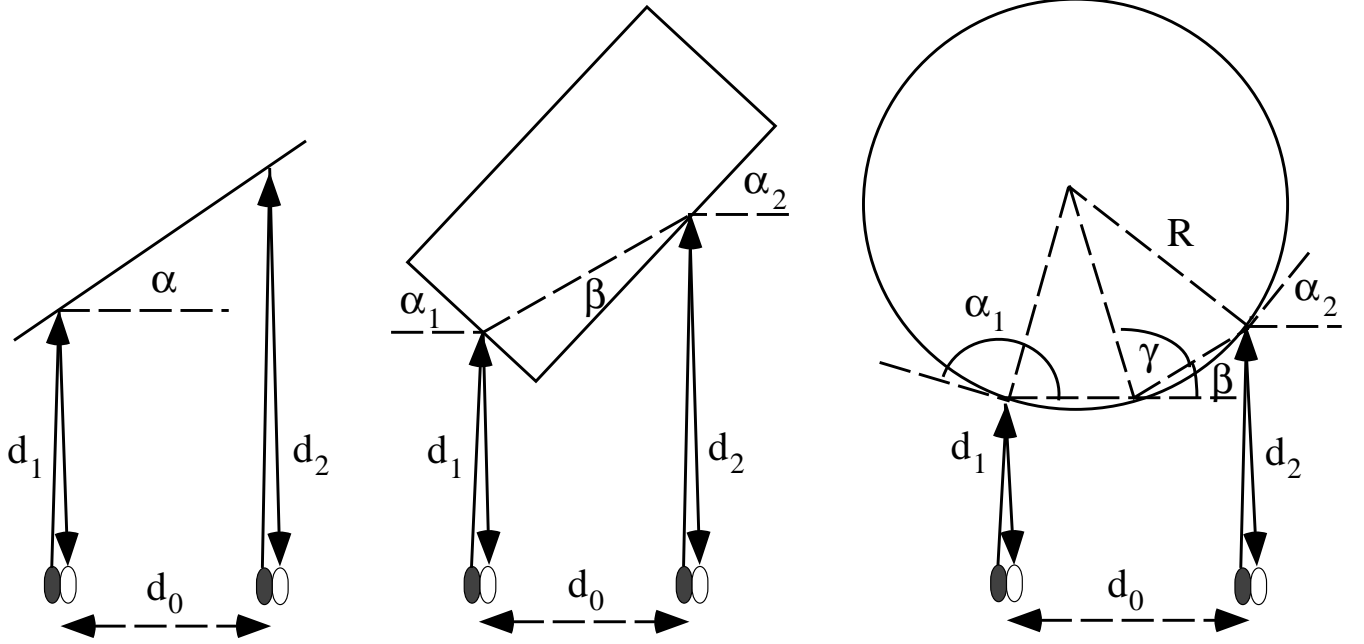


Figure 1: Sensing a Planar, Rectangular and Circular Object

**Rectangular Object:**

For a rectangle (Figure 1), the added equations are

$$\frac{\pi}{2} - \alpha_1 + \alpha_2 = 0$$

$$d_2 - d_1 = d_0 \tan(\alpha_2 - \beta),$$

$$\sin\beta = \frac{(d_0 - \frac{d_2 - d_1}{\tan\alpha_2}) \sin\alpha_2}{\sqrt{d_0^2 + (d_2 - d_1)^2}}.$$

**Circular Surface:**

In case of a circular or cylindrical object (Figure 1), we use the same method and obtain the additional equations

$$d_0 = 2R \cos(\alpha_1 - \frac{\pi}{2}) + \frac{d_2 - d_1}{\tan\beta},$$

$$d_2 - d_1 = 2R \cos\gamma \sin\beta,$$

where  $\beta = \frac{\pi}{2} - \frac{\alpha_1 - \alpha_2}{2}$ ,  $\gamma = \pi - \frac{\alpha_1 + \alpha_2}{2}$ .

This technique provides an effective method for locating the object from IR sensors, and is currently being extended to the spatial case with a larger number of sensors and more complex object geometries. The resulting system of equations are highly nonlinear and have to be solved online via efficient numerical methods.

## 2.4 Object Localization Experiment

For our object localization experiments we have selected a readily available and inexpensive transmitter/receiver pair, the Optek OP290A GaAlAs IR LED and the OP804SL photo transistor. Each device measures  $1.3mm$  diameter and  $3mm$  height, and satisfies the size constraints set out earlier. With a given albedo parameter of the object, in our case white paper, sample data showing the variation in sensor data as a function of object distance and orientation is shown in Figure 2.

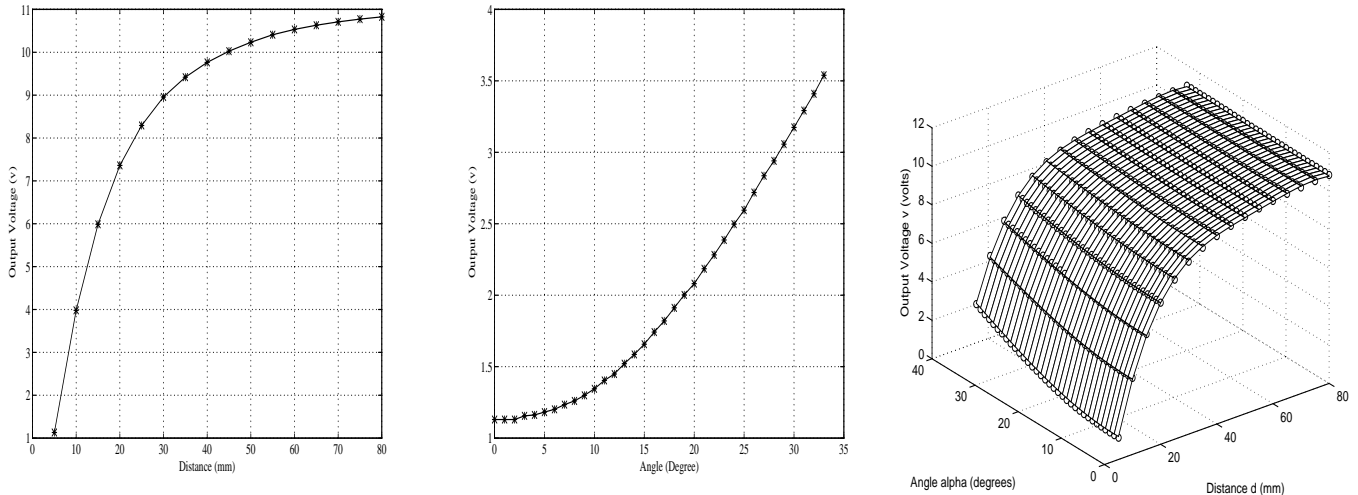


Figure 2: Output Voltage Curve vs. Distance and Angle. Surface fitting (grid) of the experimental data (o).

By logging the sensor output while scanning a range of distances from  $5mm$  to  $80mm$  and surface orientations from  $0$  to  $33deg$  the experimental surface plot shown in Figure 2 was obtained. The same plot shows the surface fitting results.

Based on this sensor characterization, we have implemented the object localization experiment for a planar surface as described above. From the two sensor measurements  $v_1, v_2$ , the sensor pair separation  $d_0$ , and the sensor characterization  $f(d, \alpha)$  we estimate the object pose using Matlab as  $\hat{d}_1 = 4.82mm, \hat{d}_2 = 20.08mm, \hat{\alpha} = 15.25deg$ . With a true pose of  $d_1 = 5mm, d_2 = 20mm, \alpha = 15deg$ , the errors in proximities  $d_1 - \hat{d}_1 = 0.18mm, d_2 - \hat{d}_2 = -0.08mm$  are well within our requirements.

## 3 Dynamic Grasping based on Geometric Controllers

Once we succeed in obtaining proper task-oriented sensing, surprisingly simple feedback controllers can be employed. In the context of dynamic grasping, “proper task-oriented sensing” means, for example, an increasing sensing accuracy and rate as the object approaches the robot end effector. In the remainder of this paper, we will review a controller strategy which translates relative cartesian position and velocity information between robot and object into an online reference trajectory, which, if tracked, will result in a smooth contact with few assumptions on the object trajectory. A more detailed account of this approach has been presented in [22].

### 3.1 Problem Formulation

Let

$$\mathbf{r}_d = \begin{bmatrix} r_{d,x} \\ r_{d,y} \\ r_{d,z} \end{bmatrix}, \mathbf{r} = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \in \mathbb{R}^3$$

represent the desired and actual robot end effector position and the actual object position, respectively. The end point position error vector is

$$\mathbf{e}_b = \mathbf{r}_d - \mathbf{b}$$

and the tracking error

$$\mathbf{e}_r = \mathbf{r}_d - \mathbf{r}.$$

We translate the temporal specification for making a grasp contact into a geometric one by assuming that the contact time  $t_c$  is the time when the object traverses the horizontal “grasp surface”

$$b_z(t_c) = 0. \quad (1)$$

Depending on the task and the robot’s workspace, other surfaces can be selected.

A **smooth grasp** of a moving object is defined as a match between the position, velocity and acceleration of the object and the robot’s end effector on the grasp surface, i.e.

**Condition 1:**

$$\mathbf{e}_b(t_c) = \dot{\mathbf{e}}_b(t_c) = \ddot{\mathbf{e}}_b(t_c) = \mathbf{0}. \quad (2)$$

In addition, in order to avoid pre-grasp collisions between robot and object, we require that

**Condition 2:**

$$r_{d,z}(t) < b_z(t) \quad \forall t < t_c. \quad (3)$$

Just like in the related literature, the tracking problem is not directly addressed here. However, in order to facilitate tracking, the initial desired and actual robot positions, velocities and accelerations at  $t_0$  should match,

**Condition 3:**

$$\mathbf{e}_r(t_0) = \dot{\mathbf{e}}_r(t_0) = \ddot{\mathbf{e}}_r(t_0) = \mathbf{0}. \quad (4)$$

Since we are focussing on the online trajectory generation problem, we also make the following assumptions:

1. The inverse kinematics, possible singularities and interference with the workspace boundaries and other obstacles are dealt with elsewhere. There are several solutions available, for example [10].
2. An integrated sensing and processing system provides the object’s cartesian position and velocity relative to the robot’s end effector. A system based on real time stereo vision has been successfully implemented for robot juggling ([21]).
3. The robot end effector is equipped with a gripper which can be closed when the smooth grasp conditions are satisfied.

### 3.2 Motivation

Surprisingly, the basic ideas for simple geometric feedback controllers came from a seemingly complex task domain - robot juggling. A successful geometric controller for robot juggling is described in [6] where they were applied to juggling one and two balls in the plane [5]. This concept was extended further by Rizzi and Koditschek [21] to spatial juggling of two balls with a three degree-of-freedom direct drive robot, guided by a real time stereo vision system.

It turns out that a simplified version of the geometric controller for juggling can accommodate a velocity and acceleration matching contact with a moving object at a desired position. To illustrate the basic idea, consider the trivial 1 dof prismatic robot in Figure 3 with vertical coordinate  $r$  and an object moving along a line, with height  $b$ .

A geometric controller for the desired robot height  $r_d$  to smoothly grasp the object would be

$$r_d = \kappa \cdot \text{atan}(b/\kappa), \quad (5)$$

where  $\kappa > 0$  is a user selectable gain. For purposes of improved tracking, the desired robot velocity is available to the PD controller as well:

$$\dot{r}_d = \kappa \frac{\dot{b}/\kappa}{1 + (b/\kappa)^2}. \quad (6)$$

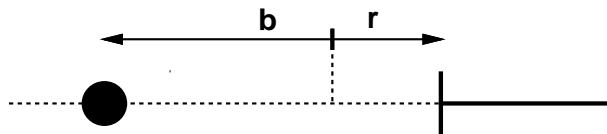


Figure 3: 1 dof illustration

Assuming perfect tracking,  $r(t) = r_d(t)$  robot-object contacts can only occur when  $r = b = 0$ , and at that point a velocity matching contact is guaranteed,

$$\dot{r}_c = \dot{r}(r = b = 0) = \kappa \frac{\dot{b}/\kappa}{1 + (b/\kappa)^2} = \dot{b}_c.$$

The object's acceleration is matched as well:

$$\ddot{r}_c = \ddot{r}(r = b = 0) = \frac{\ddot{b}}{1 + (b/\kappa)^2} - \frac{2(b/\kappa^2)\dot{b}^2}{(1 + (b/\kappa)^2)^2} = \ddot{b}_c.$$

The controller is called geometric since it contains no dynamics itself, it is simply a map from current object phase space to a desired current robot state. All the planning to ensure smooth grasping is done off line, when designing the controller. Notice that this approach completely eliminates the online trajectory planning in the traditional sense. The task is encoded in (5) and is achieved by tracking the set-points computed by the same equation at every instant. No assumptions were made about the object's trajectory, and the algorithm will succeed as long as tracking is achieved. In the next section we will generalize this basic controller (5) for dynamic grasping in six degrees-of-freedom.

### 3.3 A Geometric Controller

There are two major shortcomings in the simple geometric catch controller (5). First, it needs to be extended for spatial grasping. This is non-trivial, since a straightforward addition of similar controllers for the additional degrees of freedom does not guarantee time synchronization such that all degrees of freedom satisfy the smooth grasp conditions at the same time. Second, at  $t = 0$  there might be a large initial error  $e_r$  between the commanded and the actual robot position, which leads to large transient torque requirements and might ultimately prevent a sufficiently small asymptotic tracking error at time of contact  $t_c$ , which would violate Condition 3 (4). It turns out that there exist many functions that would result in a velocity and acceleration matching contact, in a similar fashion to (5), and a few can even satisfy the additional conditions (4) and (3). However, the construction of polynomials is the simplest solution and allows a procedure equivalent to spline generation. In the sequel we will derive a spatial geometric controller based on polynomials of the object position for the vertical and the horizontal components.

#### Vertical Component

A quintic function of the vertical object position satisfies all the grasping conditions:

$$r_{d,z}(b_z) = \sum_{n=0}^5 \kappa_{z,n} b_z^n = q_z(b_z). \quad (7)$$

The three gains  $\kappa_{z,0}, \kappa_{z,1}, \kappa_{z,2}$  can be obtained from Condition 1 (2). For the vertical robot position at contact we obtain from (7)

$$r_{d,z}(t_c) = q_z(b_z(t_c)) = q_z(0) = \kappa_{z,0}$$

and to satisfy the grasping condition (2)

$$r_{d,z}(t_c) = b_z(t_c) = 0$$

requires  $\kappa_{z,0} = 0$ . Similarly, the velocity and acceleration conditions result in  $\kappa_{z,1} = 1$  and  $\kappa_{z,2} = 0$ . The remaining three gains  $\kappa_{z,3}, \kappa_{z,4}, \kappa_{z,5}$  are derived from the three initial conditions (4):

$$r_{d,z}(t_0) = q_z(b_{z,0}) = r_{z,0}$$

with

$$r_{d,z,0} = r_{d,z}(t_0), \quad r_{z,0} = r_z(t_0).$$

The velocity and acceleration initial conditions result in

$$\dot{q}_z(b_{z,0}) = \dot{r}_{z,0}, \quad \ddot{q}_z(b_{z,0}) = \ddot{r}_{z,0}.$$

The solution for the last three gains can be expressed as

$$\begin{bmatrix} \kappa_{z,3} \\ \kappa_{z,4} \\ \kappa_{z,5} \end{bmatrix} = \mathbf{B}_{z,0}^{-1} \cdot \begin{bmatrix} r_{z,0} - b_{z,0} \\ \dot{r}_{z,0} - \dot{b}_{z,0} \\ \ddot{r}_{z,0} - \ddot{b}_{z,0} \end{bmatrix}$$

where

$$\mathbf{B}_{z,0} = \begin{bmatrix} b_{z,0}^3 & b_{z,0}^4 & b_{z,0}^5 \\ 3b_{z,0}^2 \dot{b}_{z,0} & 4b_{z,0}^3 \dot{b}_{z,0} & 5b_{z,0}^4 \dot{b}_{z,0} \\ 6b_{z,0} \dot{b}_{z,0}^2 + 3b_{z,0}^2 \ddot{b}_{z,0} & 12b_{z,0}^2 \dot{b}_{z,0}^2 + 4b_{z,0}^3 \ddot{b}_{z,0} & 20b_{z,0}^3 \dot{b}_{z,0}^2 + 5b_{z,0}^4 \ddot{b}_{z,0} \end{bmatrix}. \quad (8)$$

Subscript 0 refers to the initial condition. We have now obtained the desired robot trajectory which satisfies the grasping conditions (2) and (4). Notice that this computation is performed only once at  $t = 0$ , and subsequently only the evaluation of the polynomial (7) is required (in addition to its derivative for tracking purposes).

If the initial object velocities and accelerations are not available, a third order polynomial is sufficient to match the initial position only. In this case,  $\kappa_{z,3} = (r_{z,0} - b_{z,0})/b_{z,0}^3$  and the geometric controller for the vertical component takes on the extremely simple form

$$r_{d,z} = b_z + \kappa_{z,3} b_z^3. \quad (9)$$

Note that Condition 2 (3) is satisfied by (9) as long as it is satisfied at the initial time,

$$\kappa_{z,3} = (r_{z,0} - b_{z,0})/b_{z,0}^3 < 0.$$

### Horizontal Components

In this section, the algorithm for online generation of a robot reference trajectory is developed which achieves velocity and acceleration matching for the  $r_x$  and  $r_y$  components. The critical task here is that this matching must be accomplished at the object's spatial position and **at the same time**  $t_c$  when the vertical component is contacting the object. This is an illustrative example how purely geometric controllers are used to achieve time synchronization. Time synchronization between two processes has been achieved once before with purely geometric controllers, however in an entirely different fashion. In [7] it was critical for juggling two balls which required the control of a certain phase relationship between them. For brevity, only the  $r_x$  component is developed. The algorithm is identical for the  $r_y$  component.

In order to satisfy the grasping conditions at  $t_c$  and the initial condition at  $t_0$  in the x coordinate, a quintic polynomial of the **combined** ball positions in x and z coordinates must be used.

$$r_x = b_x + \sum_{n=3}^5 \kappa_{x,n} b_z^n \quad (10)$$

satisfies all the requirements, since on the contact surface  $b_{z,c} = 0$ , and thus

$$\dot{r}_{x,c} = \dot{b}_{x,c} + \dot{b}_{z,c} \sum_{n=3}^5 n \kappa_{x,n} b_{z,c}^{n-1} = \dot{b}_{x,c} \quad (11)$$

$$\begin{aligned} \ddot{r}_{x,c} &= \ddot{b}_{x,c} + \dot{b}_{z,c}^2 \sum_{n=3}^5 n(n-1) \kappa_{x,n} b_{z,c}^{n-2} + \ddot{b}_{z,c} \sum_{n=3}^5 n \kappa_{x,n} b_{z,c}^{n-1} \\ &= \ddot{b}_{x,c}. \end{aligned} \quad (12)$$

The solution for the last three gains can be expressed from (10), (11), (12) and Condition 3 (4) as

$$\begin{bmatrix} \kappa_{x,3} \\ \kappa_{x,4} \\ \kappa_{x,5} \end{bmatrix} = \mathbf{B}_{z,0}^{-1} \cdot \begin{bmatrix} r_{x,0} - b_{x,0} \\ \dot{r}_{x,0} - \dot{b}_{x,0} \\ \ddot{r}_{x,0} - \ddot{b}_{x,0} \end{bmatrix}$$

with  $\mathbf{B}_{z,0}$  as defined in (8).

Again, if only initial position matching is required, we can use the third order polynomial version of (10). The constant  $\kappa_{x,3}$  is determined as above,  $\kappa_{x,3} = (r_{x,0} - b_{x,0})/b_{z,0}^3$  and

$$r_{d,x} = b_x + \kappa_{x,3} b_z^3. \quad (13)$$

Notice that there is no interference problem since the vertical controller assures Condition 2, namely that the robot's end effector is always below the object. So there are no spatial restrictions on the horizontal components.

### Orientation

For any practical implementation, the end effector orientation, angular velocity and acceleration should be matched to the object's at time of contact as well. In addition, the initial desired end effector orientation should match the actual orientation. In the Cartesian case this is implemented in the geometric controller framework by specifying the Euler angles of gripper ( $r_\phi, r_\theta, r_\chi$ ) with respect to the base coordinates as a polynomial of the object's Euler angles w.r.t. the gripper frame ( $b_\phi, b_\theta, b_\chi$ ) expressed in the base coordinates and the vertical object component  $b_z$

$$r_i = b_i + \kappa_{i,3} b_z^3; \quad \kappa_{i,3} = \frac{r_{i0} - b_{i0}}{b_{z0}^3} \quad i = \phi, \theta, \chi. \quad (14)$$

Figure 4 illustrates the planar case in the vertical x-z-plane. Corresponding robot-object trajectory points are connected via a dashed line, and the robot's end effector orientation is indicated by a short solid line. Notice how the initially large orientation error is eliminated by (14) to match the object's angle, angular velocity and angular acceleration with respect to the robot's end effector.

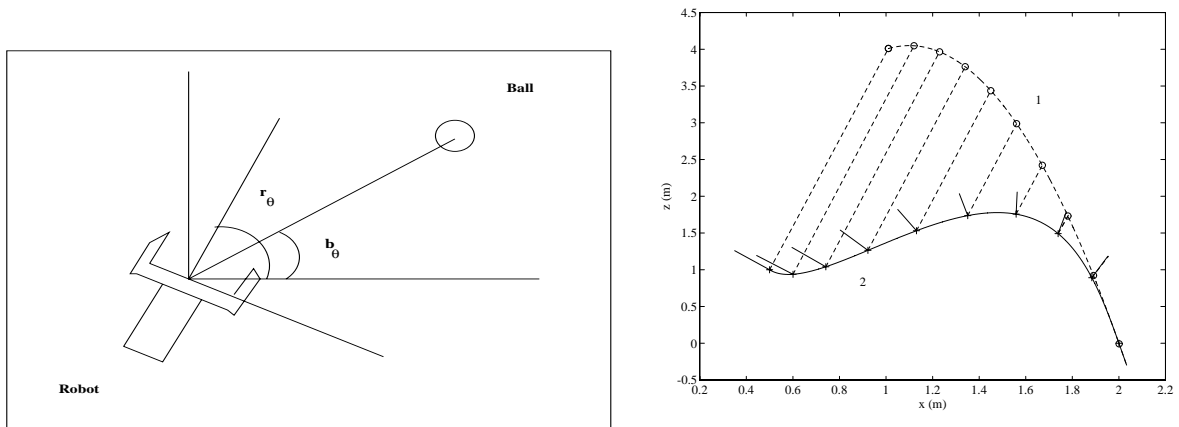


Figure 4: End Effector Orientation Control Variables, and Orientation Control for Parabolic Object Trajectory, 1: object, 2: robot

### 3.4 Comparisons

In this section we compare the performance of the geometric control approach to smooth grasping as described above to the more conventional approach based on trajectory planning using intercept predictions based on current and past state measurements. Both versions, which assume either constant velocity or acceleration for estimating the future object trajectory are evaluated. For ease of exposition we limit ourselves to the planar problem, and endpoint position only. The results for the full spatial case with orientation would be analogous. Therefore the geometric control is completely described by equations (9) and (13). The results are shown in Figure 5 for a parabolic and a sinusoidal object trajectory, respectively. In both plots the object trajectory is trajectory 1 (dashed line), the geometric controller is trajectory 2 (solid line), the replanning approach based on constant acceleration object prediction is trajectory 3 (dashed dot) and the one on constant velocity object prediction is trajectory 4 (dot).

Overall, the robot trajectories generated by all three approaches are quite similar, despite the drastic difference between the replanning and the geometric controller. However, the geometric controller performs more closely to the better one of the replanning approaches. For example, for the parabolic object trajectory (Figure 5), the replanning approach assuming constant object acceleration (3) is predicting correctly. In this case the geometric controller (2) results in almost the same trajectory. In the case of the sinusoidal object trajectories (Figure 5), the replanning approach assuming constant object acceleration (3) predicts incorrectly, and the constant velocity (4) replanning approach results in a shorter, smoother trajectory. In this case, the geometric approach (2) results in a robot reference trajectory similar or better than the constant velocity approach.

## 4 Conclusion

We have selected an electro-optical local proximity sensor which satisfies the size, range, accuracy and sensing rate requirements. The major drawback of the amplitude sensing method, the angle dependence of the measurements, are avoided by exploiting the knowledge of the object geometry. We are now extending the planar case with two sensor pairs presented in this paper to the spatial case with a larger number of sensors located in multiple finger pads. By integrating such a sensor network with a long range machine vision system, continuous object proximity information is available. To utilize this information for smooth dynamic grasping we present a sensor based approach which generates online robot trajectories. The new geometric controller is based only on current object state measurements and requires no information about the object dynamical model nor its future trajectory.

## References

- [1] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Trajectory filtering and prediction for automated tracking and grasping of a moving object. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1850–1856, May 1992.

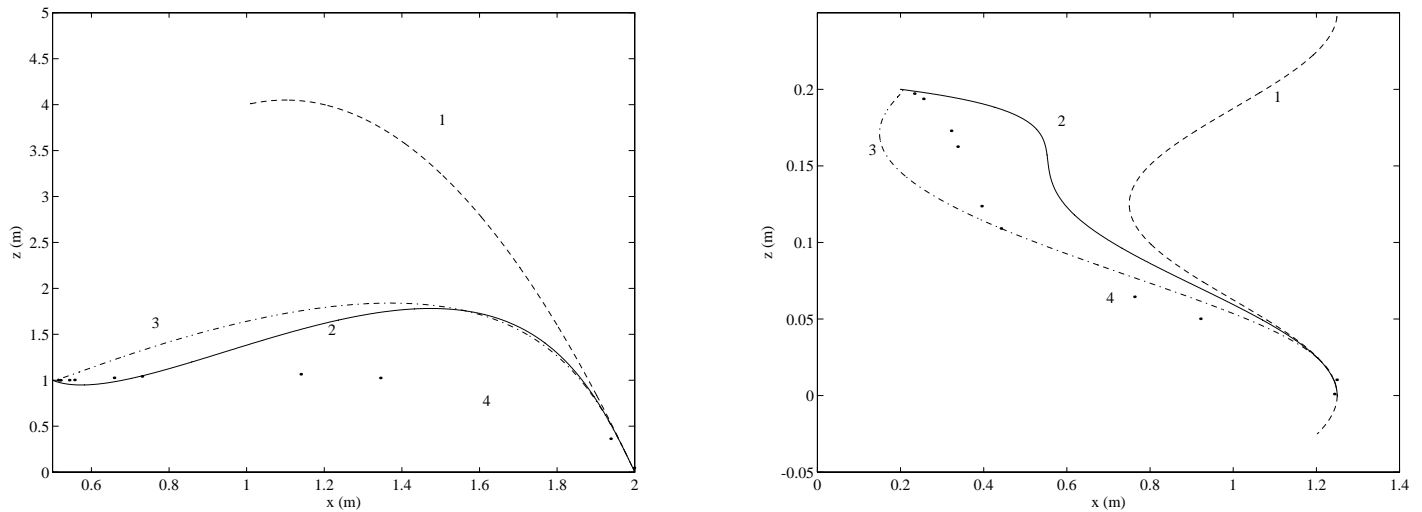


Figure 5: Parabolic and Sinusoidal Object Trajectories (1: object, 2: geometric controller, 3: replanning, const. acceleration, 4: replanning, const. velocity)

- [2] R. L. Andersson. Real time intelligent visual control of a robot. In *IEEE Workshop on Intelligent Control*, Aug 1986.
- [3] D.J. Balek and R.B. Kelley. Using gripper mounted infrared proximity sensors for robot feedback control. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 282–7, Silver Spring, MD, March 1985. IEEE Computer Society Press.
- [4] A. K. Bejczy. Effect of hand-based sensors on manipulator control performance. *Mechanism and Machine Theory*, 12(5):547–67, 1977.
- [5] M. Buehler, D. E. Koditschek, and P. J. Kindlmann. A Simple Juggling Robot: Theory and Experimentation. In V. Hayward and O. Khatib, editors, *Experimental Robotics I*. Springer-Verlag, 1989.
- [6] M. Buehler, D. E. Koditschek, and P. J. Kindlmann. A Family of Robot Control Strategies for Intermittent Dynamical Environments. *IEEE Control Systems Magazine*, 10(2), Feb 1990.
- [7] M. Buehler, D. E. Koditschek, and P. J. Kindlmann. Planning and Control of Robotic Juggling and Catching Tasks. *Int. J. Robotics Research*, 13(2):101–118, 1994.
- [8] G. Buttazzo, B. Allotta, and F. Fanizza. Mousebuster: A robot for real-time catching. *IEEE Control Systems Magazine*, 14(1), Feb 1994.
- [9] H. A. Ernst. *MH-1, a computer-operated mechanical hand*. PhD thesis, M.I.T., Dec 1961.
- [10] C. Gosselin, J. Côté, and D. Laurendeau. Inverse kinematic functions for approach and catching operations. *IEEE Trans. Systems, Man, and Cybernetics*, 1993, to appear.
- [11] N. Houshangi. Control of a robotic manipulator to grasp a moving target using vision. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1990.
- [12] B. Hove and J.-J. E. Slotine. Experiments in robotic catching. In *American Control Conference*, 1991.
- [13] A.R. Johnston. Proximity sensing technology for manipulator end effectors. *Mechanism and Machine Theory*, 12:95–109, 1977.
- [14] Y.F. Li. Characteristics and signal processing of a proximity sensor. *Robotica*, 12:335–41, 1994.
- [15] Z. Lin, V. Zeman, and R. V. Patel. On-line robot trajectory planning for catching a moving object. In *Proc. IEEE Int. Conf. Robotics and Automation*, 1989.
- [16] J. Lloyd and V. Hayward. Trajectory generation for dynamic, sensor-driven environments. *Int. J. Robotics Research*, 12(4), 1993.
- [17] J. Marszalec. A proximity sensing system for an intelligent optically-powered robot gripper. *International Journal of Optoelectronics*, 4(3/4):343–55, May-Aug 1989.
- [18] R. Masuda. Multifunctional optical proximity sensor using phase modulation. *Journal of Robotic Systems*, 3(2):137–47, 1986.
- [19] B. Benhabib O. Parta and A. Goldenberg. Analysis and design of a robotic distance sensor. *Journal of Robotic Systems*, 10(4):427–45, 1993.
- [20] J.Y. Catros A. Dore B. Espiau M. Parent and J.Y. Yclon. Automatic grasping using infrared sensors. In *Proceedings of the 8th International Symposium on Industrial Robots*, pages 132–42, Stuttgart, Germany, May 1978.
- [21] A. A. Rizzi and D. E. Koditschek. Further progress in robot juggling: The spatial two-juggle. In *Proc. IEEE Int. Conf. Robotics and Automation*, Atlanta, Georgia, May 1993.
- [22] M. Zhang and M. Buehler. Sensor-based online trajectory generation for smoothly grasping moving objects. In *Int. Symp. Intelligent Control*, pages 141–146, Columbus, OH, Aug 1994.