AD-A256 226

DTIC
ELECTE
OCT 8 1992
S D
C

Real-Time Gaze Holding in Binocular Robot Vision

David J. Coombs

Technical Report 415
June 1992

92-26689

UNIVERSITY OF
ROCHESTER
COMPUTER SCIENCE

Real-time Gaze Holding in Binocular Robot Vision

by

David John Coombs

Submitted in Partial Fulfillment

of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

Supervised by Christopher M. Brown

Department of Computer Science

University of Rochester

Rochester, New York

June 1992

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist Avail and/or Special
A-1

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1992	technical report

4. TITLE AND SUBTITLE
Real-Time Gaze Holding in Binocular Robot Vision

5. FUNDING NUMBERS
N00014-82-K-0193

6. AUTHOR(S)
David J. Coombs

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Science Dept.
734 Computer Studies Bldg.
University of Rochester
Rochester, NY 14627-0226

8. PERFORMING ORGANIZATION REPORT NUMBER
TR 415

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Information Systems
Arlington, VA 22217

DARPA
1400 Wilson Blvd.
Arlington, VA 22209

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution of this document is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Using a binocular, maneuverable visual system, a robot that holds its gaze on a visual target can enjoy improved visual perception and performance in interacting with the world. This dissertation examines the problem of holding gaze on a moving object from a moving platform, without requiring the ability to recognize the target. A novel aspect of the approach taken is the use of controlled camera movements to simplify the visual processing necessary to keep the cameras locked on the target. A gaze holding system on the Rochester robot's binocular head demonstrates this approach. Even while the robot is moving, the cameras are able to track an object that rotates and moves in three dimensions.

14. SUBJECT TERMS
visual following; tracking; pursuit; vergence; active vision; animate vision; precategorical vision; gaze stabilization; sensory-motor systems; egomotion compensation; mobile robots

15. NUMBER OF PAGES
154

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
unclassified	unclassified	unclassified	UL

Acknowledgments

I would like to first thank my advisor, Chris Brown. He has always been an incredible source of inspiration, and I have learned a great deal from him; this work has benefited in countless ways from his guidance. My committee has contributed in many ways as well. Dana Ballard has often helped me see things from a less-myopic perspective than is my wont. Randal Nelson's technical insights have always been keen. Mike King has graciously suffered my naive questions about primate eye movements and physiology, and has soberly listened to my clumsy analogies with robotic systems.

Parts of this work were developed in collaboration with colleagues at Rochester. In particular, Chapter 3 describes the vergence work, done with Tom Olson, that provides the foundation for the rest of the work. Chapter 5 reports empirical studies of some ideas for improving Zero-Disparity Filtering that were carried out by Peter vonKaenel under supervision of Chris Brown and myself. Finally, Chapter 6 describes the results of attempts by Chris Brown and myself to relate some common control methods for coping with delays in sensory-motor systems.

The members of the robotics and vision group at Rochester have made the lab a great place to work and have always offered lively discussions. I would especially like to thank Amit Bandopadhay, Paul Chou, Paul Cooper, Polly Pook, Rob Potter, Ray Rimey, Jeff Schneider, Mike Swain, Lambert Wixson, and Brian Yamauchi. It has been a pleasure to work with John Soong and Peter vonKaenel. Dave Tilley, Tim Becker, Liud Bukys, and Jim Roche provided a rich software and hardware environment which I will miss. Tom Olson was a terrific co-author and mentor, who always had not only time for a question, but also a patient smiling answer.

My life in the department has been enriched by the faculty, staff and students in ways too numerous to mention. In addition to the department, the Center for Visual Science has had a tremendous influence on me both scientifically and personally. I will have to be satisfied with blanket thanks for most of the wonderful ways people in Rochester have touched my life. However, I will especially thank Keri Jackson, Brian Marsh, Lata Narayanan, Cesar Quiroz, Neil Smithline, and Steve Whitehead for being good friends throughout it all.

I need to extend special thanks to Martin Herman and Ernest Kent at NIST for their patient consideration and encouragement in the final stages of completion of this document.

I thank my family for the support and encouragement they have always given me. I thank my parents Les and Molly for their love and guidance. I thank my siblings, Jean,

Joe, and John, for teaching me to love in the midst of strife. I thank Robert Dodd for his continual support and encouragement and his calm unbiased eye.

I reserve my final thanks for my best friend, Melissa Coombs, who makes each day worthwhile. She has given me both quiet support and stern encouragement, and somehow known when I needed each.

This material is based upon work supported by the National Science Foundation under Grants numbered IRI-8903582, CDA-8822724, and IRI-89220771, and by ONR/DARPA research contract number N000114-82-K-0193.

Abstract

Using a binocular, maneuverable visual system, a robot that holds its gaze on a visual target can enjoy improved visual perception and performance in interacting with the world. This dissertation examines the problem of holding gaze on a moving object from a moving platform, without requiring the ability to recognize the target. A novel aspect of the approach taken is the use of controlled camera movements to simplify the visual processing necessary to keep the cameras locked on the target. A gaze holding system on the Rochester robot's binocular head demonstrates this approach. Even while the robot is moving, the cameras are able to track an object that rotates and moves in three dimensions.

The key observation is that visual fixation can help separate an object of interest from distracting surroundings. Camera vergence produces a horopter (surface of zero stereo disparity) in the scene. Binocular features with no disparity can be extracted with a simple filter, showing the object's location in the image. Similarly, an object that is being tracked will be imaged near the center of the field of view, so spatially-localized processing helps concentrate on the target. Rochester's binocular robot exploits these observations. The vergence and smooth tracking systems cooperate to hold the cameras on an object moving in three dimensions. The vergence system changes the vergence angle of the cameras to drive the disparity of the target to zero, relying on the tracking system to keep the target in the central field of view. The tracking system centers the cameras on the zero-disparity signals, relying on the vergence system to hold vergence on the target. Instead of requiring a way to recognize the target, the system relies on active control of camera movements and binocular fixation segmentation.

KEYWORDS: visual following, tracking, pursuit, vergence, active vision, animate vision, precategorical vision, sensory-motor systems, gaze stabilization, egomotion compensation, visual fixation, binocular vision, mobile robots.

Table of Contents

List of Figures

1 Introduction

The importance of eye movements to biological visual systems is obvious from their ubiquity. Even insects exhibit eye movements, although they are accomplished by head movements [Land, 1975]. In contrast, controlled camera movements have played a small role in computer vision research. However, there is growing interest in the role of camera movements in robotic visual perception, and some lessons for computer vision systems may be learned from studying biological vision systems. Both have limits on resolution and field of view, and they must therefore direct their visual sensors toward areas of the environment that are of interest. Also, both animal and robot visual systems exist to provide visual perception of the dynamic environments in which their owners operate.

There is a growing trend in computer vision to consider the visual system in the context of the behavior of a robot interacting with a dynamic environment. The *active vision* approach [Krotkov, 1989; Aloimonos *et al.*, 1988; Bajcsy, 1988] observes that constraints derived from known camera motion can replace other assumptions (*e.g.*, smoothness) that had previously been employed to solve mathematically ill-posed problems. *Animate vision* [Ballard, 1991] considers that visual perception is one of several behaviors employed by a creature in order to achieve its goals in a dynamic environment. Interest in animate vision has been sparked at least partly by the recent availability of real-time image processing equipment, which has enabled researchers to consider visual perception as a viable sensory input to a robot interacting with a dynamic world. Thus vision has begun to be considered as a means for gathering information that is relevant to the task in which the robot is engaged.

One of the principal tenets of animate vision is that sensing and motor behavior interact closely with one another. In this highly synergistic relationship, sensors provide perception to inform the creature's behavior, and motor actions make the creature an animate observer, using its sensors to maximum advantage. A corollary of this idea is that the sensor and the motors that move it must be considered together to arrive at the description of the perception system. One way of viewing this is to consider each creature as a sensory-motor system, consisting of perception, control, and effectors.

This dissertation considers the application of this idea to the problem of holding gaze on a moving object from a moving platform. We move through a world of still

and moving objects, and we need to look at them, recognize them and keep our eyes on them. This dissertation demonstrates that deliberate control of the cameras can simplify the sensory processing that is required.

1.1 Precategorical Gaze Holding

Holding gaze is a fundamental capability of biological visual systems. There are several visual perception and behavioral motivations for holding gaze in robotic as well as biological systems, including foveal vision, and motion blur.

Foveal Vision: Foveal visual systems have small areas of high resolution. To get a high resolution image of an object, the fovea must be directed at the target. Obviously, high resolution stereoscopic vision can only be achieved with both foveas directed at the point of interest simultaneously.

An argument can be made for the ultimate necessity of non-uniform resolution, in order to provide both high resolution and a wide field of view [Tsotsos, 1988]. Thus future robot systems may be equipped with foveas. If so they will require vergence systems for the same reasons that humans require them. Work on spatially-variant visual sensors has begun [Van der Spiegel *et al.*, 1989; Rojer and Schwartz, 1990], so we can expect to use camera systems with foveas in the near future.

Motion Blur: Motion blur degrades the resolution of moving images. An image that moves over the retina[1] will be degraded by motion blur, according to the integration time of the receptors (which have limited spatiotemporal resolution). It is easy to demonstrate this to yourself by moving your finger in front of some text and following your finger with your eyes. The background blurs as a result. If you fixate on the background, your moving finger will appear blurry. You can clearly read the text behind your finger.

Facilitating Stereo Fusion: In addition, active vergence control can facilitate stereo fusion. By definition, the fixation point has a stereoscopic disparity of zero, and points nearby tend to have small disparities. This makes it possible to use stereo algorithms that accept only a limited range of disparities. Such systems can be very fast and are amenable to hardware implementation [Nishihara, 1984; Olson, 1991].

Indexical Behavior: A consequence of actively following the target with the sensor is that simple visual sensing techniques that are uniquely available in this situation can be used to segment the target. This opens the possibility of developing

[1] The *retina* is the name of the inner surface of the back of the eye that contains the photoreceptors. We will also use the term "retina" to refer to the receptor array of a camera.

a suite of *indexical* behaviors [Agre and Chapman, 1987; Whitehead and Ballard, 1990] that the robot uses to interact with "the object of regard", such as `go-to-the-fixated-object`, and `pick-up-the-fixated-object`.

For gaze holding to be as general as possible, it should not require object recognition. It is time-consuming to recognize or locate an object. Doing so can provide robust information, but only at the expense of long latencies. Further, it is important to able to follow a moving object whose identity is not known. In summary, what is needed is the ability to follow an object with the cameras, without a model of the object. Of course, if the object can be located using a model, this reliable information should be used when it is available.[2] Nevertheless, there must be a faster system that relies only on *precategorical* cues (*i.e.*, cues available prior to object recognition) to keep the cameras on the target in the meantime.

The goal of this work has been to build a robot gaze holding system whose only knowledge of the target is essentially that the cameras are initially pointed at it. The situation is depicted in Figure 1.1. The gaze holding problem is to maintain fixation on a moving visual target from a moving platform. In order to do this, the errors in camera orientation must be determined, so the location of the target's image on the retina must be found. How can this be done without recognizing the target? The approach taken in this work exploits binocular cues and the fact that the cameras are actively following the target.

1.2 Control of Sensors and Simplified Sensing

As a consequence of gaze holding, the visual target is easier to pick out. Thus, it is easier to follow an object with moving cameras than to track its images in stereo images with static vergence and no control of camera movement [Coombs *et al.*, 1990]. For instance, during active following, motion blur de-emphasizes the background. Further, simple visual sensing techniques that are uniquely available during gaze holding can be used to segment the object being fixated, as illustrated in Figure 1.2. Foveal vision emphasizes the fixated object simply by spatially localized processing or enhanced resolution. Disparity filtering picks out features near the robot's *horopter* (the set of points in the scene whose stereo disparity is zero in the robot's cameras).[3] The demonstration system locates the target by foveally filtering the output of the zero-disparity filter (ZDF), essentially producing the intersection of the fovea and the robot horopter. The

[2] It should be noted that most of the time, the visual system of an animal (especially a higher mammal) will be pursuing an object that it is able to recognize, so the visual system likely has access to such reliable information about the target, albeit with some latency.

[3] The robot's horopter is here intended to be roughly analogous to the human empirical horopter. The relation between these horopters and zero-disparity filtering is explored in Section 5.1.

Binocular Gaze Geometry

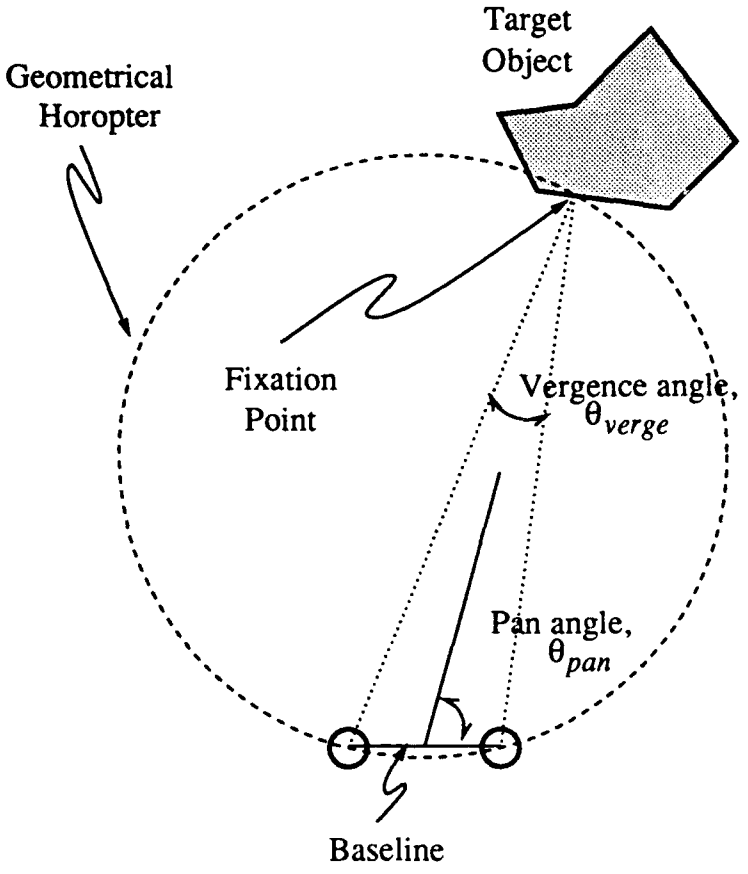


Figure 1.1: Top view of binocular gaze geometry. The goal of gaze holding is to keep the eyes or cameras fixed on a common world point or visual target. The gaze vector, $\vec{\theta}$, consists of the gaze pan and tilt angles and vergence angle. In order to keep the world point fixated, the gaze holding system must generate gaze and vergence angles that keep the cameras directed toward the target. The result of fixating a target is that the object lies near the robot's *horopter*, which is the set of world points whose disparity is zero. The stereo images of an object that lies near the horopter have a narrow range of disparities.

target's retinotopic location provides the error signals the gaze control system needs to control the gaze and vergence angles.

1.3 Overview

The remainder of the dissertation first motivates and outlines the problem of holding gaze in robots using visual cues alone. General issues facing a visual tracking system are considered in Chapter 2. These include the extraction of error signals from the visual signal and the interaction between the observation of visually encoded signals and the control of visuomotor systems that use them.

The gaze holding problem is comprised of pursuing the visual target with the cameras, and for binocular systems, verging the cameras on the object. We will call the direction of the cameras the gaze angle or direction. The pursuit system rotates the cameras in tandem to keep gaze directed toward the target. The vergence angle is the angle between the visual axes of the cameras. Vergence system control consists of rotating the cameras in opposite directions so the visual axes of the cameras intersect at the distance of the object of interest. The vergence and pursuit problems are discussed and demonstration systems are described in Chapters 3 and 4. The pursuit demonstration system is predated by and built upon the vergence system, so the vergence system is presented first.

The demonstration system exploits binocular cues and deliberate control of the cameras that enable precategorical segmentation of the fixation target. These ideas are embodied partly in the zero-disparity filter (ZDF), which reveals only objects that lie at the fixation distance. The set of points that stimulate the ZDF defines the robot's horopter; in this way it is somewhat analogous to the human empirical horopter. This relationship is explored briefly in Chapter 5, which relates our experiences with zero-disparity filtering.

One of the problems faced by sensory-motor systems is the difficulty of controlling systems that contain delays. Delays are unavoidable since processing the signals takes time. Methods for coping with delays are described in Chapter 6. Finally, Chapter 7 summarizes the results and describes some directions future investigations might take.

It should be made clear that the scope and goals of this work extend only to holding gaze on a visual target that is already fixated by the robot's cameras. Further, it is the express intention to use only precategorical cues (*i.e.*, prior to object recognition). This leaves open the questions of how to select a visual target and how to initially acquire fixation of it. Both of these capabilities are clearly essential for a fully functional gaze control system. While the approach taken is inspired somewhat by biological models and the questions they raise (*e.g.*, how to visually extract the target signals), the design of the demonstration system bears little resemblance to biological models of smooth pursuit. In particular, it is not widely believed that binocular cues play a significant

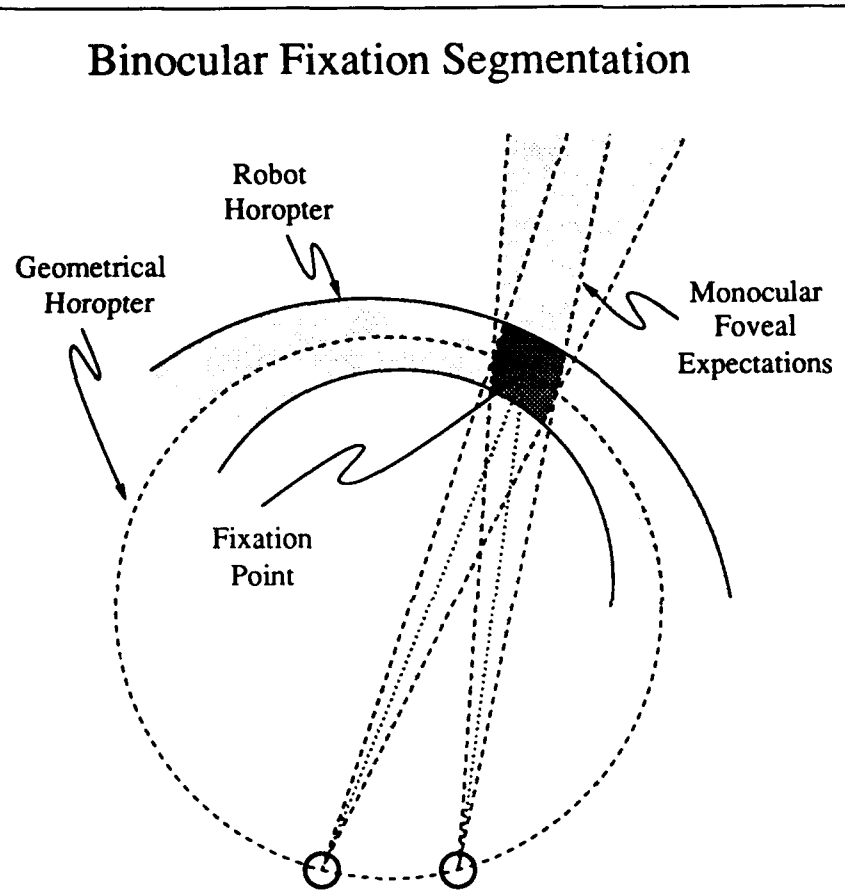


Figure 1.2: In this top view of binocular fixation, the lightly shaded areas are the regions of space that are attended to by foveal and zero-disparity filtering. The intersection of these areas is shaded darker, and it corresponds to the area around the fixation point in which objects can be segmented. (The robot's horopter is here defined by the results of zero-disparity filtering, and the "thickness" of this horopter arises from the tolerance of the ZDF to small disparities.)

role in pursuit and visual following, though they may influence stabilization of gaze on stationary scenes. Nevertheless, the central thesis, that localizing attention in 3D space makes simple precategorical visual processing sufficient to hold gaze, remains relevant. Although binocular cues cannot distinguish a visual target in all situations, an appropriate conjunction of precategorical cues offers promise for holding gaze in dynamic environments.

2 Gaze Holding and Background

The larger problem of gaze control can be broken down functionally into the subproblems of *holding gaze* on a visual target and *shifting gaze* between objects. Gaze shifts, called *saccades*, transfer fixation rapidly from one visual target to another. Holding gaze involves maintaining fixation on a moving object from a moving gaze platform. The general gaze holding problem is to maintain fixation on a moving visual target from a moving platform. In the case of binocular visual systems, this involves keeping the visual axes of the eyes directed at the target (as illustrated in Figure 1.1), and can be decomposed into *version*, when both eyes pan and tilt together, and *vergence*, when the eyes pan in opposite directions to fixate nearer or farther targets.

Gaze holding has several aspects that are accomplished in animals by a set of specialized controls that are mentioned here and sometimes referred to by analogy later. The *smooth pursuit* system tracks continuous target motion. The *vestibulo-ocular reflex (VOR)* and *otolith-ocular reflex* systems rotate the eyes to compensate for head motion that is detected by the vestibular and otolith organs in the inner ear. The *opto-kinetic reflex (OKR)* and other *visual following* mechanisms stabilize the eyes on stationary scenes. The *vergence* system rotates the eyes in opposite directions, controlling the distance of the fixation point (at which the optic axes intersect). The *saccadic* system provides fast gaze shifts. In animals these systems cooperate in complex ways [Carpenter, 1988].

Vision alone provides the measure of how well gaze is being held steady on an object. Non-visual cues (*e.g.*, head motion) can be used to stabilize gaze on a fixed point in the world. Sometimes such cues are available with low latency and minimal sensing and processing, leading to higher performance from stabilization systems that use them. However, only vision can provide cues to the motion of a moving object. Therefore, this work focuses on the problem of using visual cues alone to hold gaze on a moving object from a moving platform.

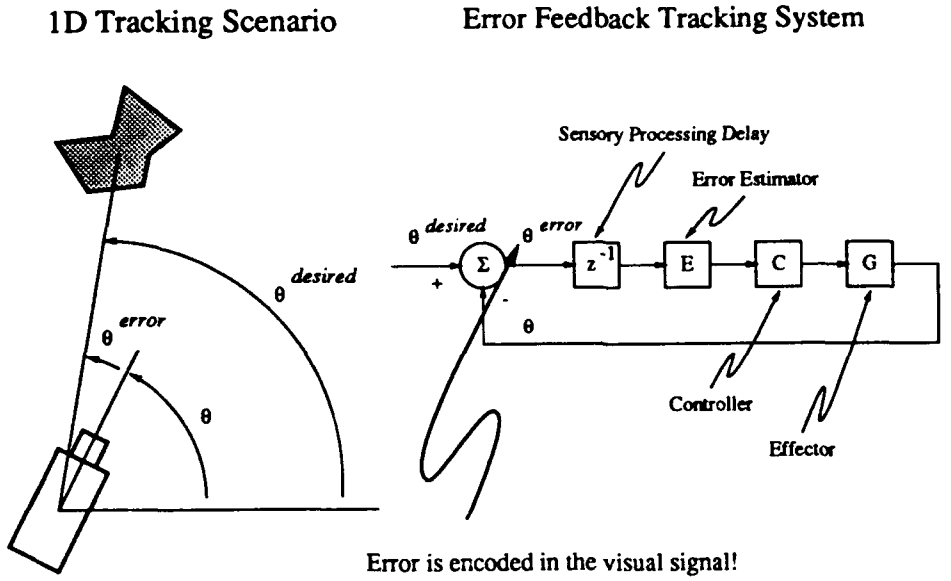


Figure 2.1: Simple 1D tracking example: In the one-dimensional setup of (a), the camera should be pointed at the object. The visual system is modeled in the diagram (b) by the delay (z^{-1}) and the error estimator, E. The controller, C, converts that error to a set of control signals that direct the camera motor (*i.e.*, the plant, G) to produce appropriate changes in θ.

2.1 Tracking Systems

Before discussing the vergence and pursuit systems, consider a simple one-dimensional visual tracking task (a) and the tracking system modeled by the block diagram (b) of Figure 2.1. The objective of the system is to follow the desired angle, $\theta^{desired}$, which represents the gaze angle the camera must achieve to fixate the target object. The system's response, the actual gaze angle, is denoted by θ. The difference between the desired and actual response is the error in tracking performance, θ^{error}. At the most abstract level, any error feedback solution to a visual following problem will have three major components: a *sensory system* that determines how the current response differs from the ideal, a *controller* that generates a response to the errors, and a *motor system* that executes the controller's commands. The visual system is modeled in the diagram by the delay (z^{-1}) and the error estimator, E. The controller, C, converts that error to a set of control signals that direct the camera motors (*i.e.*, the plant, G) to produce appropriate changes in θ.

It is important to note that in a visual tracking system, the error is encoded in the visual signal. *I.e.*, although θ and $\theta^{desired}$ are scalar, θ^{error} represents image data! Thus, the error estimation consists of visual processing to determine the error in the system's response.

In a visually-driven oculomotor servo system, the sensor that provides the driving signal is being moved, and therefore the observation of the error signal is directly affected by the oculomotor behavior. *I.e.*, the observation of the error signal and the control of the system are not separable since they interact. This is very important, since the error estimator and the control algorithm must be designed with this interaction in mind. Consider that the error in gaze direction is encoded by the retinal location of the target. Moving the camera to correct any positional error will cause the target image to slip across the retina. This may consequently blur the image of the target, potentially causing difficulty for subsequent visual processing. Conversely, the control system faces the dilemma of trying to correct both positional and velocity errors simultaneously. One solution is simply to sacrifice one to some extent and only try to control the other. For instance, a control system that attempts to position the target's image on the fovea with smooth movements will sacrifice image stability to achieve the desired position. Similarly, a velocity-matching system will not correct sustained positional errors. It is commonly assumed that the primate oculomotor system's solution to this dilemma is that the pursuit system matches the target's velocity with smooth movements of the eyes, and it uses catch-up saccades to correct positional errors. However, it is also possible to elicit "compromise" behavior from monkeys in which smooth eye movements alone attempt to minimize both velocity and position errors [Lisberger, 1990].

This tight coupling between observation and control is especially crucial if there are delays in the system, and since computer programs and neural computation (especially visual processing) do not execute instantaneously, delays are unavoidable. Suppose the tracking system needs to follow a target that jumps to a new position, and that there is a delay in estimating the error. The error is not apparent to the control system immediately, but once it is observed the system responds. The system's response is not immediately noticed due to the processing delay, so the response to the old observed error is sustained. Consequently, the system over-reacts to the input signal. If the delay is longer, the system will react to older sensory data, and the resulting overshoot will be greater. If the delay is too long with respect to the system's response time, the system as a whole will be unstable. This behavior is evident in the performance of the control systems of the demonstration implementation, exemplified by the vergence system in Chapter 3.

A simple way to prevent the system from over-reacting is to reduce its responsiveness. A better approach incorporates knowledge of the delay in the system to avoid over-responding, but the response is still late. The only way to avoid being late is to predict the target signal's behavior in advance. This is possible for predictable (*i.e.*,

modelable) signals, or for signals that can be approximated by linear predictive filters (*e.g.*, a Kalman filter [Bar-Shalom and Fortmann, 1988]). It is important to make the predictions in a coordinate system that is not perturbed by the control system so the target signal will be stable in that space. For instance, the target trajectory may be relatively smooth in head-centered coordinates, and yet the target's image may not move smoothly across the retina since the retinal slip of the image depends also on the camera movements. These issues are touched on briefly in Chapter 4 and considered in more depth in Chapter 6.

2.2 Gaze Holding Overview

A critical parameter in the design of a gaze holding control system is the nature of the input signal—how the desired gaze angles (pan, tilt, and vergence, as described in Figure 1.1) change with time. This will depend on both the motion of the target and the movement of the observer and its cameras. It seems reasonable to base initial discussions on the known characteristics of primate eye movements [Carpenter, 1988]. That is, camera movements will generally consist of intervals of smooth pursuit or fixation of an object punctuated by discontinuous jumps (saccades) to new visual targets, and continuous-mode gaze controls should be designed with this in mind.

The two types of expected changes in desired gaze angles differ in fundamental ways. During pursuit and fixation, changes in desired angles are determined by the dynamics of observer and object motion. The laws of physics restrict what can happen; for example, accelerations and velocities must be finite. A target may cross the visual field with high velocity, requiring high speed panning of the cameras to pursue it. However, rapid changes in desired vergence angle will be rare: they arise from very rapid target movements in depth (especially near the observer), so the target will soon pass through the image plane (if it is approaching) or recede to a depth at which desired vergence angle changes more slowly. In short, the input to the vergence control loop during pursuit and fixation will be smooth, with finite second derivatives and small first derivatives. The situation for the pursuit input is similar to the vergence system. However, the target's retinal position can be expected to change faster than its disparity. Therefore the pursuit system will be expected to encounter faster changes in the desired pan and tilt angles than will the vergence system. Note that even with a smoothly moving target object, noise in the estimation of error signals can introduce discontinuities in otherwise smooth signals. The pursuit and vergence controls should be robust to such disturbances.

During a saccade the input signals will behave quite differently than during smooth gaze holding. If the pursuit and vergence control systems are to operate continuously, they must be robust to the discontinuities in the input signals that are introduced by saccades. At the very least, the knowledge that a saccade is occurring can be used as a

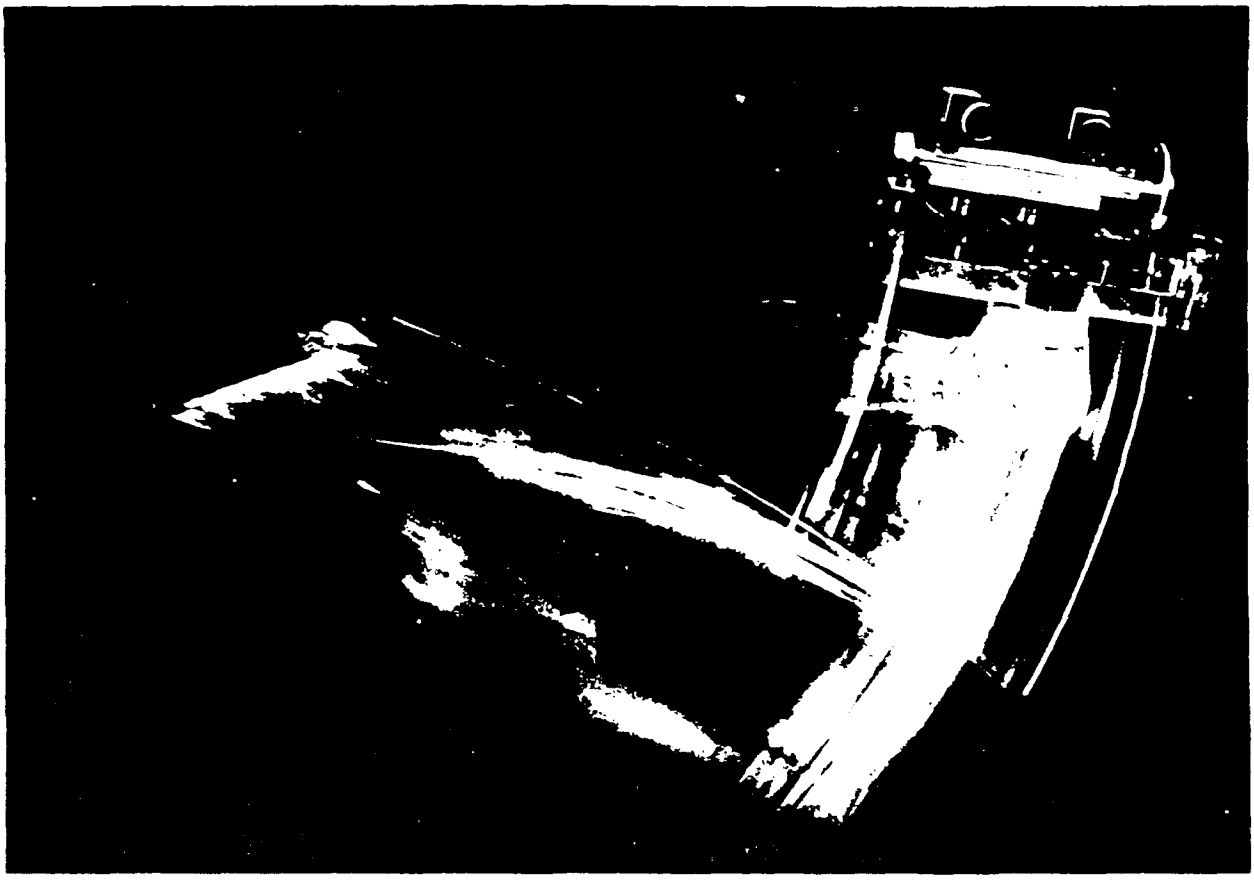
warning that discontinuities in the input signals are to be expected. Continuous systems could then reset their ongoing processes to prevent the use of expectations based on the behavior of the previous target object. However, since the focus of this dissertation is limited to following smoothly moving objects, there will be no detailed consideration of interaction between smooth and saccadic camera movements.

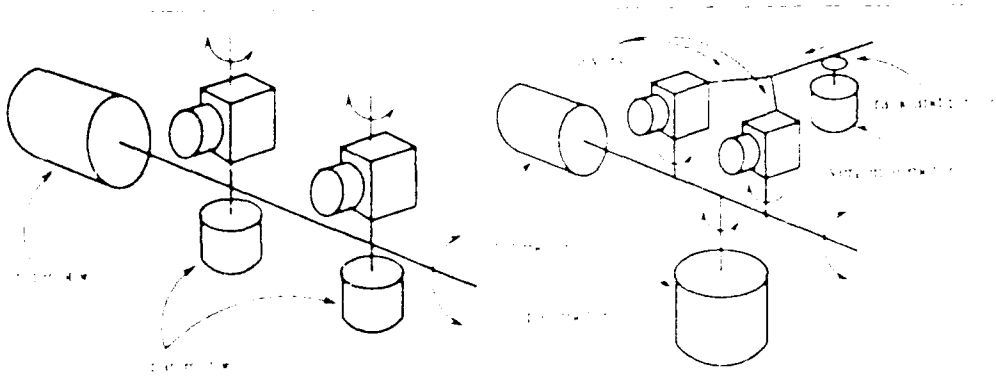
This dissertation presents a system implemented on the Rochester robot that demonstrates that gaze holding can be achieved using only *precategorical* visual processing (*i.e.*, prior to the level of object recognition). The gaze holding system is initially engaged with the cameras fixating the target and moving at the correct speed to keep the target fixated. (In all experiments described here, the target is initially still.) In addition, the target's approximate size is also supplied to the system. The system "identifies" the target only by the fact that the target is initially fixated (*i.e.*, its approximate location is known) and its approximate size is known. Despite the discussion above of the relations between saccades and smooth camera movements in holding gaze, the demonstration system uses only smooth camera movements and makes no provision for interacting with saccades or any other gaze control mechanism.

2.3 The Rochester Robot

The demonstration system is implemented on the Rochester robot head, which has three degrees of freedom (DOF), as shown in the photo of Figure 2.2. The head is mounted on the end of an industrial PUMA six DOF robot arm with a six-foot radius reach. The cameras can move at speeds exceeding 400 deg/s so they can approximate human saccade speeds. The cameras tilt up and down together on a common platform, and pan independently from side to side, driven by three motors, one for the tilt platform and twin pan motors, as sketched in Figure 2.3(a). A mechanical advantage of the Rochester head design is its simplicity: the compact mechanism and fairly direct linkages facilitate rapid saccades.

The Rochester head design differs from another common head design (*e.g.*, [Clark and Ferrier, 1988; Krotkov, 1989].) which is shown in Figure 2.3(b). This design has a motor that controls vergence directly, and the pair of cameras pan and tilt together as the entire mechanism is rotated. The vergence angle is controlled by a single motor that converges both cameras symmetrically via a mechanical linkage, such as a rack and pinion driving a pair of levers that rotate the cameras about vertical axes. The advantage of this design is that gaze angles and the vergence angle are controlled by separate motors and are orthogonal—gaze direction and vergence can be altered without disturbing one another. Thus, this design is well suited to employ traditional models of gaze control, in which vergence is considered to be largely independent of other types of camera movements. However, saccades may be more brisk with the Rochester design,





since only the cameras and tilt platform must be moved, whereas the shared vergence design requires that essentially the entire head must be rotated to shift the gaze.

The quality of the motor system, or plant, is determined by how quickly and faithfully it translates control signals into changes in camera angles. Current generation CCD cameras and motor controllers make it relatively easy to move the cameras quickly. Care is required, however, to insure that the camera mounting is able to tolerate the stresses generated by rapid camera movements. The large accelerations required for saccadic movements can cause "ringing", *i.e.*, vibrations that persist after the motors have come to a stop. Avoiding these problems involves mechanical engineering considerations that are beyond the scope of this dissertation, so they will not be discussed further.

Another important aspect of the mechanical design is the location of the axes of rotation with respect to the nodal points of the cameras. The nodal point of an optical system is the point about which a rotation of the system results in a change in the projected image that depends only on the rotation and not on the distance of the imaged point. The nodal point of the system corresponds to the pinhole of a pinhole camera or to the front nodal point of an ideal thick lens system [Horn, 1986]. For the purposes of gaze control it is desirable to mount the cameras so that their axes of rotation pass through the nodal points, since doing so makes it possible to predict the effects of a rotation without knowing the depths of objects in the scene.

If the axes of rotation do not pass through the nodal point of the camera system, each camera rotation necessarily includes a small translational component. This causes the displacement of the target in the image to vary with target depth as well as the amount of rotation. The depth dependence may of course be useful, *e.g.*, to constrain stereo correspondences [Geiger and Yuille, 1987], but it complicates the control problem by making the relationship between camera position and vergence error more complicated. However, it is difficult to design a system that rotates the cameras about their nodal points. The nodal points are often far from the centers of gravity of the cameras, and they may move when the lenses are changed or moved to adjust focus or zoom. Fortunately, the depth dependencies introduced by minor deviations from the ideal are negligible for feedback servo control purposes unless the targets are very close to the camera system. In the experiments we conducted, the effect is sometimes noticeable, but it was always relatively small.

The mechanical design of the Rochester Robot head is such that the nodal points of the cameras do not lie on the axes of rotation. This means that camera movements necessarily include a small translational component as well as the desired rotation, so that (for example) increasing the vergence angle slightly reduces the baseline of the camera system. For most purposes the translational movement can be ignored, although its effects were noticeable in the experiments described below.

More sophisticated binocular systems may have several other control parameters to consider. One is focus depth, and interactions between focus depth and vergence angle

have been explored recently in robotic vision experiments [Abbott and Ahuja, 1988; Krotkov, 1989]. Another possible degree of freedom is *torsion* or *roll*, *i.e.*, rotation about the optic axis. Torsional movements can be used to reduce retinal slip due to head rotations about an axis parallel to the gaze direction. Torsional stabilization in humans can be readily observed by watching one's eyes in a mirror while rocking one's head from side to side. For computer vision, torsion controls could be used to align camera scan-lines with the epipolar line. This alignment is often assumed by stereopsis algorithms.

Reduction gearing gives the cameras theoretical angular resolutions of $\frac{1}{278}$ degree in yaw and $\frac{1}{2500}$ degree in pitch. Inaccuracy introduced by gear lash reduces the usefulness of this resolution to an unknown degree, but camera positioning is still accurate to substantially better than the angle subtended by one pixel with the 16mm lenses that are commonly used, which yield a field of view of about 20 degrees.

Figure 2.4 describes the configuration of laboratory hardware on which the demonstration systems are implemented. The MaxVideo^{TM} captures images from the cameras and performs the bulk of the image processing. It is coordinated by the SUN host, which interprets the visual signals to produce the appropriate motor commands for the motor controllers.

The robot host computer sends commands to the motors via intelligent stepping motor controllers that allow the control program to issue commands in terms of absolute position, relative position, velocity or velocity profile. The ability to issue buffered velocity commands enables the control program to generate smooth movements without paying constant attention to the motors.

The mechanical design of the motor and camera system was dictated by a desire to perform saccadic movements at speeds comparable to those of humans. It seemed probable that a camera platform powerful and rigid enough to perform saccades quickly and without noticeable ringing would be able to handle the gentler movements required for vergence with relative ease. To date this has proven to be the case, and the performance of the pursuit and vergence systems has been limited by the speed of the error estimators (and therefore the achievable servo control rates) rather than the physical capabilities of the motor system.

Laboratory Hardware

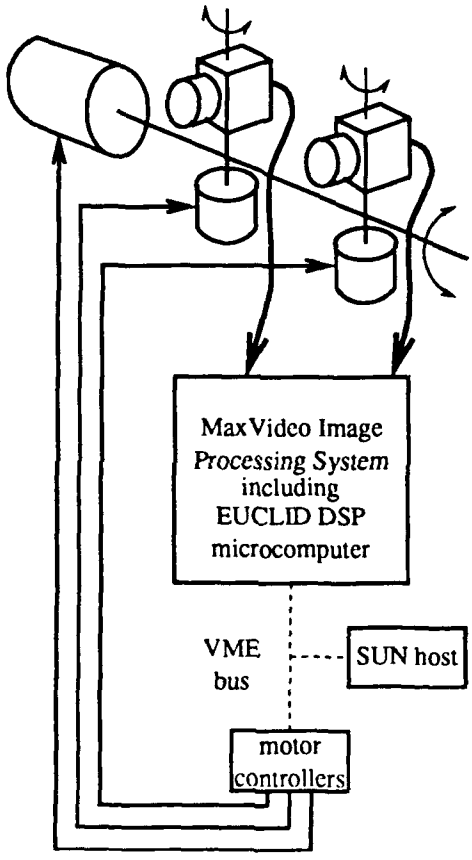


Figure 2.4:
Hardware configuration used for pursuit and vergence control experiments on the Rochester Robot.

3 Vergence

Natural systems are generally multi-ocular. Multi-ocularity confers the basic advantages of redundancy and increased visual field, and allows algorithms for stereo fusion (which is aided by vergence), static stereo and gloss detection (which rely on stereo fusion but not directly on vergence), and segmentation of an object from the rest of the scene (which relies on vergence but not fusion).

Vergence may be the result of a mechanism that foveates a target in both cameras (see Figure 1.1). Alternatively, vergence may arise from independent foveation mechanisms acting in each camera. For instance, with moving targets, vergence might emerge from independent smooth pursuit systems attempting to stabilize the target's images on each retina. Vergence methods that depend on identical independent behaviors in two different cameras depend on the behaviors being coordinated. For instance, the independent tracking scheme will not verge if left and right cameras lock on to different targets. Thus in this work it is assumed that vergence is a system that should be driven by robust two- and three-dimensional cues, including a representation of the three-dimensional fixation point, visual disparity (computed over an appropriate area of the visual field), and focusing information (which should in turn be affected by vergence).

This chapter describes work reported in [Olson and Coombs, 1991]. It describes the design, implementation and performance of a gaze control module responsible for controlling the vergence angle of the cameras. The next section discusses vergence in the abstract, presenting reasons for verging and issues that any vergence control system must address. This discussion leads to a general strategy for vergence control, described in Section 3.3. Sections 3.4, 3.5 and 3.6 describe the application of this vergence control strategy to the problem of controlling vergence on the Rochester Robot, and present empirical results on the performance of the error estimator and the overall vergence system.

3.1 The Vergence Problem

The *vergence angle* of a binocular system is the angle between the optic axes of its cameras. The vergence angle, baseline (or interocular distance) and gaze direction of a

binocular system determine a particular fixation point, as shown in Figure 1.1. Narrowly speaking, the function of the vergence system is to control the distance from the cameras to the fixation point along some specified gaze direction. In most cases the motivation for vergence is to keep the fixation point near some target object. Thus, the vergence problem can be defined as that of controlling the vergence angle to keep the fixation distance appropriate for the current gaze target. Since the desired vergence angle is directly related to target distance, any sensory cue to depth or depth changes may be useful to the vergence system. The most commonly used cues are binocular disparity and focus error, but other depth cues (such as motion, texture, and shading) can also be used, as can information about depth changes (*e.g.*, measured or predicted self motions, dilations or contractions of the visual field).

Vergence is one aspect of the larger problem of gaze control, which involves control of the gaze direction and focal distance as well. Vergence control must meet different demands during gaze shifting and gaze holding. During gaze holding, a change in the target position relative to the observer produces a smooth change in the desired vergence angle. A saccade transfers the fixation point almost instantaneously from one visual target to another, producing a step change in the desired vergence angle.

The treatment of vergence presented here reflects current models of vergence in primates. Primate visual systems exhibit sophisticated vergence responses that meet the varied demands of the vergence task. Recent experiments have challenged traditionally held views of ocular vergence and its control (*e.g.*, those of Yarbus [1967]). It has been thought that vergence changes are always smooth and much slower than smooth pursuit (tracking) movements, and that vergence changes required to shift gaze to a target are achieved by smooth vergence movements superimposed on conjugate (equal and symmetric) saccades in the two eyes. Under natural viewing conditions, however, Erkelens *et al.* [1989a; 1989b] observed in humans not only smooth vergence movements rivalling the speeds of other smooth eye movements, but also vergence changes mediated almost entirely by saccades that incorporated a vergence change explicitly rather than merely superimposing symmetric saccades on smooth vergence movements. Similar behavior has also been observed in monkeys [Maxwell and King, 1990].

These results acknowledge that the vergence problem can be considered to consist of *acquiring vergence* and *holding vergence* on an object or scene. The traditional model of the vergence system is a compromise, describing a single system that aims to accomplish both these goals, achieving robustness and generality and sacrificing optimality. The recent data of Erkelens *et al.* suggest that the control of vergence is accomplished in humans by a combination of control systems, including a substantial component of vergence control by the saccadic control systems as well as the traditional smooth vergence control. The demonstration system implements only the smooth vergence control component, as no saccadic control is implemented. The system must expect to encounter small steps in the desired vergence angle, and the smooth controller is

designed to be robust to them.

3.2 Related Work

There is relatively little work on binocular gaze control. Some of the work is modeled after the traditional model of gaze control, and other work is focused more on manipulating the parameters of binocular systems for building a model of a static scene.

Clark and Ferrier [1988] built a gaze control system based on the model described in [Robinson, 1987]. The system acquires and tracks white and black blobs using the first few moments and intensity value of each object. The vergence control follows the traditional model. However, the "disparity" estimate is not estimated directly from the visual signal, but rather it is inferred from the location of the target's image in each of the cameras. It is assumed that the target is sufficiently unique that it can be identified and located in each of the stereo images unambiguously. The disparity is the difference between the locations of the target in the two images. *I.e.*, the correspondence problem is assumed to insignificant enough to be easily solved by precategorical processing, and even without resorting to explicit object recognition.

Vergence has recently been used cooperatively with focus and stereopsis for surface reconstruction [Abbott and Ahuja, 1988] and active exploration of the environment [Krotkov, 1989]. Deliberate control of vergence has demonstrated advantages in both robustness of results and increased speed in stereoscopic processing. Krotkov and Abbott and Ahuja applied vergence in active vision systems in a stricter sense, and their goals are similar although their approaches differ. They combine stereo, vergence and focus to build precise range maps; vergence enables the systems to "foveate" areas of interest to obtain higher precision and confidence in their range estimates. The main difference in approaches is that Krotkov improves a rough full-field map (obtained initially by stereo) by active control of camera vergence and focus accommodation, while Abbot and Ahuja incrementally extend the partial map with each new fixation.

3.2.1 Why Verge?

Part of the motivation for studying vergence comes from an interest in human vision: human eyes verge, and we would like to know more about how they do it. The human visual system's need for a vergence control system is obvious, and follows from the extremely non-uniform spatial resolution of the photoreceptor array. Vergence movements allow humans to register an object of interest on the fovea (central, high-resolution region of the retina) of each eye, so that the greatest possible amount of information about the object can be extracted.

Currently most robot vision systems do not have foveas, and so the most obvious motivation for vergence control in humans does not apply to them. However, vergence

has many advantages even for systems without foveas.

Mathematical Simplification: Fixating an object of interest puts points on the object near the optic axis in both cameras. In some cases this permits the use of simplifying assumptions (*e.g.*, replacing perspective projection with orthography) that make analysis significantly easier. For example, Ballard and Ozcandarli [1988] used this fact to develop a simple and efficient kinetic depth estimator for systems that fixate.

Facilitating Stereo Fusion: By definition, the fixation point has a stereoscopic disparity of zero, and points nearby tend to have small disparities. This makes it possible to use stereo algorithms that accept only a limited range of disparities. Such systems can be very fast and are amenable to hardware implementation [Nishihara, 1984; Olson, 1991].

Useful Coordinate Systems: As Ballard [1989] argues, having a unique fixation point at the intersection of the visual axes defines a coordinate system that is related as much to the object being observed as it is to the observer. It is thus a step in the direction of an object-centered coordinate system.

3.3 Strategies for Vergence Control

At the most abstract level, any error feedback solution to the vergence problem will have three major components: a *sensory system* that determines how the current vergence angle differs from the desired vergence angle, a *controller* that generates a response to the error, and a *motor system* that executes the controller's commands. These three components can be mapped onto the traditional block diagram model of a feedback system, shown in Figure 3.1. The desired vergence angle, $\theta_{verge}^{desired}$, models the vergence angle needed to verge on the target object. The binocular visual signal, modeled by θ_{verge}^{error}, encodes the difference, $\theta_{verge}^{desired} - \theta_{verge}$, between the desired and current vergence angles. The visual system is modeled in the diagram by the delay (z^{-1}) and the vergence error estimator, E. The controller, C, converts that error to a set of control signals that direct the camera motors (*i.e.*, the plant, G) to produce appropriate changes in θ_{verge}. This section discusses general considerations in the design and use of these three components.

3.3.1 Controller

One of the most important factors influencing the design of a vergence control system is the nature of the input signal—how the desired vergence angle changes with time. This will depend to some degree on details of the system design, particularly the nature of the

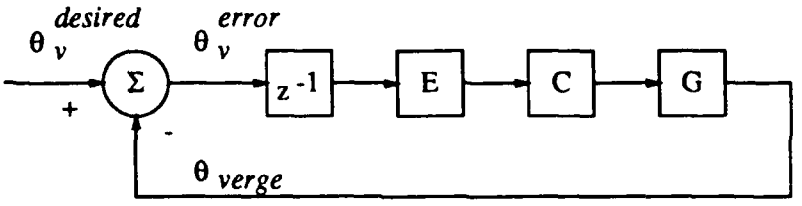


Figure 3.1: Vergence System Model. $\theta_{verge}^{desired}$ represents the desired vergence angle, and θ_{verge} is the actual vergence angle. The error in vergence angle, θ_{verge}^{error}, is encoded by visual signals, and the delay (z^{-1}) represents the delay due to visual processing. Process E represents the vergence error estimator, C the vergence control law, and G the oculomotor plant. The control law, C, is designed to drive the estimated error to zero.

processes responsible for other aspects of gaze control and movement (*e.g.*, pursuit and saccades). our expectations are based on the general view of gaze control described in Section 2.2. That is, in general camera movements are expected to consist of intervals of smooth pursuit or fixation punctuated by discontinuous jumps (saccades). Thus vergence behavior will consist of periods of maintaining vergence on an object or scene separated by vergence changes to acquire the vergence angle of a new visual target. The saccade is expected to account for most of the vergence change in shifting gaze between objects, but small steps in desired vergence angle must be expected, and the vergence system's behavior should be robust to these disturbances.

The two types of expected changes in target angle differ in fundamental ways. During pursuit and fixation, changes in target angle are determined by the dynamics of observer and object motion. The laws of physics restrict what can happen; for example, accelerations and velocities must be finite. Furthermore, although a target may cross the visual field with high velocity, rapid changes in target vergence angle will be rare. Rapid change in desired vergence angle corresponds to very rapid object movement in depth (especially near the observer), so the target will soon pass through the image plane (if it is approaching) or recede to a depth at which target angle changes more slowly. As a result, the input to the control loop during pursuit and fixation will be smooth, with finite second derivatives and small first derivatives.

During a saccade the input signal will behave quite differently. A saccade can produce a step change in the desired vergence angle, as well as a discontinuity in its temporal derivative. The magnitude of these changes may be predictable if the saccade is to a previously visited target, or if target depth and relative motion are approximately known from other depth cues. At the very least, the fact that a saccade is occurring can serve as a warning that discontinuities in the input signal are to be expected.

The fact that there are two distinct types of changes in desired vergence angle sug-

gests the use of two modes of control. The normal operating mode should be optimized for the smooth, continuous changes expected during pursuit and fixation. Saccadic movements should replace the smooth movements of the normal control loop with brief intervals of "bang-bang" control. That is, the estimated error should be corrected by an open-loop move to the new desired vergence angle at the maximum rate of which the motor system is capable. The open-loop vergence correction can be performed during the saccade, if the distance to the saccade target is known; if not there will be some delay while the error estimator determines a new target angle. The error estimator may need to be suppressed during a saccade, because of the possibility that motion blur and shearing deformations caused by camera movements may corrupt the results. Before the normal control loop is restarted, it should be reinitialized to prevent any tendency to smooth target angle velocity across the saccade. The details of how this is done will of course depend on the natures of the smooth control loop, the underlying hardware and the saccade generating process.

An alternative design reflects the traditional model of primate vergence control. This model uses a single control strategy that is able to respond to both kinds of expected changes in desired vergence angle. This simple design can stand alone, decoupled from other gaze control mechanisms, achieving robustness at the expense of optimality. Notwithstanding the discussion of possible organizations for a vergence system in the larger context of the gaze control systems, for our purposes of building a simple demonstration system consisting strictly of smooth camera movements and with no provisions for saccades or any other gaze controls besides pursuit, we have chosen to use this simple design.

3.3.2 Error Estimator

In order to keep the cameras verged on a target, the vergence system must measure the current vergence error (and perhaps its derivatives). The most important source of this information is the visual system, but other sources may also be useful. We have already noted the possibility of predicting the error that will result from a saccade to a target of known depth. Vergence changes due to self motion can also be taken into account, either by making predictions based on planned, voluntary head movements, or by sensing head motions via the vestibular system (as in the human vestibulo-ocular reflex) or reading back the positions of a robot. However, note again that vision is the only source of information for target motion, and visual signals ultimately define vergence performance.

A number of different types of visual information are available for estimating vergence error. One feature that is correlated with desired vergence angle under ordinary conditions is blur, which has been used cooperatively with vergence and stereo to construct depth maps [Abbott and Ahuja, 1988; Krotkov, 1989]. Any depth cue can be used if the absolute vergence angle of the system is known, because desired vergence

angle is a function of target distance. Humans apparently make use of cues that may indicate change in depth, since changes in the size of a visual target induce transient vergence responses [Erkelens and Regan, 1984].

The most useful visual cue to vergence error, however, is binocular disparity. The mapping from disparity to vergence error is particularly simple, and (unlike monocular depth cues) does not require knowledge of the absolute vergence angle of the system. Reliable disparity estimates can be computed more easily and quickly than depth estimates, permitting shorter processing delays and simpler control strategies. These advantages may be reflected in the structure of the human vergence control system; although vergence in humans can be driven by a variety of cues, responses are much slower under monocular viewing conditions than they are when disparity information is available [Erkelens *et al.*, 1989b].

Disparity measurement has been studied extensively in the context of stereo depth reconstruction [Barnard and Fischler, 1982]. Unfortunately most of the disparity estimators used for stereopsis are poorly suited to the real-time vergence application. They are optimized for positional accuracy and density rather than for robustness, and depend on optimization of global criteria to yield a more robust disparity field. They cannot provide single disparity estimates without essentially solving the stereo problem, which entails considerable computational expense.

For real-time vergence what is needed is a simple algorithm that estimates a single disparity in a fixed amount of time. This narrows the field to image processing methods such as cross-correlation. Past attempts to use such methods for stereo depth recovery have uncovered many problems (see [Horn, 1986] for a review). However, a class of operators that are closely related to correlation appears to work quite well for vergence. These operators are described in Section 3.5 and Appendix A. If correlation methods prove too slow, related methods such as phase comparison [Jepson and Jenkin, 1989] may be suitable, provided that care is taken to detect and compensate for various predictable errors [Fleet *et al.*, 1989].

Handling Multiple Disparities

An important issue in correlation-based disparity estimation for vergence concerns handling scenes that have multiple disparities. It is tightly linked to the selection of sample window size. The sample windows must be large enough to handle the expected range of disparities, but this almost guarantees that multiple disparities will be present at least some of the time. Therefore, some additional processing will almost certainly be required to insure that the desired correlation peak dominates the output.

An obvious way to improve the performance of the error estimator is to filter the input images so that the target object is more prominent. If detailed information about the target location is available, the image can be multiplied by a mask that emphasizes

details near that location. A simpler approach is to let the target be designated implicitly by one of the cameras. That is, consider one of the cameras to be dominant, and define the region near its optic axis to be the target. This strategy requires only that the dominant camera's image be multiplied by a centrally weighted mask before correlating.

Multiple disparities must be handled in different ways depending whether vergence is to be acquired on a new target or vergence is merely being maintained on a target over time. In maintaining vergence, expectations of the target's disparity can be based on recent experience. However, there is no such experience on which to rely for acquiring a new target.

Even with the signal of the target being emphasized in the input to the disparity estimator, it is likely that multiple disparities will be present. The disparity estimator will be forced to choose the target disparity from among those present. Once the target disparity is identified and the goal is to maintain vergence on a visual target, hysteresis in the disparity estimate is desirable. The system should be able to use a prediction of the target disparity (*e.g.*, with a Kalman filter [Bar-Shalom and Fortmann, 1988]) to select it from among the detected disparities. Care must be taken to adjust the prediction for vergence changes and delays in the system. It is essential with linear and quadratic predictors (or estimators) that the target signal be predicted in a coordinate system in which it will be fairly stable. Otherwise the performance of the low order polynomial predictions will be poor.

Acquiring vergence after a gaze shift requires picking out the right disparity. A reasonable strategy in the absence of input from higher-level processes is to choose the strongest disparity signal, effectively treating vergence error estimation as an image registration problem. This approach simply chooses the strongest disparity signal. This disparity will be the one that best describes the shift between the images. In correlation-based methods, the signal strength is related to peak height, so this reduces to selecting the disparity that produces the largest peak.

3.4 Vergence on the Rochester Robot

The general considerations discussed in the preceding sections formed the basis for the demonstration system on the Rochester Robot. This section and the two that follow describe the sensory and control components of the system, and discuss its performance as measured in the laboratory.

A demonstration vergence system, diagrammed in Figure 3.2, is implemented on the Rochester robot head. Visual processing begins with foveally-processed visual signals. The vergence system keeps the cameras converged on the target, using an estimate of the binocular disparity between the foveal images to measure the vergence error. It should be noted that the target must remain prominently in both left and right foveal images.

The disparity estimator only provides information on *what* disparities are present in the images and not *where* in the images those disparities arise; therefore, the vergence system has no way of ensuring that the target remains within the foveae. The system generates camera vergence velocity commands, but the motors are not configured with those degrees of freedom mechanically. However, they are linearly related and the motor controller performs the conversion from camera coordinates to motor coordinates.

In the demonstration system, the cameras smoothly verge on the target by using a position servo to minimize the retinal disparity of the camera foveae. Early experiments used a PD control law, and later experiments (in conjunction with pursuit) have used a PI controller. Prediction has not been attempted to mitigate the effects of delay. For the system to predict disparities, the target should be predicted in a stable space; therefore target depth and not retinal disparity should be used, since control responses affect the latter but not the former coordinate system. This requires an estimate of the range to the target, which in turn requires the vergence angles to be calibrated.

3.5 Vergence Error Estimation

The vergence error estimator is based on disparity, since (as argued in Section 3.3) disparity is the most direct and reliable measure of vergence error. One approach to disparity estimation would have been to use the MaxVideo™ convolution hardware to compare central patches of one image with the other image by correlation. However, previous attempts in our lab to use that approach for tracking had encountered many difficulties [Brown *et al.*, 1988]. Instead the disparity estimator is based on the cepstral filter [Yeshurun and Schwartz, 1989]. The cepstral disparity estimator performed well from the beginning, but the reasons for its success were initially unclear. Our efforts to achieve a better understanding of the algorithm led us to an interpretation of the cepstral disparity estimator as one of a family of operators of which phase correlation [Kuglin and Hines, 1975] is a logical end point. This section describes the basic operation and performance of the cepstral disparity estimator. The reasons for its success and its relationship to phase correlation are explored in Appendix A.

3.5.1 Measuring Disparity with the Cepstral Filter

The *cepstrum* of a signal is the Fourier transform of the log of its power spectrum.[1] It was introduced in [Bogert *et al.*, 1963] as a tool for analyzing signals containing echoes. Such signals can be modeled as an original signal $S(t)$ convolved with a train of impulses, *i.e.*,

$$R(t) = S(t) * (\delta(t) + a_0\delta(t - t_0) + a_1\delta(t - t_1) + \ldots)$$

[1] This is sometimes referred to as the *power cepstrum* to distinguish it from the *complex cepstrum*, which is the Fourier transform of the complex log of the Fourier transform.

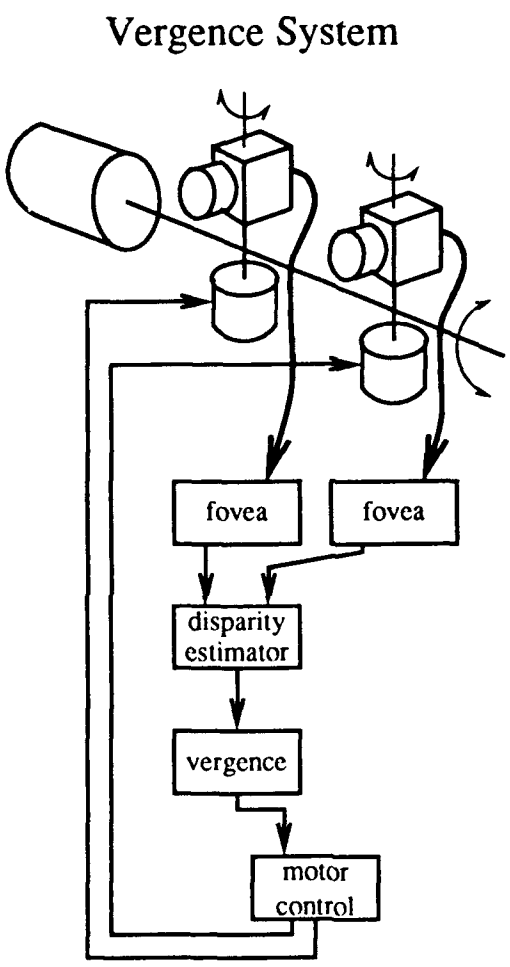


Figure 3.2: Vergence system: The visual signal is foveally processed before use. The vergence system estimates the disparity of the foveal images and controls the vergence angle to minimize the disparity. The motor controller translates vergence velocity commands by simple linear relations into the required left and right pan motor velocities.

where * denotes convolution. Taking the log of the power spectrum transforms the received signal into a sum of two terms, one of which depends only on $S(t)$ and the other of which is a combination of distorted sinusoids with frequencies related to t_0, t_1, *etc.* If the cepstrum of $S(t)$ does not overlap the frequencies of the echo terms, conventional linear filtering techniques can be used to extract the values of the echo delays.

Recently Yeshurun and Schwartz [1989] developed a way of using the two-dimensional cepstrum as a disparity estimator. The first step of their method is to extract sample windows of size $h \times w$ from the left and right images. The sample windows are then spliced together along one edge to produce an image of size $h \times 2w$. Assuming that the right and left images differ only by a shift, the spliced image may be thought of as an original image at $(0,0)$ plus an echo at $(w + d_h, d_v)$, where d_h and d_v are the horizontal and vertical disparities. The periodic term in the log power spectrum of such a signal will have fundamental frequencies of $w+d_h$ horizontally and d_v vertically. These are high frequencies relative to the window size. The image-dependent term, by contrast, will be composed of much lower frequencies, barring pathological images. Thus, as Yeshurun and Schwartz show, the cepstrum of the signal will usually have clear, isolated peaks at $(\pm(w + d_h), \pm d_v)$.

3.5.2 Implementation

Early experiments with a workstation-based implementation of the cepstral disparity estimator showed that it was robust enough for the vergence application, provided that the sample windows were of adequate resolution. At low sampling densities the estimator failed frequently, finding peaks that did not correspond to any plausible disparity in the scene. These failures usually resulted in anomalous vertical disparities and were therefore easily detectable. However, they introduced unpredictable latencies into the error estimation process. In order to gain the highest possible speed for real-time performance, we chose the smallest sampling density that gave reliable disparity estimates. This density was found empirically to be 32×32, obtained by subsampling over central 256×256 regions of the left and right input images.

Unfortunately, even at this greatly reduced resolution the original implementation required a few seconds per computation on the Sun that acts as the robot's system controller. In order to obtain a faster sample rate of disparity estimates the algorithm was re-implemented on the MaxVideo image processing system. The system is diagrammed in Figure 3.3. The processing begins by digitizing a stereo pair of images from the robot head's cameras. The cameras are synchronized so the images are taken simultaneously. The stereo images are convolved with anti-aliasing filters (*e.g.*, Gaussian, $\sigma = 2.5$ pixels for 8-fold reduction in resolution) before being stored in frame buffer memory.[2] The

[2] For only slight reductions in resolution, blurring is not necessary and is therefore omitted. Ideally,

EUCLID digital signal processing microcomputer included in the MaxVideo image processing system is used for estimating disparity. The EUCLID computer is based on the ADSP-2100 digital signal processor [Analog Devices, 1987], which is optimized for operations such as convolution, finite impulse response filtering and Fast Fourier Transforms. The EUCLID board subsamples the stored images and performs cepstral filtering on the windowed, subsampled images. EUCLID locates the peaks in the disparity image and reports the disparity to the sun host. Thus the vergence error is measured.

The cepstral estimator incorporates a number of optimizations suggested by our analysis and described in [Olson and Coombs, 1991]. The final implementation computes the cepstral disparity estimate for 32 $\times$ 32 windows in approximately 51 milliseconds, not including digitization time or the 8 ms required to acquire the VME bus and read the sample arrays from the frame buffer. Figure 3.4 shows a sample input and a plot of the cepstral output. It should be noted that no hysteresis is incorporated in the implementation described here: the tallest peak is always taken as the target disparity. Thus the disparity estimator reports the single disparity that best accounts for the shift between the images.

Although the implementation described above is adequate for some purposes, its accuracy is limited by the coarse quantization of the sample windows. For example, with the standard 16mm lenses each pixel in the subsampled cepstral output subtends about 27 arc minutes, or nearly half a degree of visual angle. The current implementation obtains sub-pixel resolution by first finding the peak pixel value in the cepstral output region and then interpolating to better localize the disparity peak. Only the scan line containing the peak value is considered, reducing the problem to 1D peak finding. The 1D sample set is modeled as a discrete approximation to a delta function (*i.e.*, a rectangle of width one pixel and unknown height) sampled by integration over adjacent regions of width one pixel. Thus the output of a given sample as a function of disparity should be the convolution of its rectangular sampling window with the rectangular disparity pulse, *i.e.*, a triangular pulse whose base is two pixels wide. The interpolation strategy suggested by this model is to return the centroid of the peak pixel and the larger of its two neighbors.

The assumptions underlying the interpolation strategy were tested by recording values of the output samples as a function of sub-pixel disparities generated by moving the cameras while viewing a static scene. Figure 3.5 shows the responses of three adjacent output samples to a simple scene. The responses have the predicted shape and slope, showing that the model is an accurate description of what happens with real scenes. However, the triangular pulses are broadened slightly at the base. In practice this means that the interpolation strategy described above will produce discontinuities

---

the amount of blurring is proportional to the resolution reduction, but due to a shortage of convolution boards, vergence either shares the pursuit blurring or omits blurring entirely when used together with the pursuit system.

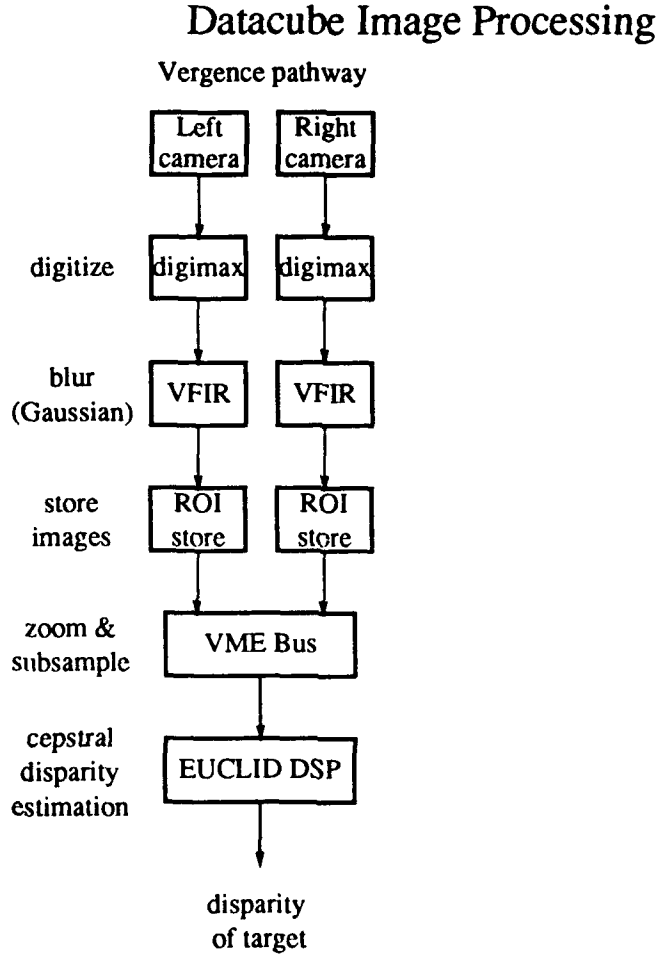
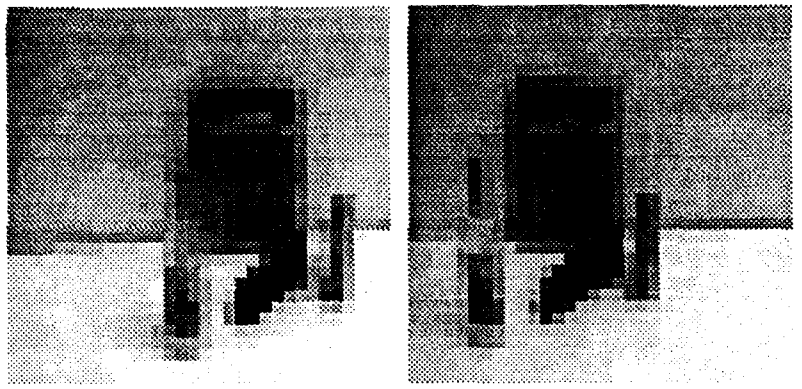
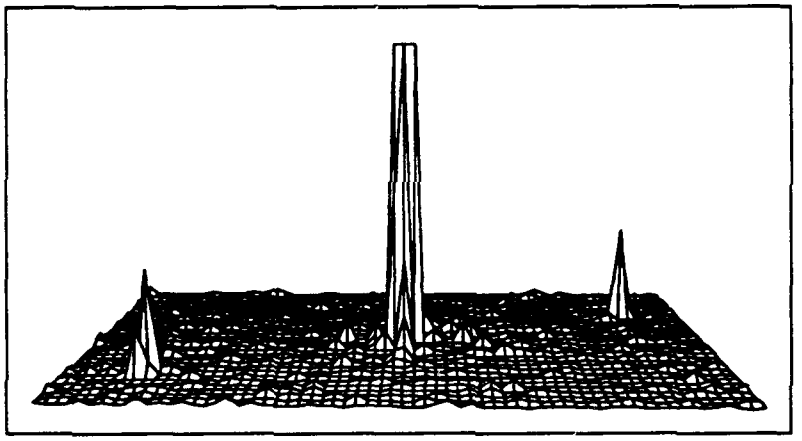


Figure 3.3: Vergence Image Processing: First, stereo images are digitized from synchronized cameras, and the images are blurred by convolution with a Gaussian kernel ($\sigma = 2.5$pixels). The vergence pathway begins with "zooming" and subsampling the images, and continues by using the cepstral filter to estimate the disparity of the "foveal" images.

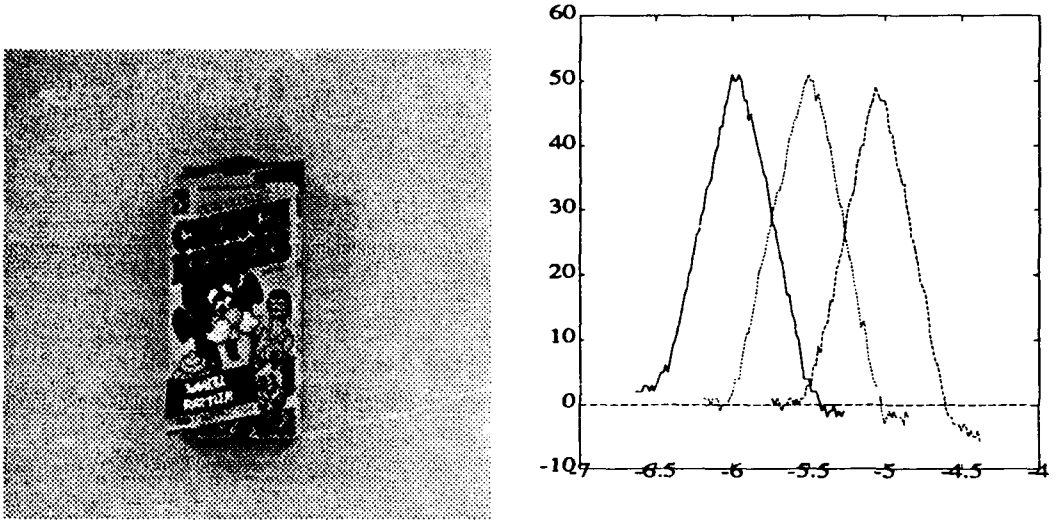


(a) (b)



(c)

Figure 3.4: Cepstral disparity estimator sample input and outputs. At top (a and b) are 32 × 32 subsampled images taken by the left and right cameras of the Rochester Robot. Below (c) is a surface plot of the power spectrum of the cepstral filter. The central peak, which is due to the autocorrelation of the joint image, has been truncated for display. The smaller peaks at left front and right rear give the disparity. Note the splitting of the foreground peak due to the presence of multiple disparities. The dominant disparity in this case is that corresponding to the textbook at the rear of the scene.



(a) (b)

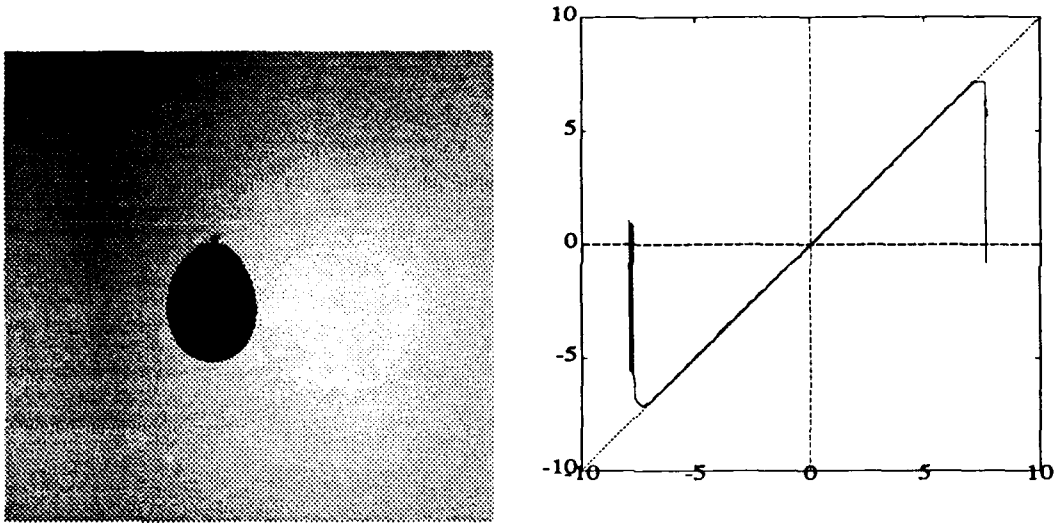
Figure 3.5: Cepstral output sampling responses. Graph (b) is a plot of pixel value versus vergence error in degrees for three adjacent pixels from the scene shown in image (a).

in the estimated disparity at the crossing point of the response curves for the left and right neighbors of the peak. In our implementation these discontinuities are avoided by incorporating a variable fraction of the smaller neighbor into the centroid. The fraction is equal to one minus the difference between the two neighbors divided by the difference between the smaller neighbor and the peak, or:

$$\text{frac} = 1 - \frac{\text{larger} - \text{smaller}}{\text{peak} - \text{smaller}}$$

3.5.3 Performance of the cepstral disparity estimator

The implementation of the cepstral estimator described above was tested in the laboratory on several scenes. For each test, the robot was directed to face the scene and the cameras were manually adjusted to approximately the correct vergence angle for the scene. Taking that angle as the home or zero-disparity position, the test program then swept the cameras over a range of vergence angles. At each position it recorded the actual disparity (represented by the difference between the commanded position and the home position) and the disparity reported by the cepstral estimator running on the EUCLID DSP computer.

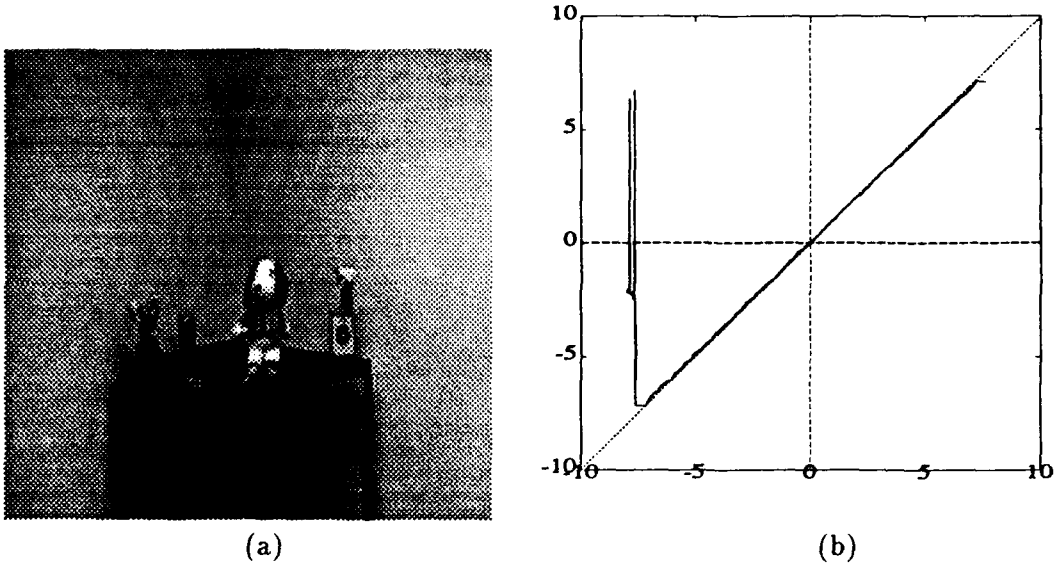


(a) (b)

Figure 3.6: Performance of the cepstral disparity estimator on the simple scene shown in image (a): (b) is a plot of measured versus actual disparity. Data were taken with 16mm lenses, giving a total field of view of 27 degrees. Sample windows used to compute the cepstral estimate were 14 degrees wide. The Dotted line shows the ideal response, *and the solid line plots the measured response.*

Since the home position was only approximately correct at the start of each run, most runs showed a systematic error of one or two pixels. In the plots below, these biases were removed by adding a constant that minimizes the RMS deviation from the ideal $x = y$ response. Figure 3.6 shows results of a test run on a nearly ideal scene consisting of a balloon against a contrasting background. The estimator fails badly at the extremes of its range, because at disparities exceeding ± 7 degrees the target object is no longer visible in both sample windows.[3] Within a ± 7 degree range, however, performance is good. The RMS error is 0.57 pixels, which (with the standard 16mm lenses) corresponds to 1.86 arc minutes. In other words, the estimate is accurate to a little more than half the width of an image pixel. *This is quite good, particularly in* view of the fact that the cepstral implementation subsamples by a factor of 8. Thus, relative to its sample window resolution, the cepstral RMS error is on the order of one sixteenth of a pixel.

[3]Errors of this type can be detected with high probability because they result in anomalous vertical disparities. The control software can use anomalous vertical disparities as a warning to disregard the measured horizontal disparity, and perhaps trigger a reaquisition process.



(a) (b)

Figure 3.7: Performance of the cepstral disparity estimator on a more complex scene: The Dotted line shows the ideal response, and the solid line plots the measured response.

Figure 3.7 shows results from a more typical laboratory scene. Although the plot looks similar to that in the previous figure, the RMS error for this test was 1.31 image pixels (4.44 arc minutes). The loss of accuracy is primarily due to a small error in the empirically determined constant multiplier used to convert error in pixels to error in degrees. Because the axes of rotation of the cameras do not pass through their nodal points, the nodal points undergo some translation when the cameras rotate. This means that the conversion constant has a small dependence on the depth of the target, or the vergence angle (and hence the stereo baseline). Compared to a best-fit straight line, this data set has an RMS error of 0.64 pixels (2.24 arc minutes), roughly comparable to the results in the ideal case. The systematic error could be removed by taking target depth (inferred from current camera position and approximate disparity) into account when converting from pixels to degrees. This has not been necessary to date, because small errors at large disparities have a negligible effect on the performance of the control loop because of the effect of feedback. High accuracy is important only at disparities near zero, where errors or discontinuities can cause the target angle to overshoot or oscillate around the desired value.

3.6 Vergence Control

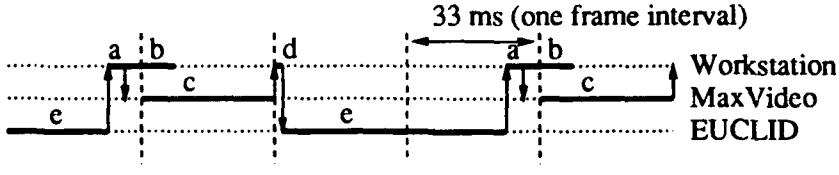
The goal of the vergence system is to generate smooth camera movements that correct the vergence error. The vergence control loop consists of three stages: digitization, error estimation, and error correction. Figure 3.8 illustrates the timing of the vergence loop. Digitization is done under control of the Sun™ host using the MaxVideo™ digitizers, convolvers and frame stores (one each per camera). It takes between one and two RS-170 frame times (33 to 67 milliseconds), depending on how much time remains in the current video frame when the command to acquire the next frame is issued. The Sun is free to do other things during digitization. Once the images are available in the frame store, the Sun signals EUCLID to extract the images from the frame buffers and estimate the disparity. This process takes approximately 59 milliseconds, after which EUCLID places the disparity estimate in a known location in shared memory and issues an interrupt to signal completion. The Sun converts the pixel disparity to angular coordinates by multiplying it by an empirically determined constant, and applies the control law to issue the appropriate velocity command to the camera motors. The Sun issues the motor commands *after* initiating the next digitization in order to allow digitization to proceed concurrently with motor control. This causes a slight delay in issuing the motor commands, but permits a substantially higher overall sampling rate. The loop consistently takes three frame times to complete. Thus, the vergence system achieves a servo rate of 10 Hz.[4]

3.6.1 The Controller

Early experiments with the vergence system used a proportional-derivative (PD) controller (*e.g.*, see [Dorf, 1980]) in cascade with the camera motor in a feedback loop, as shown in figure 3.9. (Although the target and actual vergence angle are continuous variables, since the entire system under our control is digital or presents digital interfaces we model the system discretely.) The summation node represents the effects of the desired vergence angle and actual vergence angle on the vergence error. The vergence error is encoded in the disparity of binocular images acquired from the cameras. The disparity is estimated by visual processing, which is modeled by the delay (z^{-1}) and E. The PD controller C generates oculomotor responses to reduce the estimated disparity. The controller gains were chosen empirically to obtain slightly underdamped response, resulting in a small overshoot in the step response. The system controls the velocities of the motors G, which are modeled as integrators, in order to achieve smooth responses to smoothly varying stimuli.

[4]Since disparity estimation takes 59 ms, the maximum theoretical servo rate is 15 Hz. Attempts to attain this rate have been thwarted by technical difficulties with capturing images and issuing motor commands concurrently with estimating disparity.

Vergence Timing Chart



a --- Set up frame buffers to capture images

b --- Issue motor commands

c --- Frame buffers capture images

d --- Fork cepstral disparity estimator on EUCLID

e --- EUCLID grabs subsampled images and estimates disparity

NB: times are approximate, for illustrative purposes.

Figure 3.8: Vergence Loop Timing Diagram.

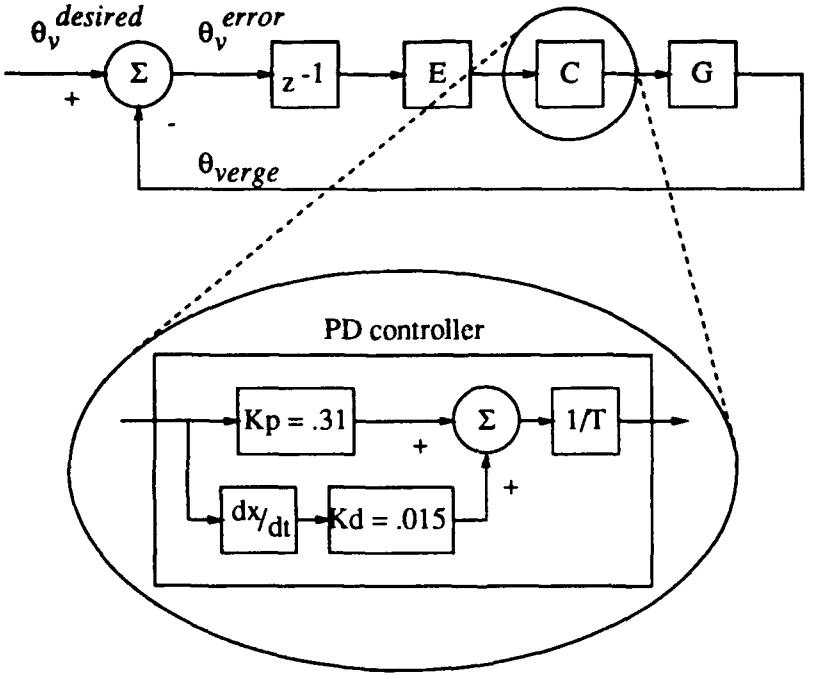


Figure 3.9: Block diagram model of the Rochester Robot vergence system. The error estimator E represents the cepstral disparity estimator. The PD controller modeled by C uses the backward difference to approximate dx/dt. The controller gains were empirically chosen to achieve slightly underdamped response. The controller generates velocity control signals for the camera motors G, which are modeled as integrators.

Since the system directly estimates only the vergence angle error, an estimate of its derivative, $\dot{\theta}_{verge}$, is approximated by the backward difference. This method obviously enhances any noise already in the estimate of vergence error. A linear predictor (*e.g.*, Kalman or α-β-γ filter) could be employed to smooth the estimates of θ_{verge}. (The α-β-γ filter is a version of the Kalman filter that makes the simplifying assumption that the system is time-invariant and the state estimation covariance converges to a steady-state value. In fact the kinematic models of constant velocity or acceleration fit the conditions for applicability of these simplified filters ([Bar-Shalom and Fortmann, 1988], page 89). In this case the state estimation covariance is reduced to a small set of parameters. Often they are chosen to be related in a way that expresses a known or assumed limit of the signal's behavior, such as its "maneuverability". See Sections 6.5.1 and 6.5.2 for more on α-β-γ filters). The α-β-γ filter assumes the target signal has constant acceleration, so this estimation would have to be done in a coordinate system in which the signal is relatively stable. The target's range (desired vergence angle) is more stable than the target's retinal disparity since the disparity is changed by vergence movements. To estimate the target's range, the target's disparity would be combined with the vergence angle, and the robot's vergence angle would have to be calibrated. This step has not been taken in the demonstration system.

3.6.2 Performance

The demonstration system's responses to step and sinusoidal stimuli were measured in the lab. Representative camera angle traces of step and sinusoidal responses are shown in Figures 3.10 and 3.11. Figure 3.12 summarizes the system's response to sinusoidal stimuli of frequencies up to 2 Hz. For ease of measurement, the system was not run in the normal mode of compensating for half the error with each camera, but rather one camera alone was moved to correct the entire error and the angle of this camera was recorded.

The step stimulus was produced by manually misconverging the "verging" camera prior to starting the system. The same effect could be achieved by misconverging the camera in the dark and then switching on the lights suddenly at time 0. In the response (Figure 3.10), observe the single time step (0.1 second) latency in detecting the disparity. As a consequence of this delay, the estimated disparity is seen to lag behind the camera's convergence angle, even though this disparity estimate provides the error signal that drives the vergence system. The small overshoot results from slight underdamping.

The effect of the proportional gain, K_p, is to drive the cameras at higher velocities when the error is larger. The derivative gain, K_d, helps accelerate the response when it is falling behind and decelerates the response when it is overtaking the stimulus, which can speed the response. However, a higher derivative gain produces oscillatory response. If K_p is increased, the overshoots become larger. If K_d is increased, oscillations appear in the steady-state response; if K_d/K_p is too large the system becomes unstable. The

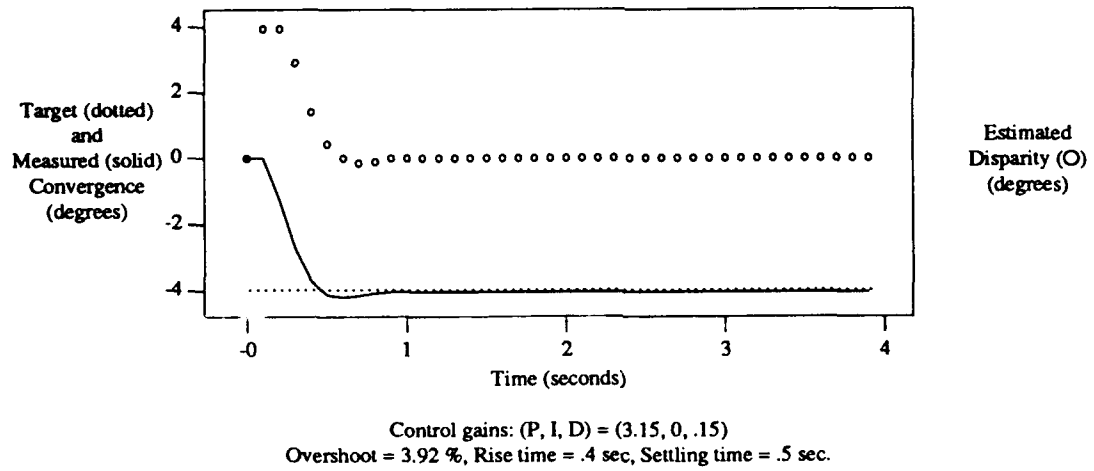


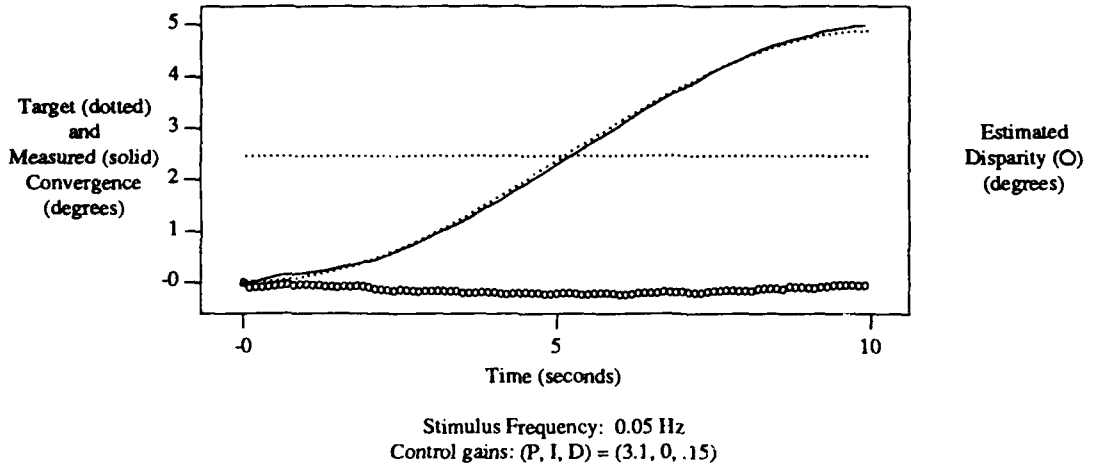
Figure 3.10: Response to a step in disparity: *Rise time* is the earliest time the response reaches 90% of its final (steady-state) value, and *settling time* is the earliest time the response stays within 5% of its final value. Note that the sample interval is 0.1 seconds.

time delay in the system contributes further to oscillations if responses are vigorous enough to overshoot before they can be detected (*e.g.*, due to high controller gains).

Analogous to the step stimulus, the sinusoidal stimuli were generated by rotating the non-verging camera sinusoidally. If the verging camera is held still, this generates a sinusoidal disparity signal. Thus, the target vergence angle was defined by the angle of the non-verging (stimulating) camera.

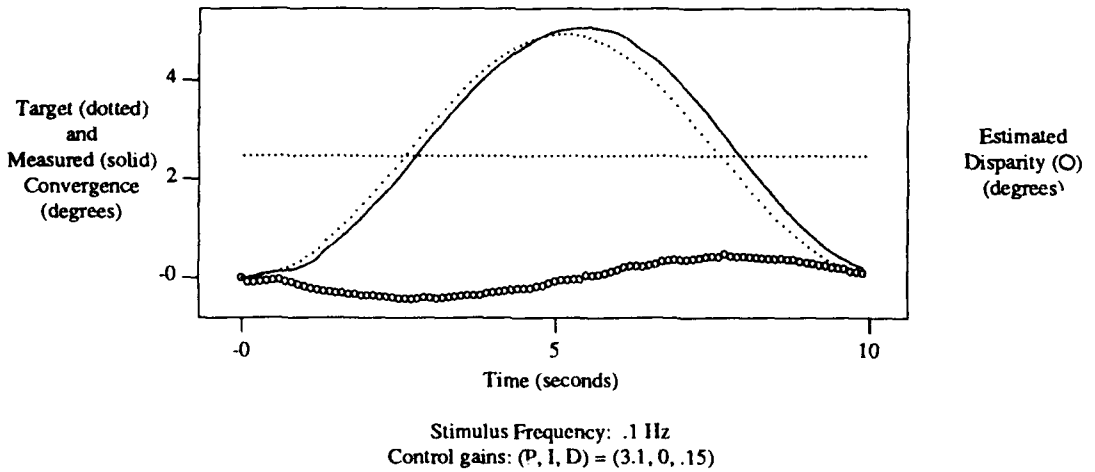
The effect of the time delay on phase shift can be seen by comparing the 0.05 Hz and 0.1 Hz responses in Figure 3.11: the same time delay contributes proportionately more to the phase lag at higher frequencies, since the time course of each cycle is shorter at higher frequencies.

The vergence responses to sinusoidal stimuli were measured for frequencies ranging from 0.05 to 2 Hz. The gain and the phase shift of the system's responses are summarized in the Bode plot of Figure 3.12. The system's behavior suggests that it may be a second order system. However, the constant time delay seems to produce a linear phase shift, as shown in Figure 3.13, since a constant time delay contributes proportionately more to phase shift at higher frequencies.



Stimulus Frequency: 0.05 Hz
Control gains: (P, I, D) = (3.1, 0, .15)

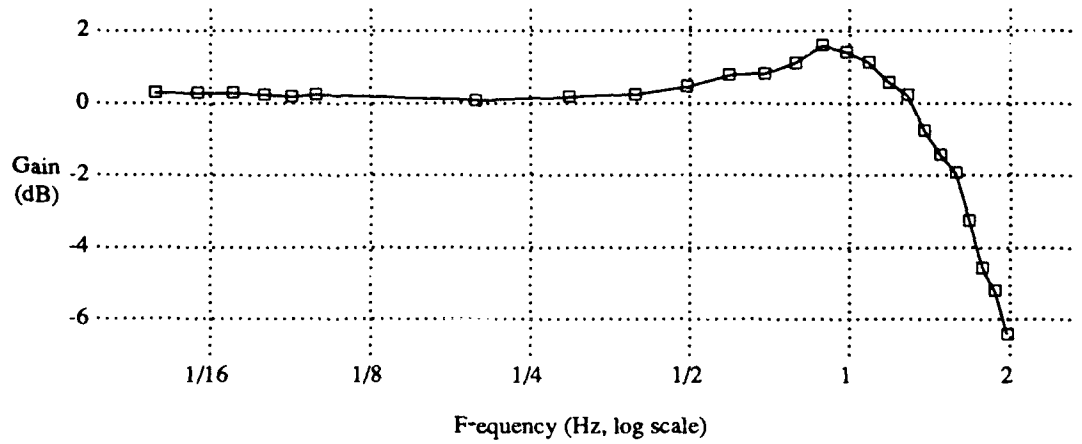
(a)



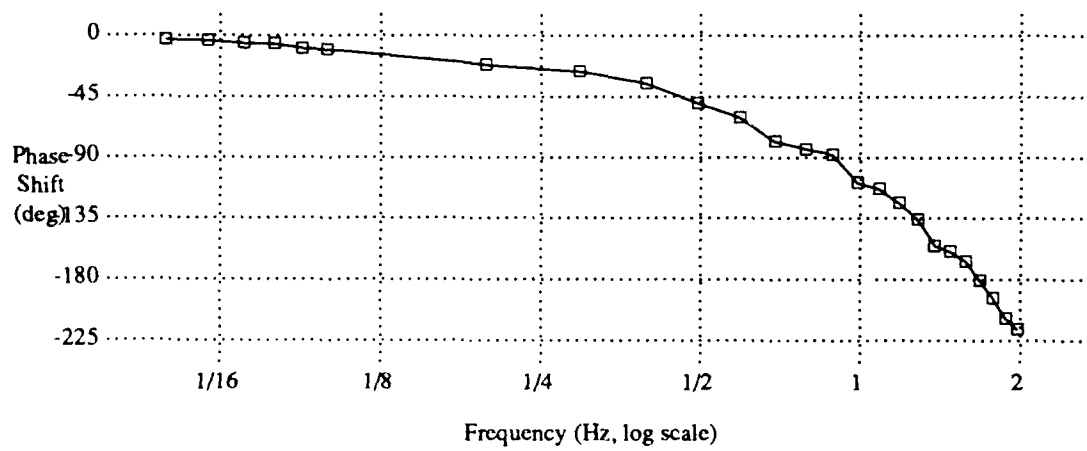
Stimulus Frequency: .1 Hz
Control gains: (P, I, D) = (3.1, 0, .15)

(b)

Figure 3.11: Response to sinusoidal disparity stimuli.



(a)



(b)

Figure 3.12: Bode plot. The gain (dB) and phase shift (degrees) of the vergence responses are shown for sinusoidal stimuli of frequencies ranging from 0.05 to 2 Hz. Gain (dB) $= 20 * \log_{10}(\frac{\text{response amplitude}}{\text{stimulus amplitude}})$ and the phase shift is the difference in the phase angle of the two signals.

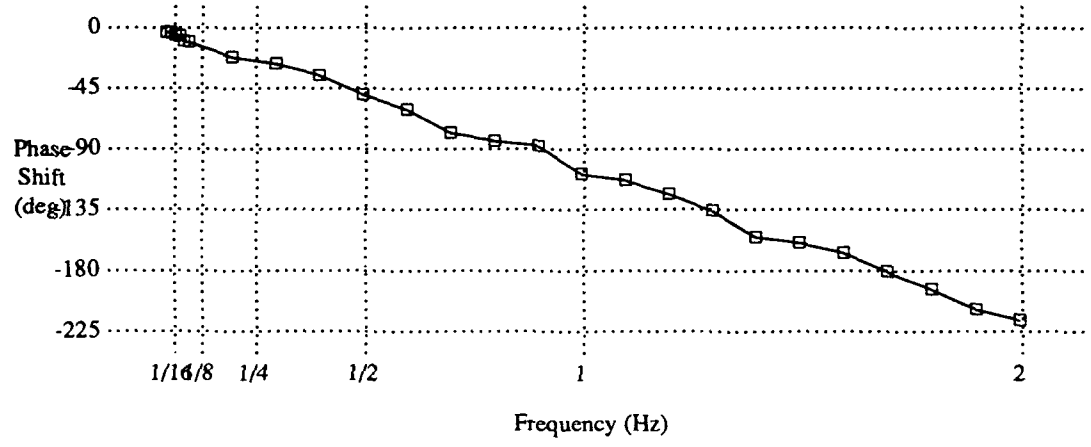


Figure 3.13: The phase shift (degrees) of the vergence responses is shown plotted against frequency on a linearly scaled abscissa, in contrast with the logarithmic abscissa of the Bode plot, to show that the phase shift seems to be linear. This character of the phase shift arises from a constant time delay, since it contributes proportionately more to phase shift at higher frequencies.

3.7 Summary

We have argued that vergence is important for active vision systems, and have discussed general issues in the design of vergence control systems. We have also described in detail the application of these ideas to develop a real-time vergence control system for the Rochester Robot.

The error estimator for the vergence system is a variant of the cepstral disparity estimator of Yeshurun and Schwartz [1989]. The estimator has been shown to be capable of remarkable accuracy, in the best case achieving an RMS error of a small fraction of a pixel. It is simple enough that with a small investment in special hardware it can be computed at speeds comparable to the video frame rate. The cepstral method of disparity estimation can be shown to be equivalent to autocorrelation of images that have been adaptively enhanced to sharpen their autocorrelation functions. It is thus closely related to phase correlation.

The demonstration system uses a position controller that generates smooth vergence camera movements in response to smooth changes in the desired vergence angle. This system also responds reasonably (but suboptimally) to step changes in target vergence angle. Optimal response to step stimuli could be achieved by saccadic vergence movements.

4 Pursuit

Primate visual systems offer an existence proof for gaze holding capabilities, and models of these systems can offer hints for the design of robot visual systems. However, most models for holding gaze do not address the visual processing necessary to implement this function. For instance, it is generally assumed that full-field optical flow is the visual signal that is used to stabilize gaze under egomotion, but optical flow is homogeneous only for rotation of the eye about its optical center. Similarly, "retinal slip of the visual target" is the visual signal commonly assumed to drive smooth pursuit eye movements that follow a moving object. In this case, it is being assumed that *the target's* retinal slip has been distinguished from the retinal slip of the rest of the scene. However, the optical flow signal is complex in general, and it is not clear how to parse the optical flow to determine the retinal slip of the target object. In both cases, what is needed is a mechanism that distinguishes the retinal slip of the visual target from that of the rest of the scene. This problem has received little attention in the biological literature. However, it has been suggested that disparity filtering mechanisms may play a role in the interpretation of optical flow by gaze stabilization systems [Howard and Simpson, 1989; Miles *et al.*, 1991], and disparity filtering of a similar sort is used by the gaze holding system that is described here.

The goal of *smooth pursuit* contrasts with computer vision's traditional *passive tracking* task. In passive tracking, the cameras move without regard to the goal of tracking the target object. For instance, the cameras on a mobile robot may point straight ahead like automobile headlights. The optical flow observed by the robot will result from the three dimensional structure of the scene and the robot's motion. Further, the target will move about in the cameras' images. In contrast, during active visual following, the cameras rotate to follow the target. Consequently, the target's retinal slip is minimal compared to the flow of the surrounding scene. In addition, the target's image is held near the center of the field of view.

In order to be as general as possible, the pursuit system should be able to follow a moving object without necessarily recognizing it first. A robot that must recognize an object to be able to follow it will only be able to use that facility in domains where every object is known to it and in which it has a means to recognize all objects. Such a robot's

applicability will clearly be limited. Therefore, the system must use *precategorical* visual cues (*i.e.*, prior to object recognition) in order to distinguish the visual target from distracting objects and the surrounding scene. Unfortunately, it is not clear how to extract this information from the visual signal.

Some visuomotor research has recently begun to appear in the computer vision literature, and various visual tracking and servoing systems have been reported. The systems perform saccadic tracking, smooth servoing on position or velocity, or combinations of both smooth pursuit and catch-up saccades. Similarly, approaches to visual processing include both optical flow processing and object identification and location. The next three sections discuss the issues and related work in visual processing and motor control for robot pursuit systems, and the remaining sections describe the approach taken on the Rochester robot head.

4.1 The Pursuit Problem

The goal of smooth pursuit behavior is to keep the visual target's image steady and centered in the camera's image. Thus the system must be able to determine the target's retinotopic position and retinal slip.

First, consider taking a hint from primate pursuit models that might help us build a robot pursuit system. Krauzlis and Lisberger [1989] present a recent model of the smooth pursuit system. However, like most such models, their model does not address the extraction of the retinal slip of the the target. Even the treatment of [Lisberger *et al.*, 1987], which is entitled "Visual Motion Processing and Sensory-Motor Integration for Smooth Pursuit Eye Movements", goes no farther than arguing that retinal slip and retinal image acceleration must be signals to which the pursuit responds, based on analysis of the behavior of the system. The means by which the retinal slip of the target is distinguished from other optical flows experienced by the visual system is so heavily influenced by cognitive factors that no studies of biological visual systems have clearly illuminated this question. Therefore we must rely on the computer vision literature.

4.1.1 Parsing Optical Flow

One approach is to try to parse the optical flow to find the target's retinal location or slip. This is easy for passive tracking if the robot is stationary and the target is the only object in motion. Simply hold the cameras still and detect the moving object. This approach will not work for gaze holding, however, since the camera movements make the entire scene appear to move, and it will not work even for passive tracking if there are several moving objects or the surrounding scene appears to move because the robot is moving. It is necessary to distinguish the target or its optical flow from the rest of

the scene and its flow. Consider that a robot moving in a 3D scene generates optical flow as a function of the distance of the objects as well as the robot's motion. How will the visual system isolate the target or its slip from the rest of the scene?

Optical flow parsing techniques in computer vision range from detection of moving objects to reconstruction of the scene. In a simple scene, Allen [1989] is able to use spatio-temporal motion energy to servo a camera on an object moving on a blank floor. In general, however, detecting moving objects against a more complicated background is computationally expensive [Heeger and Hager, 1988]. Nonetheless, Nelson [1991] has shown that object motions that result in flows that are inconsistent with the flow that arises from egomotion can be detected in real time. For instance, if the robot is rotating to the left, all optical flow due to the robot's egomotion is constrained to move to the right. Any optical flow inconsistent with this (*e.g.*, up and down) indicates a moving object. Thus it is possible to detect moving objects under some rather constrained camera movements, but the optical flow that arises during visual following of a moving object is quite complex, and it is not yet known how to parse the optical flow signal efficiently.

The optical flow patterns that arise from camera movement and egomotion in a three-dimensional scene can be quite complex. For instance, consider the simple case of a robot translating linearly along a smooth ground plane with the cameras pointed straight ahead. The optical flow bursts smoothly outward from the horizon in a radial pattern. Now if the observer simply fixates a point on the ground plane to the right of the direction of travel, the instantaneous camera rotation adds a uniform flow field to the optical flow due to the observer's translation. The optical flow that results from adding these two flow fields expands from the point of fixation on the ground plane and the flow vectors swirl outward in a counterclockwise pattern. A robot moving through a scene with less uniform structure would observe distortions of this optical flow due to the varying depths of objects in the scene. In general, the optical flow signal is complex, and it is not easy to distinguish the retinal slip of the target from that of the surrounding scene.

Both Burt *et al.* [1989] and Heeger and Simoncelli [1989] describe techniques to estimate egomotion by iteratively refining the egomotion estimate from image motion. Given the complexity of the flow signal, it is hardly surprising that these techniques are computationally expensive and do not converge quickly. On the other hand, Heeger and Jepson [1990; 1991] present a parallelizable, non-iterative method for computing three-dimensional motion and depth from optical flow signals for a rigid static scene. While this work represents an important step, the method has not yet been shown to extend robustly to correct interpretation of non-rigid scenes, and it depends on the ability to extract optical flow reliably.

Without explicitly estimating egomotion, it is possible to segment images based on independent coherent motion under roughly the same conditions that allow binocular

stereo segmentation (*i.e.*, that the change of the view of each rigid component of the scene can be approximated locally by an image shift). Girod and Kuo [1989] segment images on the basis of motion using the cepstral filter and achieve results that are comparable to the segmentations obtained by Yeshurun and Schwartz [1989]. For instance, the cepstral filtering is applied to small local image patches, and each image patch is classified by the predominant (stereo or motion) disparity in corresponding patches. Similar results are shown in [Shvaytser, 1988], by a method that might be considered "generalized difference of images". Interestingly, the operator is not only given with a frequency domain interpretation, but also it is derived in a probabilistic formulation. These approaches either require many fast local operations or they rely on fast implementations of Fourier transforms on many local image patches.

Woodfill and Zabih [1991] have achieved real-time motion and stereo segmentation of objects on a Connection machine. However, there is currently no real-time image capture mechanism available for such machines. Their method relies on the object, whose initial segmentation is given, not to change appearance much between frames. Image patches that belong to the target are correlated with nearby patches in stereo and motion disparity spaces to maintain the segmentation of the object. The object is not required to be rigid, but the appearance of the object must not change too drastically between sample frames. Using both stereo and motion provides redundancy and more robust segmentation, since each modality can sometimes fill in when the other lacks reliable information. Here again, the required correlation of many image patches is computationally expensive.

4.1.2 Matched Filters

Another approach to locating the target is to use matched filters. Several variants of this technique have been employed. Clark and Ferrier [1988] suggest using conjunctions of features in a saliency image to locate an object of known properties. Thus, it is assumed that choosing an appropriate set of primitive features and weighting function for describing their relevance (salience) to the task will suffice to make the target "pop out". However, the phenomenon of pop-out occurs in humans only for a limited number of features (such as color, and motion) and only a few conjunctions pop out [Triesman, 1985]. Similarly, computational attempts to dynamically conjoin arbitrary sets of features for the identification of relevant objects have met with limited success. In a somewhat different approach, Corke and Paul [1989] locate binary thresholded objects whose moments are known *a priori*. These features might be initialized by taking the conjunction of the features in an image window that is known to contain the target image. However, computing moments of thresholded grayscale images is likely to be brittle to changes in lighting, point of view, *etc.*. A more traditional approach is taken in Papanikolopoulos [1991], which tracks a set of correlation features on a rigid object,

but it is difficult to automatically select features that will provide robust correlation results [Thorpe, 1983].

Swain [1990] introduces a fast scheme for recognizing and locating multi-colored objects. However, since the representation is not view-invariant, several "characteristic views" of each object are used. This provides a relatively fast method for reliably locating an identified object, but it would not follow a novel object.

A general problem with matched filters is that this approach does not work if the object rotates. The difficulty is that a matched filter is view-dependent. The problem can be illustrated by considering a simple form of matched filtering. A simple strategy is to use the last subimage believed to contain the target image as a correlation template to locate the target in the next image. Unfortunately, this technique relies on a view-dependent model of the target (*i.e.*, its appearance from the last viewpoint of the observer) so it will fail if the appearance of the target image changes considerably (*e.g.*, due to rotation of the target object). This problem can be overcome by updating the correlation template with the new image of the target. However, without some other means of locking onto the target, this procedure is prone to drift off the target onto the background.

4.1.3 Fixation Segmentation

Considering that gaze is to be actively held on the target, it is possible to exploit expectations of the target's retinal location and disparity. Miles *et al.* [1991] suggest that both peripheral and foveal optical-flows contribute to gaze stabilization in monkeys. The foveal flow is attributed to gaze target, and the peripheral flows are from other depths. Howard and Simpson [1989] have found that the gain of optokinetic nystagmus (OKN) in humans is inversely proportional to binocular disparity. They argue that the activity of cells responsive to direction of motion and zero disparity selectively augments OKN, thus enabling humans to stabilize images in the plane of regard without interference from competing motion signals arising from other distances. These observations suggest mechanisms that take different approaches to extracting useful information from the heterogeneous flow field present under head translation. The differences in flow velocities are due to motion parallax caused by the presence of objects at multiple depths. The Howard and Simpson observations suggest a mechanism similar to zero disparity segmentation [Coombs, 1989] that ignores flow signals from depths other than the plane of gaze fixation.

4.2 Related Work

A few attempts have been made to pursue moving objects based on optical flow and most of them have used simple approaches to motion segmentation. *E.g.*, Lee and Wohn [1988]

use stop-and-look (which they call "static look and move (SLAM)") tracking in which the target is expected to be the only moving object. Allen [1989] uses spatio-temporal motion energy to servo a robot arm-mounted camera on a target moving in front of a blank background. Toelg [1991] simulates the traditional pursuit model [Young, 1971], with internal positive feedback and catch-up saccades. The motion-based target segmentation assumes the object is moving coherently (and therefore it must appear essentially flat). Jenkin [1991] uses stereomotion channels to estimate the target's motion. However, it is not clear how the trajectory detectors distinguish the motion of the target from the motion of the surrounding scene.

The other approaches assume the object is preselected and rely on view-dependent features and therefore suffer from the shortcomings of matched filters. Brown *et al.* [1988] describe a position servoing tracker that used intensity image patch correlation for locating the target. Corke and Paul [1989] smoothly follow the centroid of a blob that is translating in a fronto-parallel plane, by translating the camera in a fronto-parallel plane. The moments (which are known *a priori*) are computed on a binary image obtained by thresholding the grayscale input image. This approach is likely to be brittle to changes in lighting, point of view, etc. Papanikolopoulos [1991] smoothly tracks pre-selected features (image patches). It is not known how to select such features automatically, although this problem has been and will doubtlessly continue to be a subject of investigation [Matthies *et al.*, 1989; Thorpe, 1983]. Waxman *et al.* [1988] saccadically track a set of features whose spatial relationship is pre-selected, and the object is therefore identifiable.

Clark and Ferrier [1988] present a rare binocular tracking system that saccadically tracks the appropriate conjunction of features to locate an object whose properties are known *a priori*. Their system has demonstrated saccades and position-servoed vergence to binocularly fixate a target as it moves in three dimensions. The left and right images are processed independently, and the location of maximum "salience" (the desired combination of features determined to be relevant) in each image is taken to belong to the target object. However, neither is there a guarantee that the feature values will be invariant to point of view, nor are the maximum salience values necessarily unique. Consequently, the correspondence problem may not be trivial enough to yield to salience images.

4.3 Strategies for Pursuit Control

There are two natural and obvious measures of target following performance: position error and velocity mismatch. Restated, a pursuit system could attempt to center the target image, and at the opposite extreme, the system could try to stabilize the target's image on the retina (reducing slippage) by matching the target's velocity without regard to the retinal position of the target's image.

Note that the goals of image-centering and slip-minimizing can conflict since smooth camera movements can only improve one of these measures at the expense of the other. The target's image is repositioned by slipping it across the sensor array, and as the target accelerates, the image is stabilized by sacrificing the position slippage that has already occurred to prevent slip since the target would have to be slipped back to its desired position.

One approach to this problem is give precedence to one of these goals. Thus one simple pursuit system could attempt to keep the target image centered without regard to the image slip required. Another simple system could try to minimize target slip. Both of these behaviors can be elicited from monkeys [Lisberger, 1990]. However, it is generally believed that the primate pursuit behavior consists of a combination of both smooth servo control that matches velocity to minimize slip and catch-up saccades that recenter the target image when it deviates too far from the fovea. A more realistic model includes a small position-error response as well as the velocity-matching response in the smooth component of the system.

This is a clever solution to the dilemma of how to minimize both velocity and position error simultaneously. As previously noted smooth movements alone cannot achieve both goals, and certainly saccadic movements cannot reduce motion blur since they don't match velocity. The catch-up saccades perform the important function of correcting accumulated position error while introducing minimal target slip, and target slip is minimized by the smooth component that matches the target velocity with the camera velocity.

Unfortunately, implementing saccades in a binocular system requires careful attention, especially to the vergence angle. Some of the questions that arise are:

- If catch-up saccades are used for pan and tilt position errors, will they include vergence components?
- If continuous processes are holding gaze, will these processes (both sensing and control) be disrupted by saccades?
- Will the saccades occur exactly, or only approximately, simultaneously?
- Will vergence changes be accomplished in the course of the saccades?
- Will the saccades be of the same duration in both cameras?

These questions of how to coordinate saccades have led us to employ the simple strategy of using only smooth camera movements in the demonstration system.

One of the most challenging problems faced by visuomotor control systems is coping with the delays in the system. Delays can cause a system to be unstable. For instance, if a feedback system can respond more quickly than it can measure the error, it may

respond to old error signals and over-react. Consider driving a car on a slippery road. The driver turns the wheel, but the car does not immediately change its course, so the driver turns the wheel further and consequently over-steers the car. Since cameras are quickly maneuverable and visual processing is slow, visuomotor systems face a similar control problem due especially to the delays of visual processing. There are a couple of obvious ways to prevent overshooting responses. One approach is to model the delays present in the system and anticipate the response of the system with an internal model of the the visuomotor system. This method is apparently employed by the primate gaze control systems, and both the primate model and engineering solutions of this type are discussed in Chapter 6. Another, simpler approach prevents overshoots simply by reducing the responsiveness of the system enough to make it stable. A combination of these methods is used in the demonstration system.

4.4 Pursuit on the Rochester Robot

A demonstration pursuit system is implemented on the Rochester robot head. The system is diagrammed in Figure 4.1. The system has two components, vergence and pursuit, that require different visual processing. However, both components use foveally-processed visual signals. The vergence and pursuit systems perform complementary functions. The pursuit system keeps the foveas centered on the fixated object, and the vergence system keeps the cameras converged on the target. The vergence system uses an estimate of the binocular disparity between the foveal images to measure the vergence error. However, the disparity estimator provides only *what* disparity is present in the images; it does not inform the system *where* in the images the disparity arises. Conversely the Zero-Disparity Filter (ZDF) does not measure disparity, but rather locates portions of the images that have zero stereo disparity. In this way the vergence system minimizes disparity, but it cannot ensure that the target is foveated. Similarly, the pursuit system foveates on the target, but it requires that the cameras be properly verged on the object in order to locate it. The vergence and pursuit systems generate camera vergence and pan and tilt velocity commands, but the motors are not configured with those degrees of freedom mechanically. However, they are linearly related and the motor controller performs the conversion from camera coordinates to motor coordinates.

The use of binocular cues and control of the camera motions enables a simple signal processing and servo system to achieve gaze holding precategorically. *I.e.*, the system is able to hold gaze on the fixation target without the ability to recognize the object, and the target is distinguished simply because it is fixated by the robot's gaze.

As a consequence of gaze holding, the visual target is easier to pick out. Thus, it is easier to actively follow an object with moving cameras than to track its images in stereo images with static vergence and no control of camera movement [Coombs *et al.*, 1990]. For instance, during active following, motion blur de-emphasizes the background.

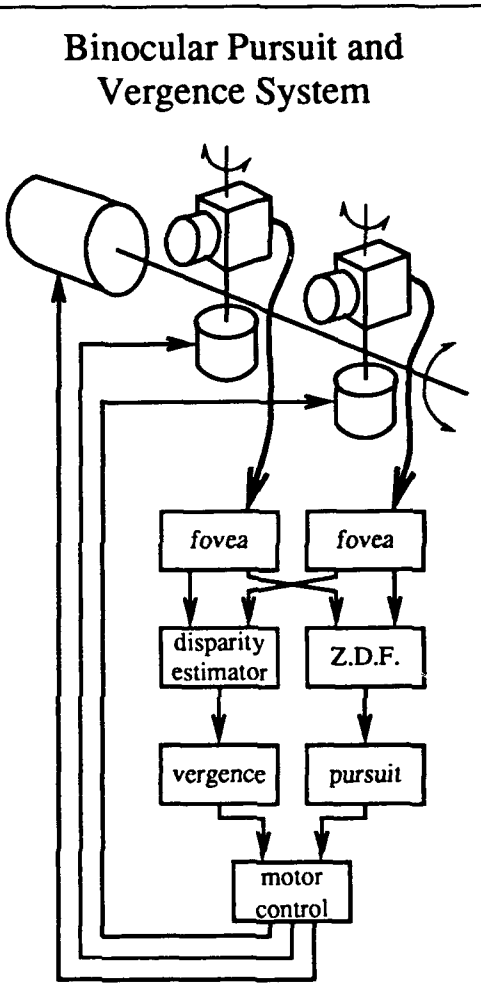


Figure 4.1: Binocular Pursuit system: The visual processing consists of vergence and pursuit branches. The visual signal is foveally processed before use in both subsystems. The vergence system estimates the disparity of the foveal images and controls the vergence angle to minimize the disparity. The pursuit system locates the retinal image of the object that is in both the fovea and the horopter, and the cameras are panned and tilted to keep the object located centrally in the fovea. The motor controller combines the pan, tilt, and vergence velocity commands and determines by simple linear relations the required left and right pan and tilt motor velocities.

Further, simple visual sensing techniques that are uniquely available during gaze holding can be used to segment the object being fixated, as illustrated in Figure 1.2. Foveal vision emphasizes the fixated object simply by spatially localized processing or increased resolution, and disparity filtering picks out features near the *horopter* (the surface in the scene whose disparity is zero). The demonstration system locates the target by foveally filtering the features found by the zero-disparity filter (ZDF), effectively producing the intersection of the fovea and ZDF. The target's retinotopic location provides the error signals the gaze control system needs to control the gaze and vergence angles.

Taking advantage of the vergence system, the pursuit system locates the target's retinal azimuth and elevation by the location of features with zero stereo disparity. The vergence system controls the vergence angle of the cameras to minimize the disparity of the foveated object. Thus the vergence system relies on the pursuit system to keep the target foveated. This symbiotic cooperation of the pursuit and vergence system enables precategorical visual processing to suffice to support gaze holding.

In the demonstration system, the cameras smoothly pursue the target by servoing on its position to minimize the retinal azimuth and elevation pointing errors of the cameras.

We have used both PID control and P control with simple target signal prediction using α-β-γ filters. Initial experiments employed PID control for simplicity and robustness, and later attempts were made to reduce the phase lag by the use of prediction to overcome the latency of visual processing. Both of these methods produce fairly smooth following behavior. PID control is a simple controller that offers some flexibility by responding to the integral and derivative of the error as well as the error signal itself. However, the α-β-γ filter is used to apply a simple linear model to the target position estimate in order to predict its future values (see Sections 6.5.1 and 6.5.2). If the signal can be successfully predicted, the effects of the latency due to visual processing can be mitigated by controlling the system with predicted signals and comparing the predicted error with the error observed once the visual signal is processed. The use of these control methods in the demonstration system is discussed in Section 4.6.1.

4.5 Visual Processing for Pursuit

The pursuit system is started up once the cameras have initially acquired fixation on the target with a saccade and matched its estimated velocity. Thus it is fair for the pursuit system to assume upon initiation that the target is roughly centered and the target velocity is approximately known. In addition, the pursuit system can assume that the target image's size is given, since the visual processing to acquire the target can be assumed to have coarsely segmented the target (*e.g.*, by motion).

If the target can be located in the image reliably, its retinal slip can be estimated. The converse is not necessarily true, since knowing the target's retinal slip does not

directly lead to the target's retinal location. However, it may be possible to find the target's location by segmenting the image based on areas of uniform flow if the target's slip is unique enough. For a target moving fast enough, this may be true since the camera rotation to follow the target will give rise to opposite optical flow of a stationary scene, while minimizing the target's flow.

However, the demonstration does not segment the target by its motion. Instead, the target is distinguished by its stereo disparity. Although the idea is similar, the zero-*stereo*-disparity can be implemented simply and robustly even when image flows are complex and difficult to parse (assuming, of course, that the vergence system is able to keep the cameras verged on the target).

4.5.1 Disparity Filtering to Locate Objects

Pursuit uses vergence to isolate the target by disparity filtering. Features that have no stereo disparity can be detected in real-time using a disparity filter. When the cameras converge on an object, it projects an image onto the "retina" (CCD array) of each camera. Figure 4.2 depicts a scene of three objects at different depths with the cameras verged on the intermediate object. Each of the objects projects an image on each retina. However, only the middle object projects to the same locations on both retinae. The region of space that contains objects that project onto the retinae with no stereo disparity is called the *horopter*, and a simple filter can detect objects that lie in the horopter.

Disparity filtering is used to isolate the target from the background. A disparity filter can detect features that have no stereo disparity more easily than interpreting the stereo disparity of the images. A real-time nonlinear filter implements zero-disparity filtering to isolate the objects in the horopter. Figure 4.3 shows an example of this sort of filtering. The pursuit system relies on the vergence system to keep the disparity of the target within the range of the disparity filter. The vergence system does this by keeping the horopter on the target object by changing the vergence angle of the cameras to follow the target. With the target in the horopter, the disparity filter provides the retinal location of the target [Coombs and Brown, 1991; Coombs *et al.*, 1990]. On the assumption that gaze will normally be directed toward objects of interest, it may be appropriate for binocular agents to ignore features at large disparities. That is, disparity may be used to filter objects that are not currently of interest out of the scene.[1]

The zero-disparity filter is a nonlinear filter that suppresses features that have non-zero stereo disparity. The features it uses are vertical edges, since they are identifiable features that can give useful information about horizontal disparity. (Clearly, horizontal edges provide no helpful information about horizontal disparity, since long horizontal

[1]For this purpose, Olson [Olson, 1991] proposes frequency-based disparity filtering to blur features with large disparity (Figure 4.4).

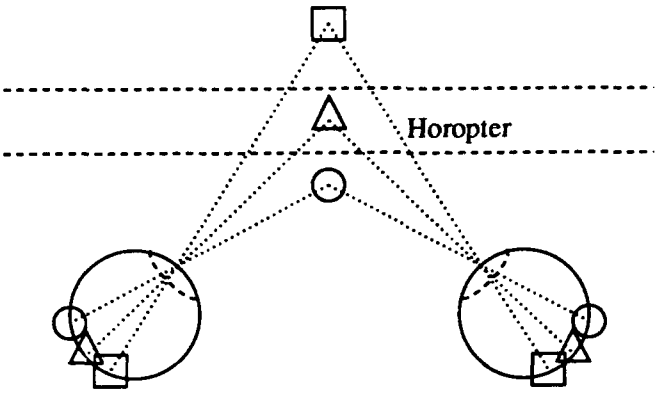
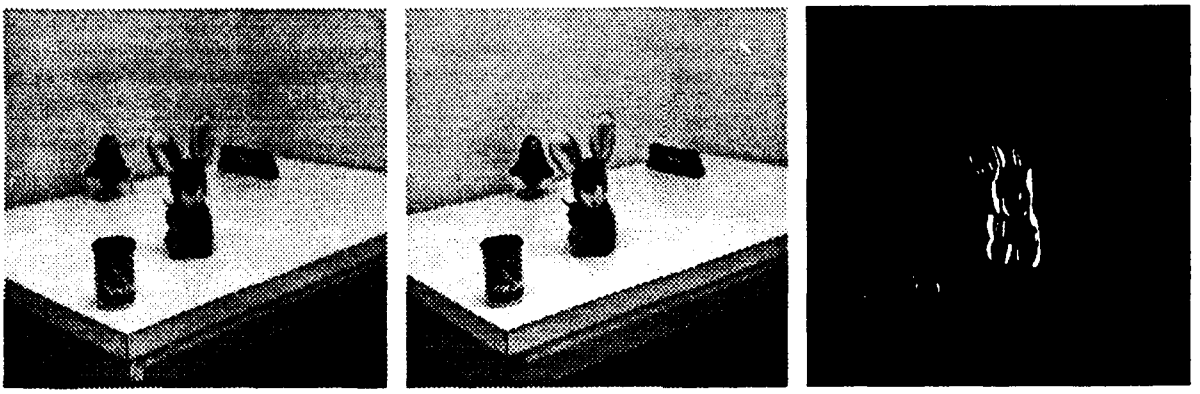


Figure 4.2: The *horopter* is operationally defined to be the region of space that contains objects that have no stereo disparity. It is a thin shell located at the fixation distance (the distance at which the cameras are verged). This figure illustrates the principle. The images of the triangle project to the same location in both retinae, whereas the images of the square, which lies beyond the horopter, have negative stereo disparity. Similarly, the circle, which is nearer than the horopter, results in stereo images with positive disparity.



(a) (b) (c)

Figure 4.3: Disparity Filtering of the scene shown in stereo images (a) and (b). Image (c) was produced by a real-time zero-disparity filter. The stereo images were first processed with vertical Sobel edge operators, and the stereo edge images were combined by a pixel-wise multiplicative 'AND' operator to produce the zero-disparity image. The effect of the filter is to suppress edges that have non-zero disparity, leaving an edge image that is dominated by objects in the horopter.

Figure 4.4: Frequency-based Disparity Filtering of the scene shown in stereo images (a) and (b) of Figure 4.3. The stereo frames were factored into laplacian pyramids [Burt and Adelson, 1983]. Then each pixel in the left image pyramid was rescaled by the thresholded normalized correlation between its local 3×3 neighborhood and the corresponding neighborhood in the right pyramid. Thus, the filter suppresses high frequency information associated with objects at large disparities. The reassembled image is shown.

edges can match over much of their length even with substantial disparity. Only their *ends* can be compared to find horizontal disparity.) The first step is to construct a vertical edge image of each image in the stereo pair. Then these images are compared in corresponding locations. If an edge is present in both images, then a feature appears in the resulting zero-disparity image. Of course the edges must be of like phase (*i.e.*, light to dark, or dark to light). Thus the filter detects features that have no stereo disparity.

The ZDF as it is described here is rather limited. However, Chapter 5 describes investigations into slightly more sophisticated zero-disparity filtering, which is mainly improved by richer measures of correlation between image patches.

4.5.2 Datacube Image Processing

The visual processing for the gaze holding system is implemented in real time almost entirely on our Datacube MaxVideo image processing system (Figure 4.5). The processing begins by digitizing a stereo pair of images from the robot head's cameras. The cameras are synchronized so the images are taken simultaneously.

The binocular gaze holding system includes the vergence system already described in addition to the pathway added to support pursuit. Like vergence, visual processing for pursuit begins with the images blurred by convolution with a Gaussian kernel ($\sigma =$ 2.5 pixels). This reduces the amount of aliased matches by removing some of the high

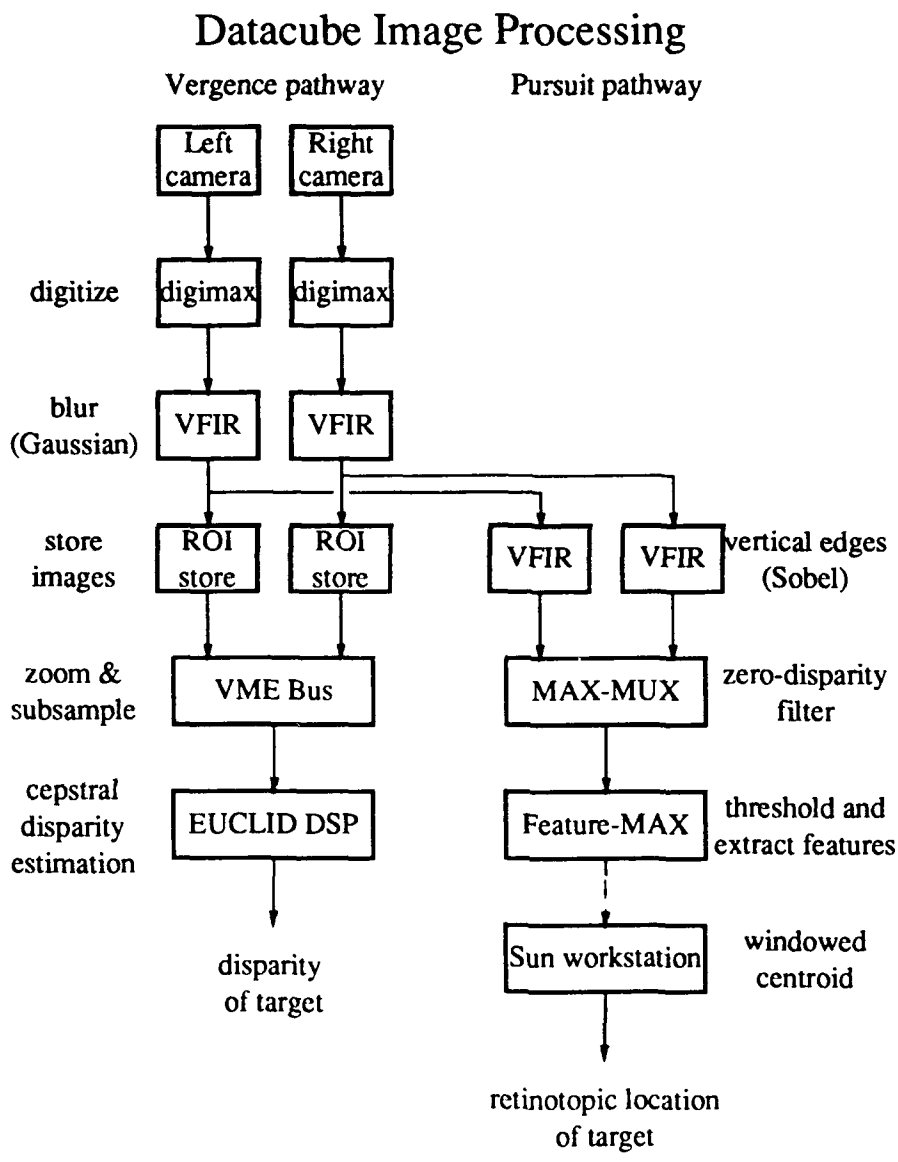


Figure 4.5: The gaze holding system does nearly all of the visual processing on our Datacube MaxVideo image processing system. There is a shared pathway in early visual processing, and two later branches for the vergence and pursuit systems. First, stereo images are digitized from synchronized cameras, and the images are blurred by convolution with a Gaussian kernel (σ = 2.5pixels). The vergence pathway begins with "zooming" and subsampling the images, and continues by using the cepstral filter to estimate the disparity of the "foveal" images. The pursuit pathway detects features (vertical edges) for disparity comparison, locates features with near-zero disparity and computes the centroid of these features that lie inside the "foveal" region.

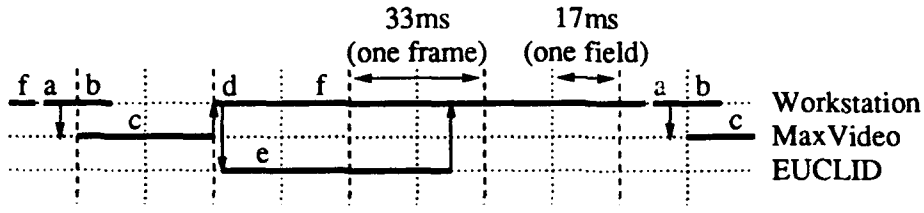
frequency features from the images. Edge operators are convolved with the blurred images to produce a stereo pair of vertical edge images. These images are then disparity-filtered. The disparity filter is implemented as a nonlinear function in a lookup table on a multiplexer board. The function compares the edge energy pixel-wise to determine whether there is a zero-disparity feature, which is indicated by matching edge features with no stereo disparity. The FeatureMax board records the locations of all the features in the zero-disparity image. The sun host then computes the centroid of the features in the central window of the image. This provides the pursuit system with the retinal location (and therefore the pursuit error) of the target.

4.6 Gaze Holding Control

The goal of the binocular pursuit system is to generate smooth camera movements that correct both gaze angle and vergence errors. The binocular pursuit control loop consists of three stages: digitization, error estimation (both gaze angle and vergence), and error correction (of both gaze and vergence angles). The vergence control system is augmented with the steps required to estimate and reduce the gaze pointing error. As with the vergence system, digitization is done under control of the Sun^{TM} host using the MaxVideo^{TM} digitizers, convolvers and frame stores (one each per camera). In addition, the zero-disparity image is thresholded by the Featuremax and the list of above-threshold features is stored in its memory. It takes between one and two RS-170 frame times (33 to 67 milliseconds), depending on how much time remains in the current video frame when the command to acquire the next frame is issued. The Sun is free to do other things during digitization. Once the images are available in the frame store, the Sun signals EUCLID to extract the images from the frame buffers and estimate the disparity. This process takes approximately 59 milliseconds, after which EUCLID places the disparity estimate in a known location in shared memory and issues an interrupt to signal completion. The Sun converts the pixel disparity to angular coordinates by multiplying it by an empirically determined constant. While EUCLID is estimating the disparity, the sun computes the centroid of the zero-disparity features. The length of time required depends on the number of zero-disparity features. Once the gaze and vergence errors are estimated, the sun applies the control law to issue the appropriate velocity commands to the camera motors. The Sun issues the motor commands *after* initiating the next digitization in order to allow digitization to proceed concurrently with motor control. This causes a slight delay in issuing the motor commands, but permits a higher overall sampling rate. The loop commonly takes 150 ms (four and one-half frame times) to complete. Thus, the combined vergence and pursuit system achieves a servo rate of approximately 7.5 to 8 Hz.[2]

[2] The lower servo rate (compared to the 10 Hz of the vergence system alone) results because not all the additional operations in the cycle can be performed in parallel. Notably, the calculation of the centroid of the zero-disparity signal adds a significant serial component to the cycle.

Pursuit and Vergence Timing Chart



a --- Set up frame buffers to capture images
Set up feature extractor to collect ZDF pixels

b --- Read motor angles
(Update gaze-vector display overlay)
Estimate retinotopic target location
Issue motor commands
(Update crosshair display overlays)
(Issue stimulus motor commands)

c --- Frame buffers capture images
Feature Extrator collects ZDF pixels

d --- Fork cepstral disparity estimator on EUCLID

e --- EUCLID grabs subsampled images and estimates disparity

f --- Collect ZDF pixels and calculate centroid

NB: times are approximate, for illustrative purposes.

Figure 4.6: Binocular Pursuit Loop Timing Diagram.

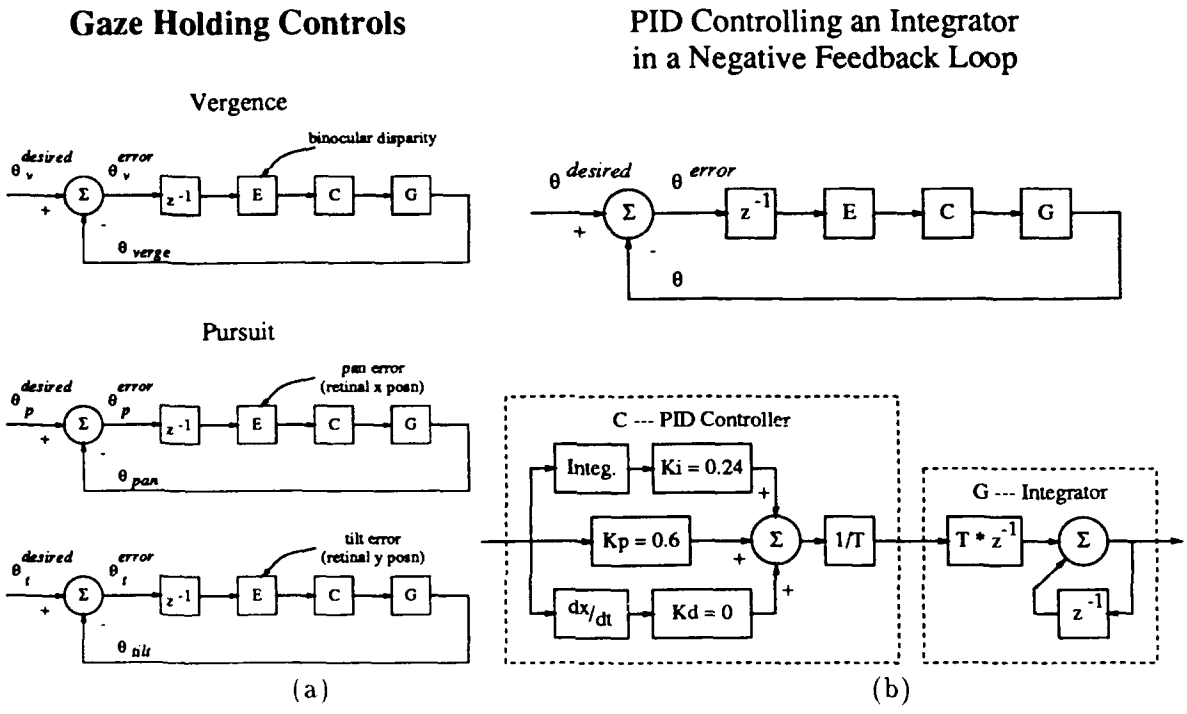


Figure 4.7: Gaze Holding Control Systems: The vergence and pursuit controls are driven by three independent feedback controllers, one each for pan, tilt, and vergence (a). The vergence system's error is found in the binocular disparity of the foveal images. The error signals for the pursuit controllers are the horizontal and vertical displacements of the target from the center of the fovea. Each of the controllers is a PI controller, and each degree of freedom of the gaze control system can be modeled approximately by the system shown in (b).

4.6.1 The Controller

The pursuit system uses a PID controller for each of pan and tilt of the robot's gaze. These controllers are the same as the controller used by the vergence system, as shown in Figure 4.7. The vergence system's error is found in the binocular disparity of the foveal images. The error signals for the pursuit controllers are the horizontal and vertical displacements of the target from the center of the fovea.

The gaze parameters, $\vec{\theta}$, map fairly directly, though not identically onto the mechanical degrees of freedom, $\vec{o}$, of the robot head. There is a single tilt motor, so

$$\theta_{tilt} = o_{tilt}.$$

The situation for pan and verge angles is sketched in "top-view" in Figure 4.8. The pan

and verge angles are related to the left and right pan camera angles by

$$\begin{aligned} \theta_{pan} &= \frac{1}{2}(\phi_{right} + \phi_{left}) \\ \theta_{verge} &= \phi_{right} - \phi_{left}. \end{aligned}$$

These equations relate the static angles. However, the motor controller must convert the pan and verge velocity commands to left and right pan motor velocities. Differentiating with respect to time, we obtain

$$\begin{aligned} 2\dot{\theta}_p &= \dot{\phi}_r + \dot{\phi}_l \\ \dot{\theta}_v &= \dot{\phi}_r - \dot{\phi}_l \\ \hline 2\dot{\theta}_p + \dot{\theta}_v &= 2\dot{\phi}_r. \end{aligned}$$

Thus, the motor velocities are

$$\begin{aligned} \dot{\phi}_r &= \dot{\theta}_p + \frac{1}{2}\dot{\theta}_v \\ \dot{\phi}_l &= \dot{\phi}_r - \dot{\theta}_v \\ &= \dot{\theta}_p - \frac{1}{2}\dot{\theta}_v. \end{aligned}$$

As one might expect, the pan velocity is transmitted to both camera pans, and the vergence is split evenly between them.

Each of these three gaze control systems operates independently, with no explicit cooperation. This simplifies the control laws. However, the gaze controls must be integrated to generate motor commands. These gaze controls do not interact deeply with one another given the Rochester head's motor configuration, but other controls could. For instance, a simple vestibular gaze-stabilizing system might interfere with the pursuit system if the robot were holding gaze on an object that was moving. Consider the extreme case of holding gaze on a target that moves in tandem with the robot. In this situation, the cameras must be held still in the head to hold gaze on the target. However, a simple vestibular gaze-stabilizing system that attempts to hold gaze on a stationary point in the environment must generate camera movements to do so, and these movements would take the cameras away from the target. The conflicting goals of the gaze-holding and gaze-stabilizing system must be reconciled because they share the common resources of the camera motors. Interacting gaze controls should be explicitly coordinated to achieve optimum performance [Brown, 1990c; Coombs and Brown, 1991]. Consider our simple example: if the vestibular gaze stabilizing system attempts to hold gaze on a target whose location is specified by a model and the model allows for object motion, then the low-latency vestibular system can assist a visually-driven gaze holding system.

One of the problems faced even by single degree-of-freedom visuomotor control systems is that the error signal is affected by every control response since the sensor is

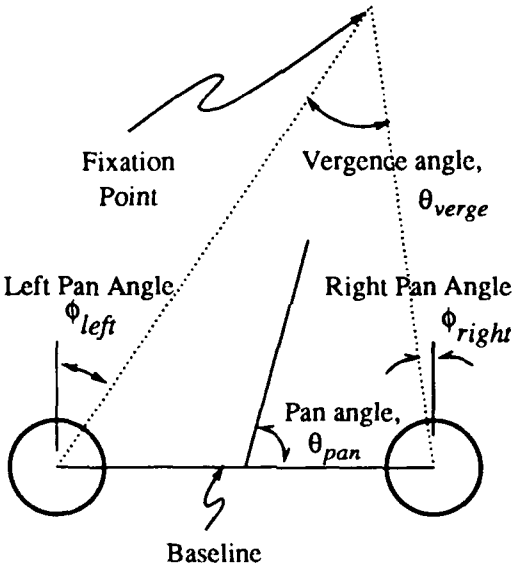


Figure 4.8: The relation between gaze angles $\vec{\theta}$ and motor angles $\vec{\phi}$ is sketched in this view of the "gaze plane" defined by the camera and the fixation point.

being moved. This can result in disruption of the visual signal. Similarly, in a dynamic scene, object motion may also perturb the signal. For example, as the view of the fixated object changes, the zero-disparity signal shifts and evolves, and sometimes it even drops out completely for brief periods of time.

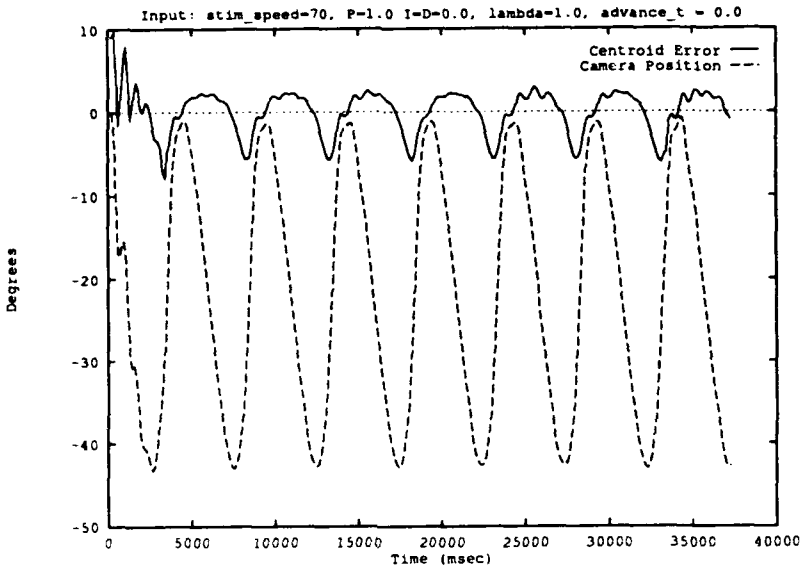
A simple remedy for signal dropout is something like visual persistence, and this can help keep the cameras moving when the signal disappears temporarily. However, this mechanism is based on a very simple model of target behavior, and it is likely to introduce sharp changes in the error signal. Another simple and common model of target behavior is the α-β-γ filter, and this filter smoothes and interpolates the target position signal (see Sections 6.5.1 and 6.5.2). The α-β-γ filter assumes the target signal has constant acceleration, so this estimation must be done in a coordinate system in which the signal is relatively stable. For gaze-holding target coordinates, this means the head-centered reference frame is much better for estimating the target position than any visual coordinate system. The visual coordinate system is based on retinal image coordinates, and the retina is constantly being moved by the gaze control system. In order to estimate the head-centric target position, the retinotopic location of the target must be combined with the camera's angle within the head.

Using the α-β-γ predictor in the loop to predict the delayed signal can lead to more accurate tracking. Figure 4.9(a) shows the camera movement and tracking error as the camera tracks the image of a dark object in approximate harmonic motion with a period of (360/70) seconds. The object is rotating in a plane and thus its distance from the camera varies and its velocity is not purely sinusoidal. The error is measured as the off-axis angle the centroid of the object's image. There is approximately a 100 ms delay in the system. The small phase difference between the sinusoidal waveforms of the target and camera motion induces a surprisingly large error. In Figure 4.9(a) an α-β-γ filter is used ($\lambda = 1$) with no predictive advance, so the tracking signal is smoothed somewhat. In Figure 4.9(b) the filter extrapolates the signal 50 ms into the future. The result is livelier tracking (in fact more advance destabilizes tracking) and reduced errors.

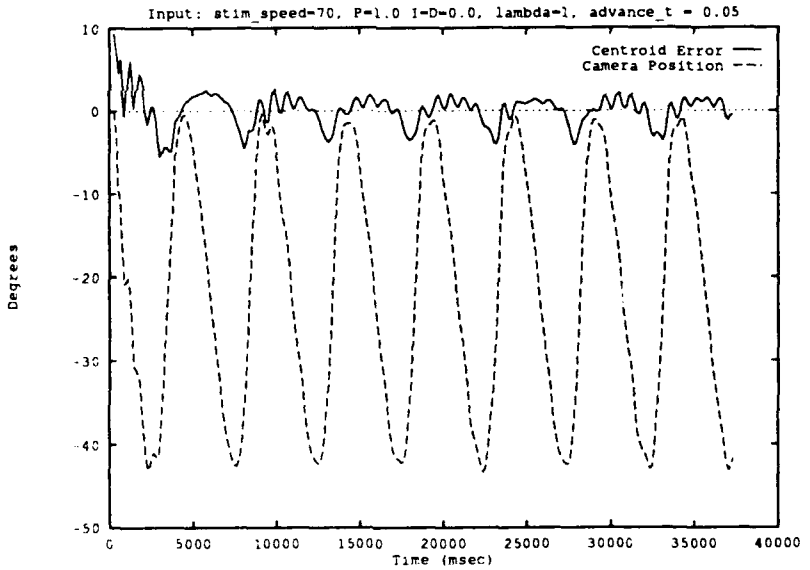
4.6.2 Gaze Holding Performance

It should be no surprise that the performance of the pursuit system is analogous to vergence performance since they use similar controllers. Note that vergence continues to function, but performance is sometimes perturbed by the changes in the scene as the target moves and the surrounding scene changes due to object motion, camera movements, and egomotion.

Figure 4.10 shows a stereo robot's-eye view of a typical stimulus setup and the measured camera pan, tilt and convergence angles and visual error signals for a target object moving through a field of distractors. The bunny was fixed to the end of the rotating stick and thus inscribed a circle, rotating while moving laterally and in distance. These measurements were recorded from a run with the stimulus rotating at 0.1 Hz, and



(a)



(b)

Figure 4.9: Camera motion and error when tracking an object in approximate harmonic motion. (a) Delay of approximately 0.1s induces small phase lag but large tracking errors. (b) Camera motion and error using α-β-γ predictor to advance the signal and overcome a delay of approximately 0.1s. Here the advance in the filter is 0.050s.

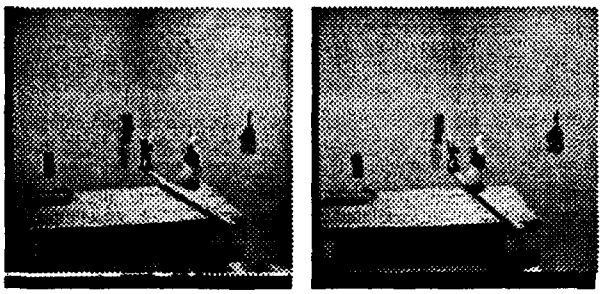
the pan angle trace reveals rotational camera velocities as high as $13\,\mathrm{deg}/s$, with the cameras lagging a bit behind the apparent target velocity, as indicated by the non-zero observed retinal error.

4.6.3 Ablation Experiments

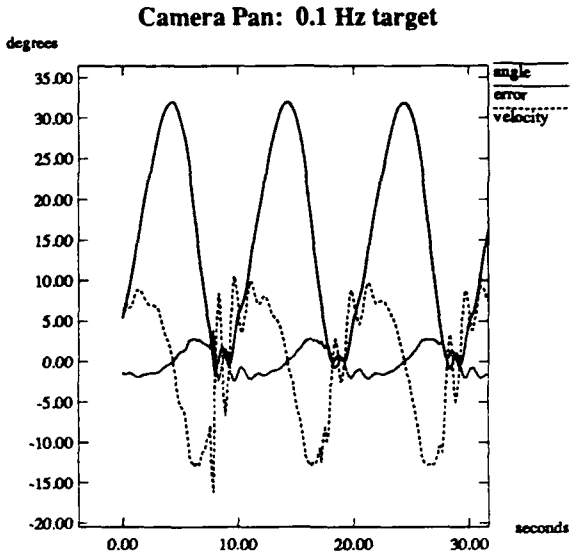
In order to illustrate the function of each component of the gaze holding system, selected components were removed from the system, and the resulting behaviors are compared with the behavior of the complete system. Traces of camera pan angle can be seen in Figure 4.11. The first trace show the uninjured system's pan angle in the typical bunny-following scenario. The second trace illustrates the effect of loss of foveal processing in vergence and pursuit. The vergence system verges the cameras on any object that captures the disparity estimator's attention, resulting in vergence bouncing around the scene as the pan angles and bunny position change. Similarly, the pursuit system attempts to center gaze on all objects that lie in the horopter. The third trace shows the behavior that results from fixing the vergence angle. Early in this run (not shown), the system wandered until it locked onto an object that shared the horopter with the target's initial location. The fourth trace shows the result of eliminating the ZDF and using instead the edge energy of one image to drive pursuit. (The large step inputs that resulted make the system unstable, so foveal reduction was also eliminated for this experiment. Including extra-foveal edges in the "target" centroid calculation dilutes the effect of features entering and leaving the "foveal" area that caused the instability.) Clearly the centroid of the edge energy was influenced by the target's motion, but it was also significantly anchored by the surrounding stationary scene. Obviously, each piece of the system contributes to the performance, and it is the combination of the simple components that allows each part to be simple.

4.7 Summary

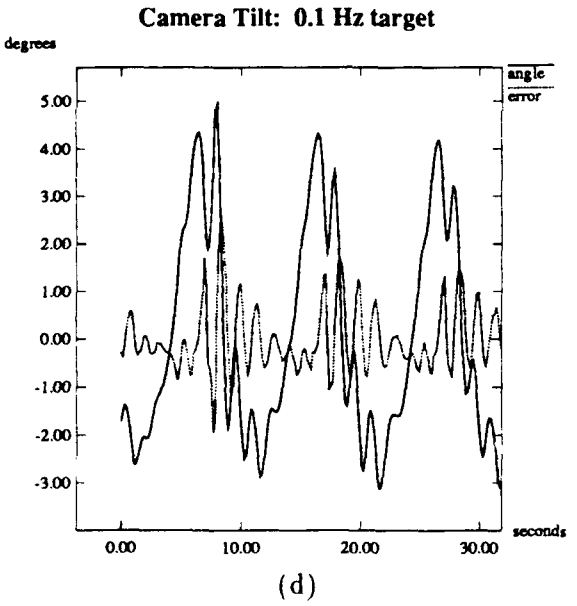
We have discussed some issues involved in holding gaze precategorically on an object that moves in three dimension with respect to the observer. In addition, we have presented a simple demonstration system that holds gaze using only precategorical visual cues. This is achieved by exploiting binocular cues to segment the target based only on its status as the fixation target together with the deliberate control of the camera movements to maintain fixation on the target.



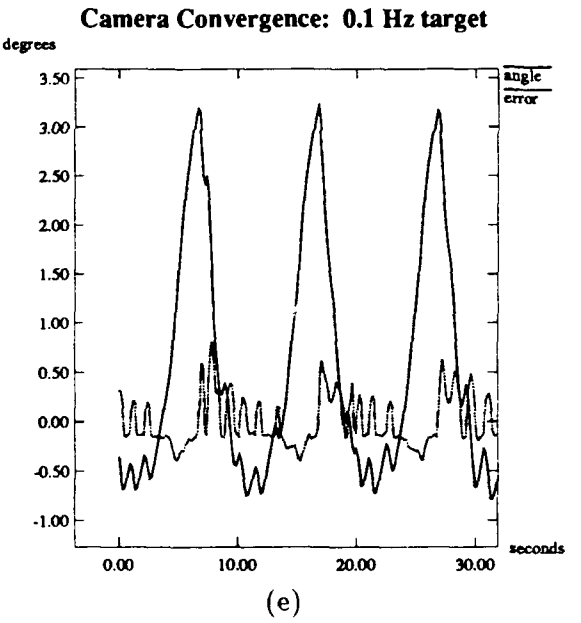
(a) (b)



(c)



(d)



(e)

Figure 4.10: Gaze holding camera traces. Measured traces of the pan (c), tilt (d) and vergence changes (e) show the performance of the gaze holding system in following a target moving in 3-D through a field of distractors. A robot's-eye stereo view of a typical stimulus setup is shown in (a,b).

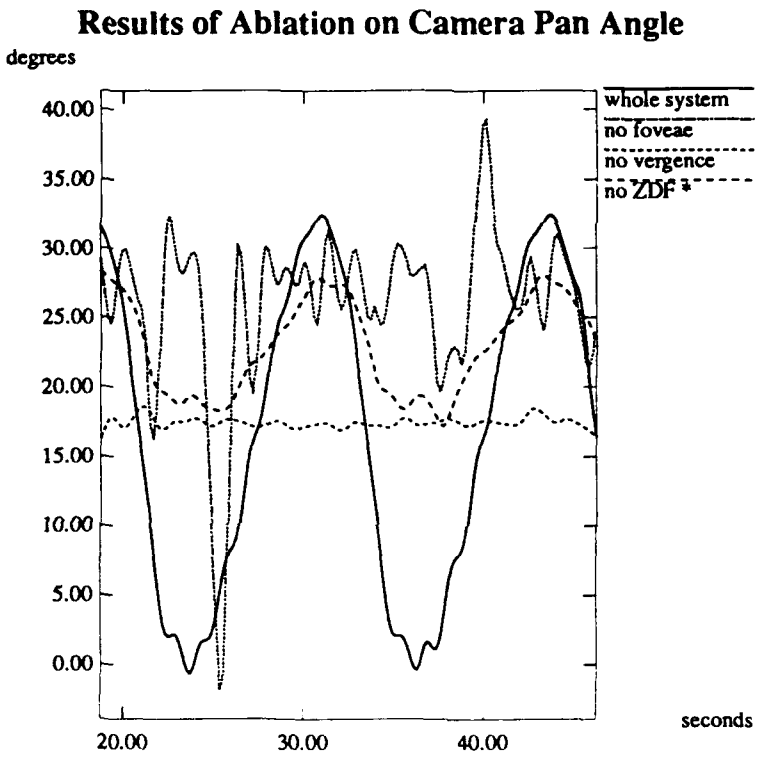


Figure 4.11: Ablation Camera Traces: The first trace shows the behavior of the unimpaired system for comparison. The second trace illustrates the loss of focus on the target object that results from the removal of foveal processing (or peripheral suppression). The third trace demonstrates the system's inability to follow the target if the vergence angle is cemented. The final trace shows how the system is distracted by objects at all distances when zero-disparity filtering is eliminated.

5 Zero Disparity Filtering

The zero-disparity filter (ZDF) is, ideally, a way to restrict the input from a binocular vision system to the *horopter*. The horopter is the surface in three dimensional space defined by the points that stimulate exactly corresponding points (*i.e.*, that have zero stereo disparity) in the two cameras (or eyes). Changing camera vergence sweeps the horopter in or out through space. Implementing the zero-disparity filter in practice involves finding areas of the images that have no binocular disparity and permitting these regions of the image to pass through the filter. The former step is basically like the stereo-matching problem. The early Datacube implementation (mentioned in Chapter 4) detects vertical edges in the stereo images, and only edges that appear in corresponding locations in both cameras are passed through the filter. It was thought that a patch-matching scheme in which two-dimensional image patches were matched could provide a more reliable match that would less vulnerable to accidental edge alignment. Such a matcher has been implemented, along with "feature goodness" operators to indicate the likelihood that a good match score actually represents a reliable indication of zero disparity. Patch matching clearly performs better in random dot stereograms, but we found the original edge-based matcher to work surprisingly well on a wide range of real and test scenes. This chapter is based on work reported in [von Kaenel *et al.*, 1991].

5.1 ZDF and Horopters

Consider two cameras (or eyes) horizontally displaced and sharing a common tilt angle. Any setting of their two pan angles induces a point of fixation in 3-D space where the camera axes intersect in the tilt plane; a horopter surface in 3-D is also defined.

Several definitions for the horopter have been proposed, and we adopt one that is particularly easy to interpret in this case. Paraphrasing from [Reading, 1983], page 88, the horopter is the surface in physical space, any point of which produces images in the two eyes [or cameras] that stimulate exactly corresponding points.[1] It is specifically and

[1] Note that this definition is particularly easy to apply to a robot's cameras since the images are

uniquely identified with the point of intersection of the two primary lines of sight and is the boundary between crossed and uncrossed disparity. That is, the binocular disparity for a point on the horopter is zero. This is illustrated by Figure 4.2.

The 3-D shape of the horopter is at least rather complicated (and at worst may not be well-defined), since it involves vertical, as well as horizontal, disparities. However, the horopter's 2-D shape in the tilt plane has received considerable attention (see Figure 5.1). The *geometrical horopter* is the circle (also known as the Vieth-Müller circle) passing through the two nodal points of the cameras and the fixation point. This is the shape of the horopter that would result if the corresponding point in each retina represents an equal angle with respect to the fixation point. The *human empirical horopter* is measured by various techniques with varying results. The empirical horopter is consistently found to deviate from the geometrical horopter in approximately the way that is depicted in the figure. The reasons for the difference include the spacing of "corresponding" points in the retina and the visual cortex, optical distortions, and distortions of the shape of the eye when it is rotated to different eccentricities. (See [Reading, 1983] for a survey.)

We are able to define the *robot's empirical horopter* directly since we have direct access to the images that are formed by the robot's cameras. Therefore, an ideal zero-disparity filter would detect points on objects that lie on the horopter. We simply define the robot's empirical horopter as the set of points that stimulate corresponding points in the robot's cameras, as detected by the ZDF. In practice the ZDF accepts small disparities near the zero-disparity boundary and this results in the thickening of the robot's horopter. The shape of the robot's horopter has a curved form similar to the other horopters, but we have not attempted to determine the relationship between the shapes. Naturally, the robot's horopter is influenced by the same sorts of distortions that affect the human empirical horopter. The planar (rather than spherical) retinal projection may introduce some additional distortion when the cameras point in an eccentric angle; in principle, this can be corrected by modeling the projective geometry so that "corresponding points" is defined in terms of equal angles from the fixation point.

Let us consider the application of the robot's horopter to visual localization. If it were possible to say for certain that left and right image points at the same (i, j) image coordinates really corresponded to the same point in space, then it would follow that the point in space lay on the horopte This would be a useful constraint in localizing the spatial point. The idea of the zero-disparity filter is to detect such zero-disparity points in two images for use by later stages of image analysis such as stereo depth calculations or segmentation.

In practice we assume the zero-disparity filter simply has as input two images. Its job is to produce an image-like array of points that indicate the filter's opinion that the

accessible, whereas considerable effort is necessary to interpret it for humans.

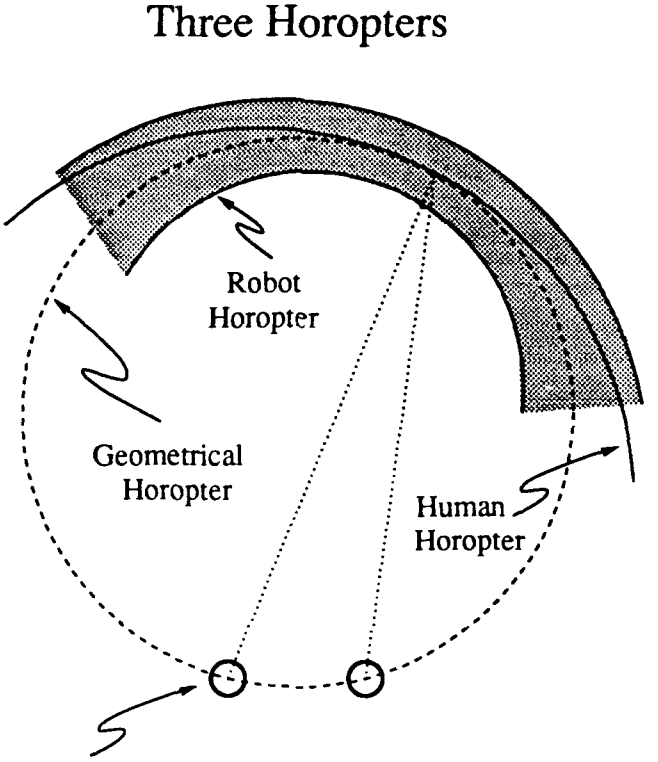


Figure 5.1:
Horopters: The Vieth-Müller circle, also called the geometrical horopter, is the circle that passes through the optical nodal points of the cameras and the fixation point. The human horopter depicts the empirical horopter, which is determined experimentally. Several factors have been identified that can influence this deviation. In contrast to the human horopter, the robot's horopter is defined by the points that stimulate the ZDF, closely following our adopted definition of horopter. This depiction is intended to be qualitative only, and it should be emphasized that rigorous efforts have not been made to determine the shape of the robot's empirical horopter.

corresponding points in the input images indeed are zero-disparity points. The problem is that the filter only has access to the two images, and that for sufficiently perverse images the correct answer may be arbitrarily difficult to determine. For two infinite perfect "picket fence" images (say, equally spaced white vertical bars against a black background) the problem is impossible due to the identical aliasing of the signal.

Thus the zero-disparity filter must do the best it can within its resource budget. This usually implies the restriction of local and low-level matching. Our approach is simply to use local matching as the criterion for zero-disparity image points. By increasing the size of the area matched and the sophistication of the match, we can eliminate increasing amounts of accidental aliasing matches that occur in real images.

5.2 The Initial Solution

The initial zero-disparity filter (ZDF) was designed and implemented to run at frame rates on MaxVideo^{TM} [Datacube, 1987] image processing hardware [Coombs, 1989; Coombs and Brown, 1991]. The ZDF is a simple non-linear image filter that suppresses features that have non-zero stereo disparity. The features it uses are vertical edges, since they are easily detected by convolution, and they can give useful information about horizontal disparity. (Clearly horizontal edges provide no helpful information about horizontal disparity, since long horizontal edges can match over much of their length even with substantial disparity. Only their *ends* can be compared to find horizontal disparity.) The first step is to construct a vertical edge image of each image in the stereo pair. Then the stereo edge images are compared in corresponding locations. If a similar edge is present in both images, then a feature appears in the resulting zero-disparity image. Of course the edges must be of like phase (*i.e.*, light to dark, or dark to light) and contrast. Thus the filter detects features that have no stereo disparity.

The ZDF applies a 5×5 up to a 9×9 Sobel vertical edge operator to each stereo intensity image (I_L and I_R) as a feature detector. The edge operator size chosen depends on what the user wants. Figure 5.2 shows the 7×7 Sobel edge operator used by the ZDF; other sizes can be derived from it. The two (signed) edge images (E_L and E_R) are then compared pixelwise: if an edge of like phase and similar contrast is present in corresponding locations in each image, then an edge of similar strength will appear in that location in the resulting zero-disparity image. In Fig. 5.3, the bunny in the center has zero disparity and the surrounding objects have non-zero disparity. Fig. 5.4 is the zero-disparity image that results from applying the ZDF to this scene. You will notice that Fig. 5.4 is comprised of twenty images. The edge operator size varies from 5×5 in the top row to 9×9 in the bottom row, and the reduction in resolution varies from 2-fold (left column) to 16-fold (right column). The figures that show the performance of the other implementations of the ZDF are of similar format.

0	0	-1	0	1	0	0
0	-1	-2	0	2	1	0
-1	-2	-4	0	4	2	1
-2	-4	-8	0	8	4	2
-1	-2	-4	0	4	2	1
0	-1	-2	0	2	1	0
0	0	-1	0	1	0	0

Figure 5.2: 7×7 Sobel Vertical Edge Operator

Figure 5.3: Pair of Stereo Images

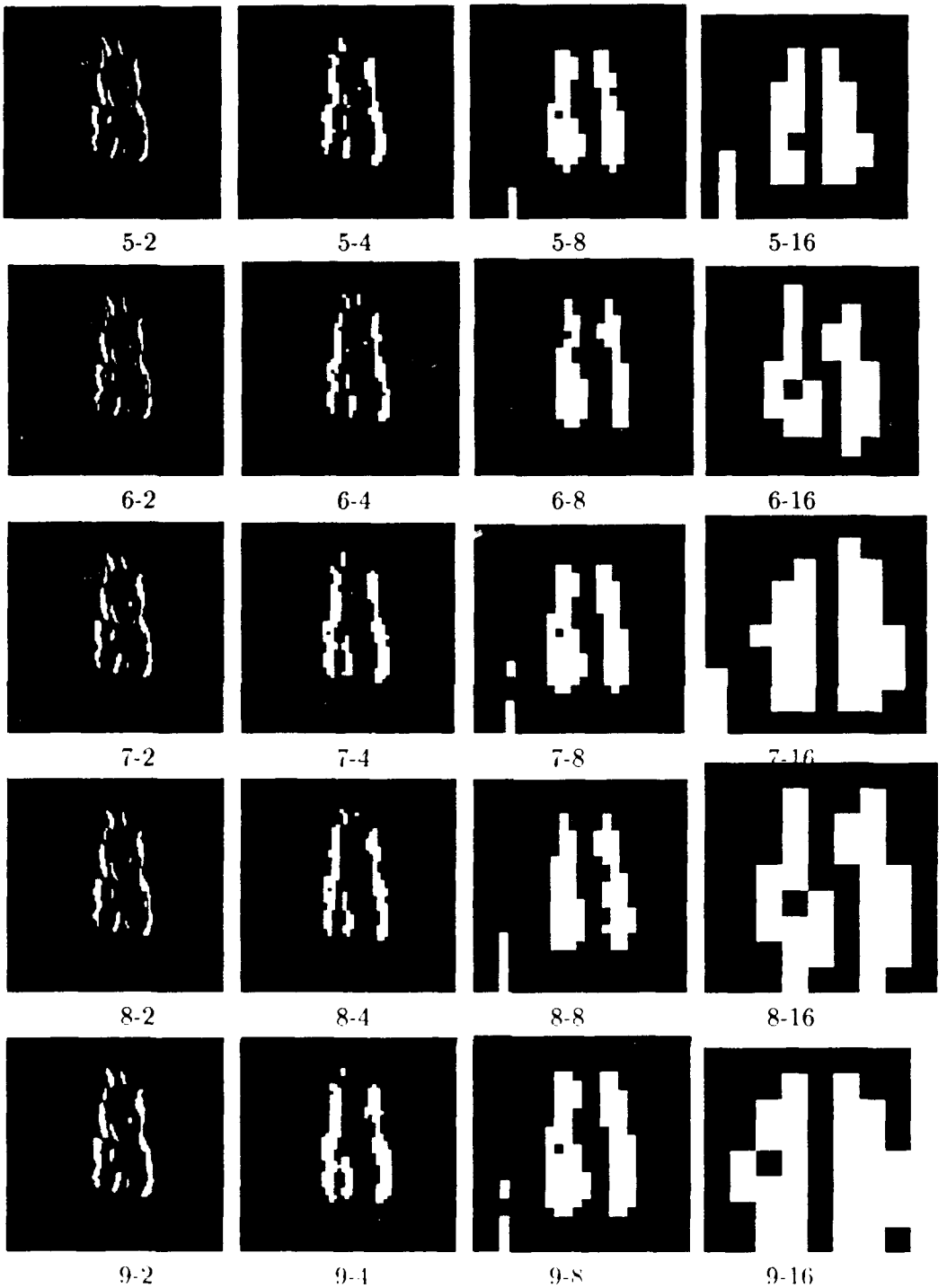


Image labels: Patch Size Reduction

Figure 5.4: Simple Edge Method

The algorithm employs two thresholds to implement the match and suppress noise. The percentage difference threshold (D) limits of difference in edge intensity (or edge contrast) for matching edges, and the percentage strength threshold (S) imposes the minimum strength (contrast) of edges to be considered for matches. For corresponding edge image pixels with coordinates (i,j), $E_L(i,j)$ and $E_R(i,j)$, the resulting zero-disparity pixel is

$$Z(i,j) = \begin{cases} \frac{a+b}{2} & \text{if } |a-b| < D \times 2^B \\ & \text{and } \left|\frac{a+b}{2}\right| > S \times 2^{B-1}; \\ 0 & \text{otherwise.} \end{cases}$$

$$\text{where } a = E_L(i,j) \text{ and } b = E_R(i,j),$$

for B-bit edge images. We have found $D = 16\%$ and $S = 40\%$ to perform reasonably well in the lab. The pixels above threshold can be amplified, and in fact the ZDF is simply binarized in the real time implementation. The list of pixels above threshold is extracted by the FEATUREMAX board, and this process loses the intensity of each feature anyway.

This method works well and has been implemented on MaxVideo hardware to run at field-rates (60 Hz), but it does have some drawbacks. The biggest problem is the high likelihood of aliasing. If there happen to be two above-threshold edges in the same place in each image an edge will exist in the filter output, even if they arise from different objects. Their orientations needn't be the same since the ZDF doesn't use that information. There is another problem; edges at the particular resolution detected by the edge operator may not always be the appropriate feature for matching. Also, the linear nature of the edge-finder step means it is more likely to give high output in high contrast areas of the image. Finally, having to choose a threshold is a problem. Thresholding is an inexact way of eliminating noise, and it can affect the final results in a positive *or* negative way, depending on the original images.

To some extent, blurring (or reducing the resolution of) the intensity images alleviates the aliasing problems. However, blurring may prevent the detection of zero-disparity in fine-resolution features, and blurring tends to reduce contrast as well. The following sections describe attempts to improve the ZDF by applying it at various resolutions and using richer matching and feature-quality measures in the filter itself.

5.3 Neighborhood Correlation Methods

In this section, we describe three extensions to the *simple edge* method: a more sophisticated matching matrix and the addition of a feature quality matrix. The difference image is created by seeing how well corresponding patches match in the stereo images. The feature quality image represents how reliable each match is likely to be.

The difference and feature quality calculations over $m \times m$ patches use nonlinear operations for difference matrices and both linear and nonlinear methods for the feature

quality. Their outputs are combined nonlinearly to make up the filter output. We do not calculate output unless the patch is entirely within the image, so the size of the output array is $m - 1$ less than that of the input array.

A reduction factor can be applied to the original images. Reduction of the images is a simple process that acts as a low pass filter: it is a way of applying matches at different resolutions.

The idea behind the difference image is simple: break the images up into $m \times m$ patches and sum the absolute value of the differences of corresponding pixel values in the patches. The value calculated from each pair of patches is then used as a pixel value in the new difference image. There is a problem with the difference image: objects in the same location in each image will not show up in the difference image if they have differing light intensities. To compensate for the possible brightness difference, a slight addition will be made to the difference algorithm described above. Before subtracting corresponding pixel values, calculate the mean of each patch and subtract it from from every pixel value in the patch. Once this is done, brightness will be compensated for, and the sum of the absolute value of the differences can be calculated. The difference image is a representation of how the two images differ from each other: the lower the value, the better the match. The output resolution is determined by pixel size, not patch size.

It appears as if the difference image is all that is needed to locate the zero-disparity object in the images, but what if there are two patches that are all one shade? They will be considered a good match. This is where the feature quality image comes into play. The function to calculate feature quality is applied to each of the patches viewed by the difference function and its job is to determine if the patches are likely to contain features that can be reliably matched between images. Returning to the example of uniform-intensity patches, it is clear that the feature quality function will consider them a poor feature since there is no variance over the patches, hence they will get a low feature quality value. As the the feature quality value drops, so does the reliability.

Feature quality plays a major role in the discovery of the zero-disparity object in a pair of stereo images. For this reason four methods (*pairwise*, *standard deviation*, *total mask*, and *center mask*) were implemented to acquire a reliable feature quality image, with varying degrees of success.

5.3.1 Pairwise correlation

The first method, which is the *pairwise* method, runs the mask $[-1, 1]$ over the corresponding patches in each image (thereby subtracting neighboring values) adding up all the differences, and takes the lower of the two image's values as a pixel value in the feature quality image. The results are, as one might expect, quite poor.

5.3.2 Standard Deviation correlation

The second method, which is the *standard deviation* method, computes the sum of the row standard deviations of a patch in each image and the lower of the two is taken. The horizontal standard deviation is a way of indicating likely vertical features. Standard deviation could be searched for in all possible directions, but the camera geometry insures we only have horizontal disparities, so horizontal standard deviation is the appropriate feature. The standard deviation formula used is as follows:

$$\sqrt{\frac{\sum_{i=1}^{n} x_i^2 - n\overline{x}^2}{n-1}}$$

Where x is the value of a pixel in the row and $\overline{x}$ is the mean of all the values in the row. The square root of sample variance is used instead of population variance since a sample of numbers is to be used [Devore and Peck, 1986]. Again, the results of this method are rather poor (although not at dismal as the pairwise method).

5.3.3 Total Mask correlation

The third method, which is the *total mask* method, combines several edge operators in the patch. The edge operators used are Sobel vertical edge operators ranging in size from 2×2 to 5×5 (Figure 5.5). As in the pairwise method, the masks are convolved with the patches and the absolute value of the output from the independent mask results are added together and the square root of each sum is taken. The values from corresponding patches are compared and the lower of the two is used as a feature quality pixel value. Although the total mask method is easy to describe this way, a faster and only slightly different method has been implemented. The idea is to add up all the edge operators and run them over the entire image *before* breaking them up into their patch sub-segments. An example of the output of the total mask method can be seen in Fig. 5.6. Adding up all the edge operators is the first difference. The sum of the edge operators does remove the non-linearity of adding the absolute values of the edge operator results, but this difference has little apparent effect on the final result. The total mask is the result of summing four Sobel vertical edge kernels (aligned at the upper-left-hand corner) as shown in Fig. 5.7. The next difference is the over-lapping or under-lapping portions of the new edge operator that do not fit in the patch. When deciding where to end the patch on the new matrix created by convolving the new edge operator over the image, it must be decided whether to run the 5×5 or 2×2 section of the mask to the end of the patch. If the 5×5 mask is run to the edge, then none of the smaller masks will make it all the way to the end. On the other hand, if the 2×2 mask is run to the edge, then each larger patch will overlap into other patches. For the examples and discussion of the total masks method in this paper, the former method is used.

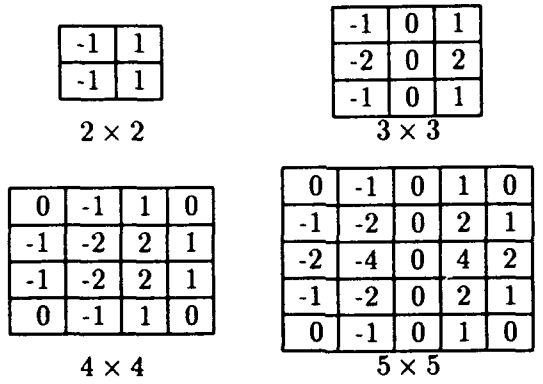


Figure 5.5: Sobel Vertical Edge Kernels

5.3.4 Center Mask correlation

The *center mask* method is similar to the total mask method and uses the same edge operators. The difference is the masks are placed only in the center of the patch before the absolute value of the results from all masks are summed. In the cases where a mask cannot be placed exactly in the center of the patch, an approximation of the center is used. The sum of the masks implementation is appropriate here as well, and the resulting sum of the masks is shown in Figure 5.8. Note: the center mask method does *not* convolve the mask with the stereo images, it is multiplied with the values in the center of the patch only. An example of the output of the center masks method can be seen in Fig. 5.9.

5.3.5 Real-time Implementation Feasibility

Both linear sum-of-the-masks implementations are appropriate for MaxVideo implementations, as is the pairwise method. In each case a VFIR mask is convolved with the image to compute the edge strengths and then a VFIR mask of all 1's is convolved to sum the effects over the patch.

Once the match difference and feature quality images have been produced, they must be combined. First, they are scaled to the range [0,255]. Since low values in the difference image mean a good match, and low values in the feature quality image means poor reliability, one of the two values must be inverted before combining the two. For the images in this paper, the match image was inverted: i.e. $match = 255 - difference$. Finally, we combine the match and feature quality values by pixelwise multiplication and rescale the result to [0,255]. Some approximation to these operations (compromising on scaling) could be done in MaxVideo at frame rates. An example of the total mask

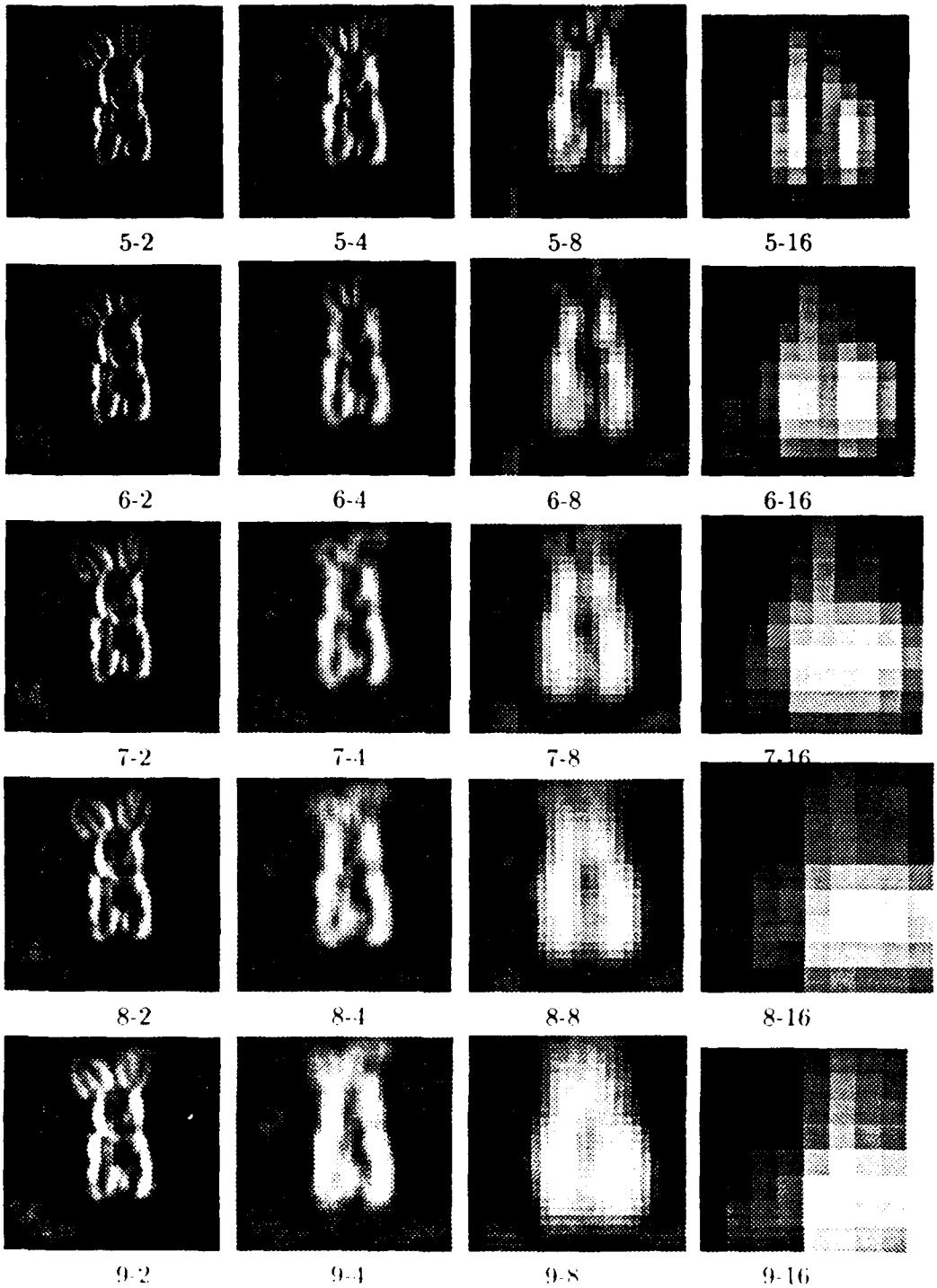
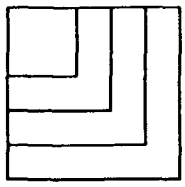


Image labels: Patch Size Reduction

Figure 5.6: Total Mask Method

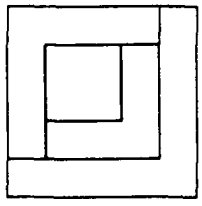


(a)

-2	-1	2	1	0
-5	-3	4	3	1
-4	-6	3	5	2
-1	-3	1	2	1
0	-1	0	1	0

(b)

Figure 5.7: Total Mask pattern of summation of Sobel vertical edge kernels (a) and resulting mask (b)



(a)

0	-2	1	1	0
-2	-6	3	4	1
-3	-9	3	7	2
-1	-4	1	3	1
0	-1	0	1	0

(b)

Figure 5.8: Center Mask pattern of summation of Sobel vertical edge kernels (a) and resulting mask (b)

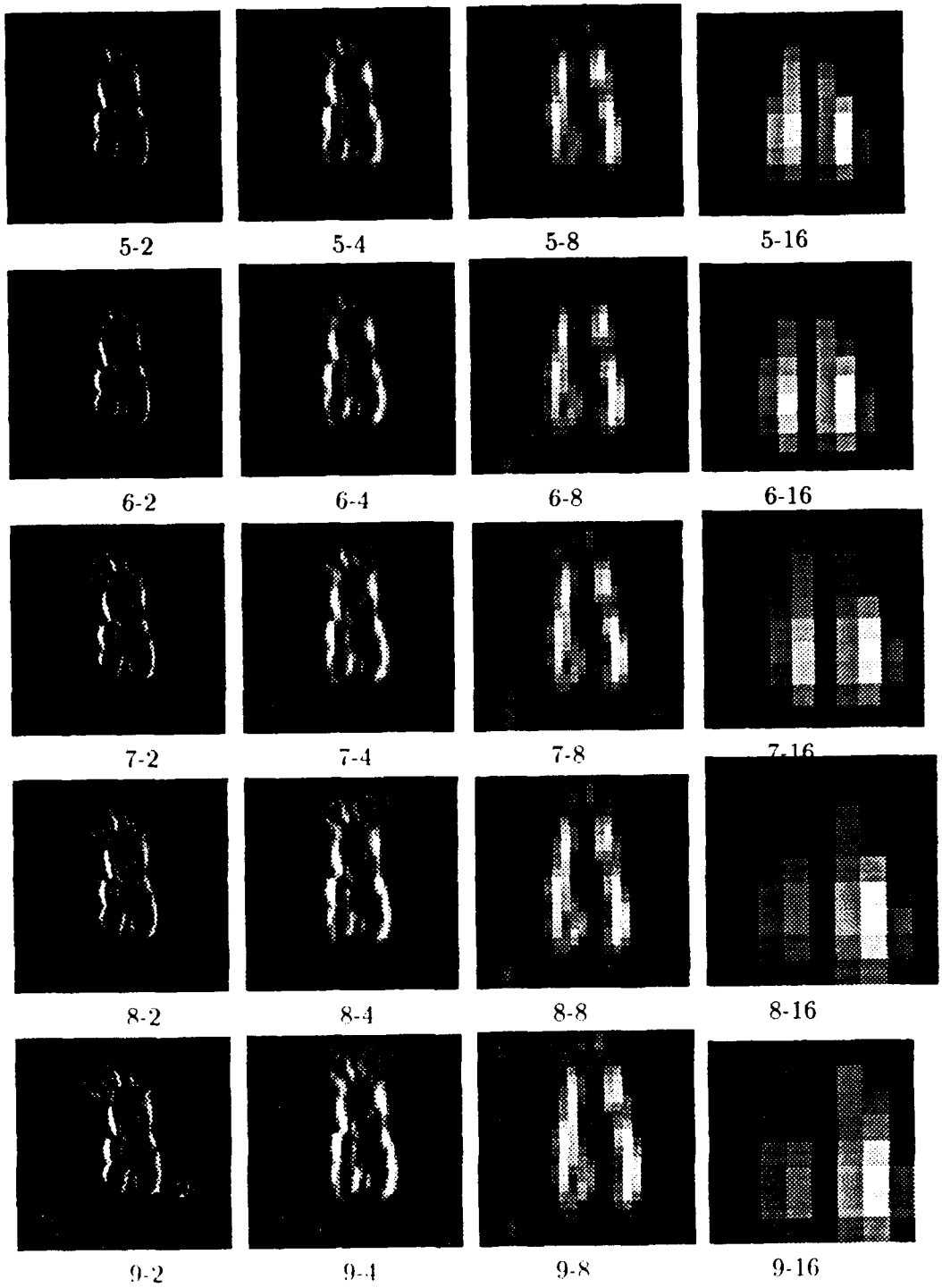
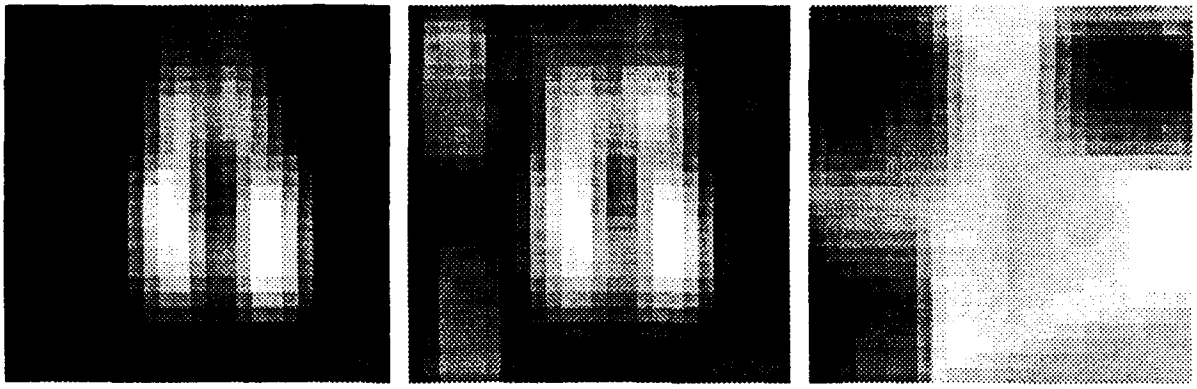


Image labels: Patch Size Reduction

Figure 5.9: Center Mask Method



(a) (b) (c)

Figure 5.10: Total Mask Results on the Bunny Scene (8×8 patch and 8-fold reduction): (a) filter output, (b) feature quality, and (c) inverse difference images.

method result and its component feature quality and difference images are shown in Figure 5.10.

The advanced zero-disparity filters can probably be implemented on the MaxVideo in real time. The image subtraction and smoothing operations can be done in the MaxSP board and VFIR-II, respectively. The nonlinear absolute value step can be done with a lookup table in any of several boards. The most difficult aspect is the need for dynamically scaling the final image. The standard deviation and total mask methods have a built-in way of bypassing this problem. Since the standard deviation is used instead of the variance, the range of values is *usually* within the gray scale values [0, 255] (it is possible to have a pixel value leap out of the range [0, 255], but when tested this never happened). The total mask method stays in the range [0, 255] because the square root of all mask convolutions is taken. Again, it is possible to exceed the gray scale range but this instance never occurred in practice.

5.4 Orientation-Magnitude Edge Matching

It was mentioned in Sec. 5.2 that a significant drawback of the simple edge method is having to use thresholds. In addition, edges of different orientation may appear to match where they cross one another. The *orientation* method was implemented to address these problems. Similar to the simple edge method, it is assumed that features can be adequately modeled as edges. However, each feature is modeled by its orientation and magnitude, rather than simply its response to a vertical edge filter. The orientation method convolve two masks of the same size over both the left and right stereo images

instead of one mask. The mask sizes range from 5×5 to 9×9 and include a horizontal and vertical Sobel edge operator. Instead of thresholding, the horizontal (0 degree) and vertical (90 degree) edge images are combined to produce a magnitude and orientation image that are calculated with the following equations.

$$M(i,j) = \sqrt{E_0^2(i,j) + E_{90}^2(i,j)}$$

$$O(i,j) = \arctan\left(\frac{E_0(i,j)}{E_{90}(i,j)}\right)$$

Once magnitude and orientation images have been created for left and right images (M_L, O_L, M_R, andO_R), they must be combined to form a difference and feature quality image. Thus this method tries not only to use the basic idea behind the simple edge method, but also to include the difference and feature quality of the more advanced methods. The difference is created by the following equation that relies on both the magnitude and orientation of edges.

$$D(i,j) = \sqrt{(|O_L(i,j) - O_R(i,j)| + 1) \times (|M_L(i,j) - M_R(i,j)| + 1)}$$

D is defined thus to prevent a good match in either orientation or magnitude from causing the difference to neglect a poor match in the other dimension. After the difference, D, has been computed, it is put into the range of integers [0, 255] and inverted: *i.e.*, $D(i,j) = 255 - D(i,j)$. This gives good matches a high value.

For the feature quality, we tried two approaches. Both depend on the idea that nearly-horizontal edges are of little use since the stereo images are taken with a horizontal baseline. The first equation only uses orientations within a specified range as follows.

$$(|O_L(i,j)| \wedge |O_R(i,j)| \in [45, 135]) \Rightarrow F(i,j) = min(M_L(i,j), M_R(i,j))$$

$$(|O_L(i,j)| \wedge |O_R(i,j)| \notin [45, 135]) \Rightarrow F(i,j) = 0$$

Where F is the feature quality. This coarse version of the orientation filter performs poorly, resulting in choppy, noisy images. The second equation de-emphasizes low angle measures instead of completely ignoring them. This is done using the sine of the orientation.

$$F(i,j) = min(\sin(|O_L|) \times M_L, \sin(|O_R|) \times M_R)$$

Multiplying the magnitude by the sine of the orientation de-emphasizes small angles. Fig. 5.11 shows the output of the sine version of the orientation method. A more extreme method of de-emphasizing the small angles by multiplying the square of sine by two was tried, but the results did not differ much from sine.

When the feature quality has been calculated, it is scaled to the range of integers [0, 255]. Then the difference and feature quality are combined using pixelwise multiplication.

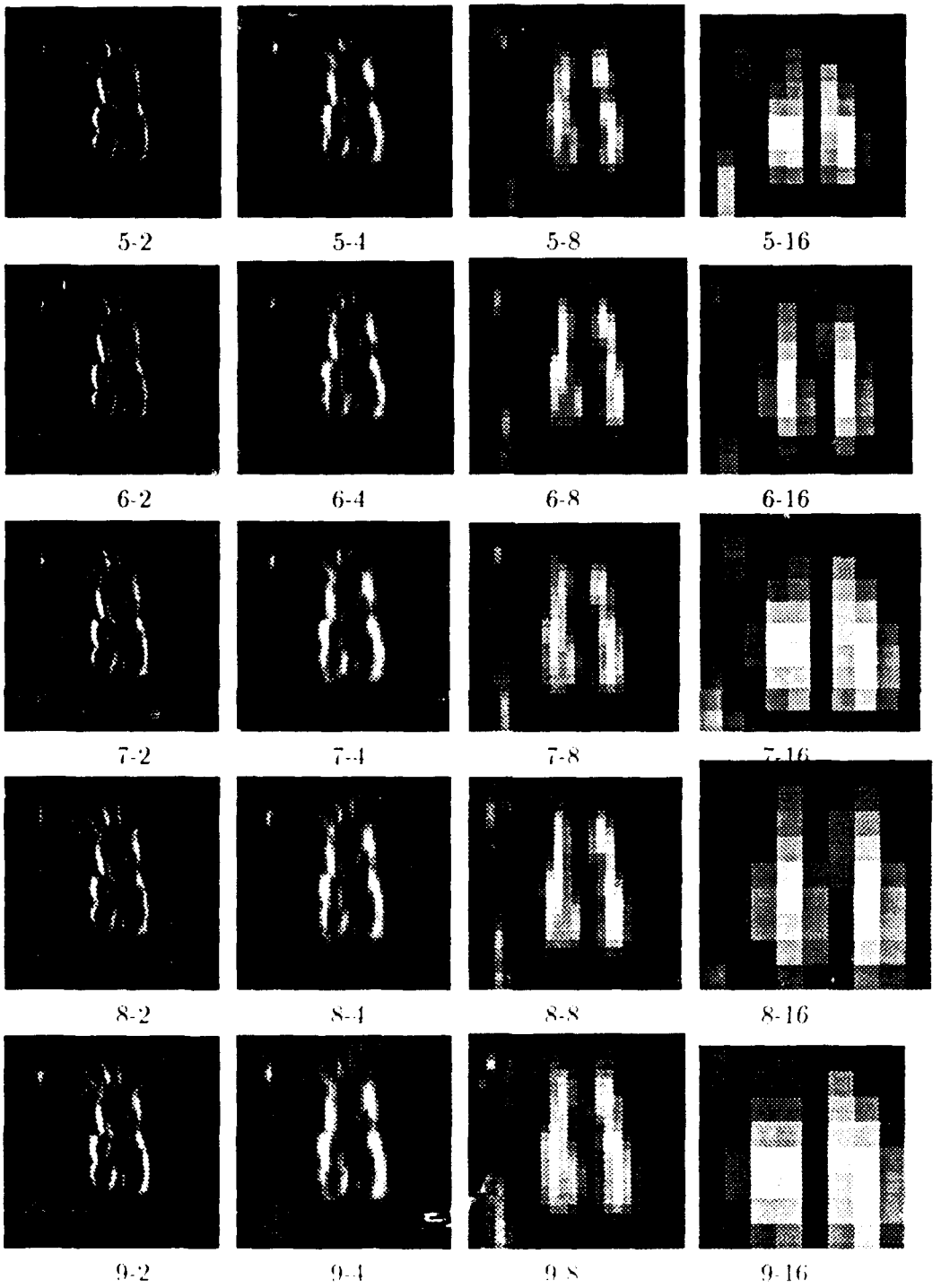


Image labels: Patch Size Reduction

Figure 5.11: Orientation Method Using Sine

(a) (b) (c)

Figure 5.12: Total Mask on a more difficult scene (10 × 10 patch and 4-fold reduction): (a,b) stereo images of Snoopy on a lab cart, and (c) the result of the Total Mask method.

5.5 Evaluation of Results

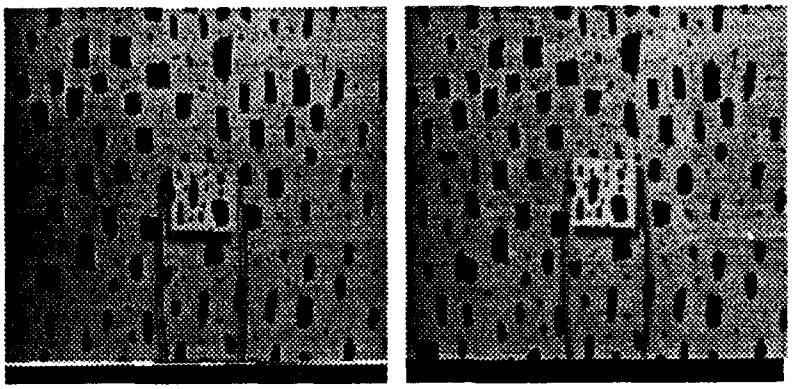
5.5.1 The Best of Each Method

The method with the worst results is the pairwise method. Generally it reduced the least amount of noise. On the bunny image, its best results are at a reduction of four. Any more reduction and the bunny loses its head; any less and we lose some of the fill-in of the bunny. At reduction four there is much noise. The best patch size is around six or seven; those choices seem to eliminate the most noise and leave plenty of bunny.

The horizontal variance method produces its best images when reduction is either four or eight and when the patch size is around seven. At reduction four, there is the least amount of noise, but there is also less fill-in in the bunny. Reduction eight increases the fill-in but it also results in increased noise.

The total mask method has good results when the reduction is set to either four or eight. Both have low noise and good fill of the bunny. Reduction four has less fill and less noise, while reduction eight has more noise and more fill. Reduction eight seems better since the added noise is not considerable. Larger patch sizes yield better results. When the patch size is increased to nine, there is very little noise and the fill-in is good. Hence, a patch size of nine with reduction eight gives good results. The stereo images of a more difficult scene and the result of the total mask method are shown in Figure 5.12.

The center mask method has best results when the reduction is either four or eight. Both have low noise, but reduction eight has less resolution and fill than would be nice. Reduction four has little fill, but without compromising resolution. Patch sizes seven, eight, and nine are all similar so choosing one above the other would be difficult.



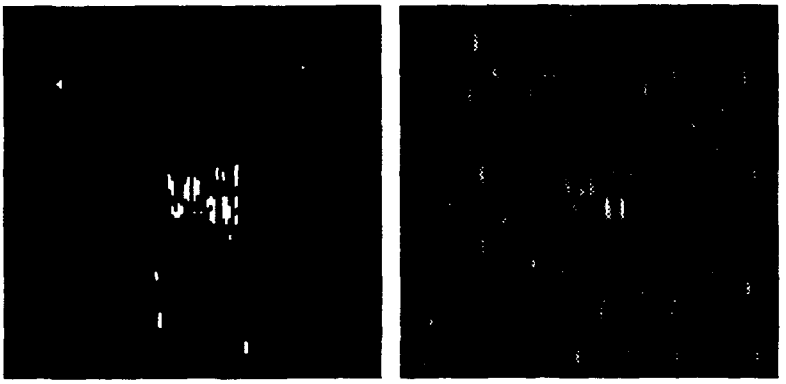
(a) (b)

Figure 5.13: Pair of Leopard Spot Stereo Images

5.5.2 The Best Method

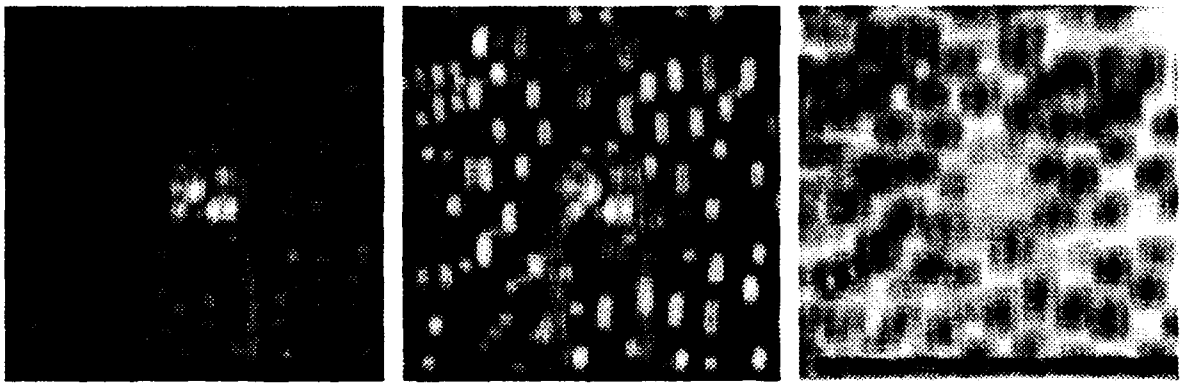
The best of the four methods tested is the total mask method. We should now compare its results to the results of the simple edge method on more difficult stereo images. Figure 5.13 is a pair of "leopard spot" stereo images. The center is the zero-disparity area. Figure 5.14 shows the results of the edge-based methods. Interestingly, the simple edge method performs notably better than expected, and the orientation method produces the worst results of these three methods. Figure 5.15 shows the output from the total mask ZDF and its component feature quality and inverse difference images. There still is noise in the final image produced by the total mask ZDF, but notice how the difference image reduced that amount so that the zero-disparity location is distinguishable from the rest of the image. The results from total mask may not appear very good on paper, but this is largely due to the dithering of the printer, which over-emphasizes low intensities. Figure 5.16(a) indicates with reverse intensity the row of the image whose intensity is plotted in Figure 5.16(b). The graph contains the pixel values from row 60 in total mask's final result. Clearly the noise values are not as large as they appear in the dithered image.

The performance of the simple edge method on the leopard spot stereogram is pretty good, but performance suffers considerably on a computer generated random dot stereogram. Figure 5.17 is a random dot stereogram that is 50% black and 50% white. They are the same except for a patch in the center that is slightly skewed to one side. This leaves a patch of non-zero-disparity in the center of the image. Figure 5.18 shows the results of edge-based ZDFs. As noted, the simple edge method performs worse on this stereogram. Again, the orientation method performs badly. The reason, of course, for the poor performance of the edge-based methods is that the edge is a poor model of



(a) (b)

Figure 5.14: Edge-based ZDFs on Leopard Spots: (a) simple edge method (7×7 patch and 4-fold reduction), and (b) orientation method (8×8 patch and 4-fold reduction).



(a) (b) (c)

Figure 5.15: Total Mask Results on Leopard Spot Images (8×8 patch and 4-fold reduction): (a) filter output, (b) feature quality, and (c) inverse difference images.

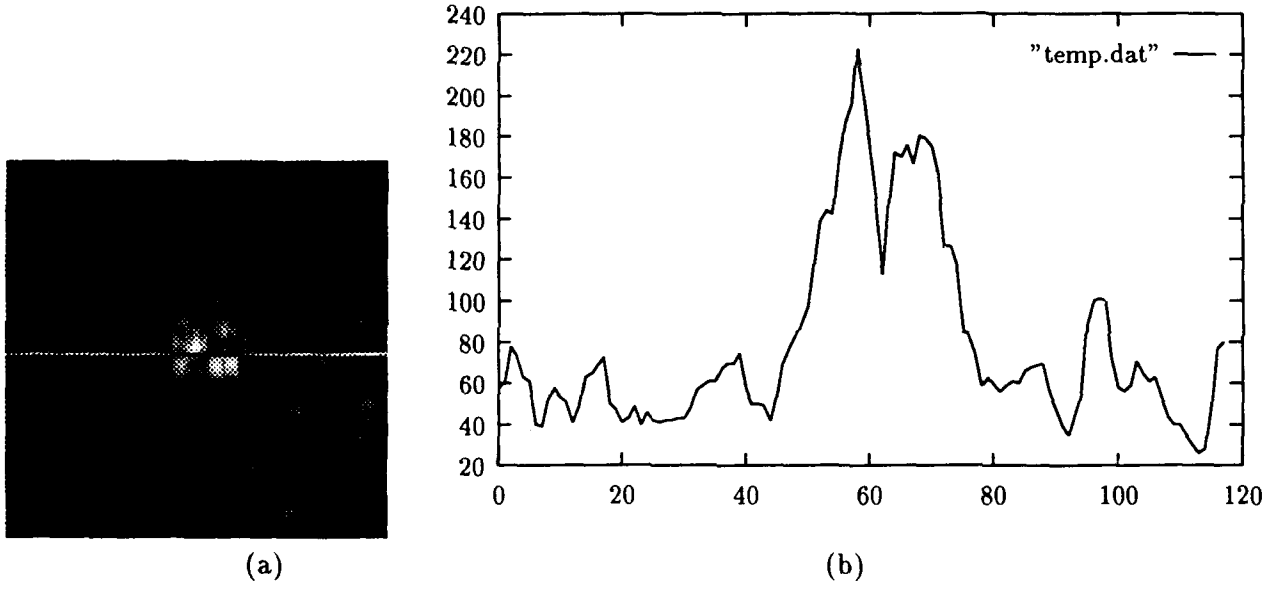


Figure 5.16: Total mask result on the random dot image: row 60 (indicated by inverse intensity in (a)) is plotted in (b).

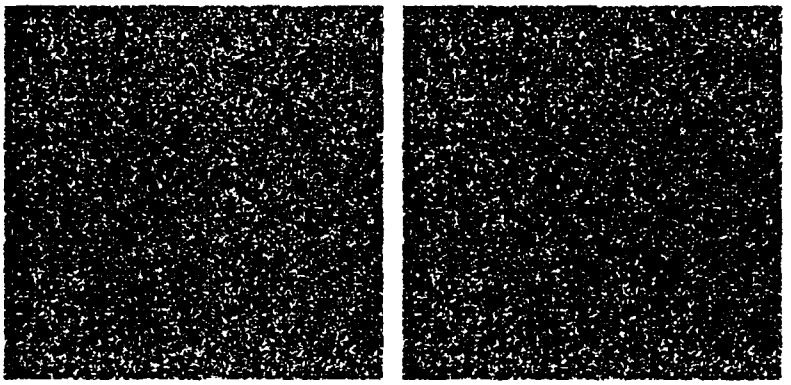


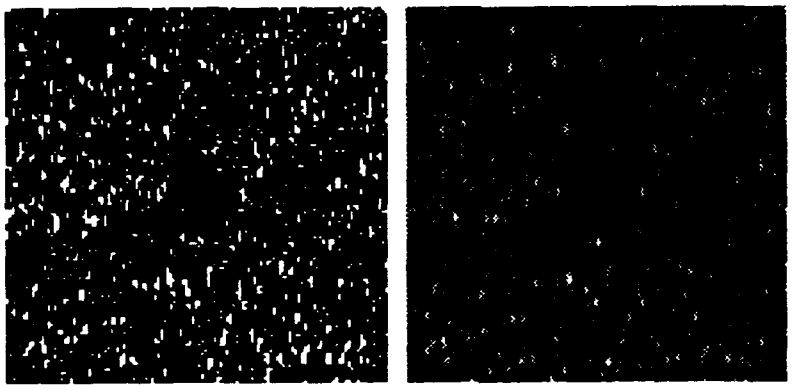
Figure 5.17: Computer Generated Random Dot Stereo Images

the correlation features in random-dot stereograms. Figure 5.19 shows the output from the total mask method and its feature quality and inverse difference images. Notice that the feature quality was uniformly good for neighborhood correlation, and only the difference image is able to distinguish the zero-disparity region.

5.6 Summary

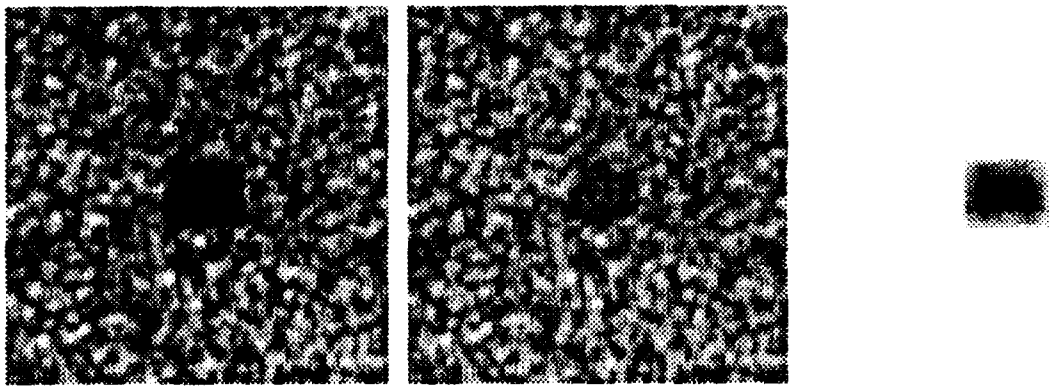
The simple edge method works surprisingly well, but it seems the total mask and center mask methods work better. The pairwise method is the worst of all methods tried, and the result from the standard deviation method works as well as or a little better than the simple edge method. Of the third and fourth methods, the third (total mask) is the better of the two.

Results from simple edge method work well in most cases, but in a pathological but noise-free case (computer produced random dot stereo images) it does not work very well. The total mask method works well even in the random dot case. The orientation method has produced mixed results; it appears as if it only really works well on simple examples like the bunny images. Therefore, of all methods tried the most general is the total mask method.



(a) (b)

Figure 5.18: Edge-based ZDFs on Random Dots: (a) simple edge method (7×7 patch and 4-fold reduction), and (b) orientation method (7×7 patch and 4-fold reduction).



(a) (b) (c)

Figure 5.19: Total Mask Results on Computer-Generated Random Dot Image (8×8 patch and 4-fold reduction): (a) filter output, (b) feature quality, and (c) inverse difference images.

6 Coping with Delays in Feedback Systems

Delays are pervasive in real systems. Signals do not travel instantaneously, and computing units like neurons and software and motors take time to respond to their inputs. Visuomotor systems of primates and robots are no exceptions. Visual processing takes a substantial amount of time. Executing the control algorithm to generate the appropriate motor control signals takes time, and there are also delays within the motor system.

Whether a system is best described by a continuous or discrete model, delays in the system complicate the problem of using feedback to control the system and getting the system's response to be on time.

Consider the relatively simple case of the Rochester Robot's visuomotor system, which can be modeled to first order as a discrete system with a delay of one sample interval in visual processing, as shown in Figure 2.1. Suppose the tracking system needs to follow a target that jumps to a new position. The error is not apparent to the control system immediately, but once it is observed the system responds. The system's response is not immediately noticed due to the processing delay, so the response to the old observed error is sustained. Consequently, the system over-reacts to the input signal. If the delay is longer, the system will react to older sensory data, and the resulting overshoot will be greater.

One way to cope with delays in a feedback system is to reduce responsiveness to prevent over-reactions. A better approach incorporates knowledge of the delay in the system to avoid over-responding. For instance, consider the internal positive feedback controller in an early model of the pursuit system [Young and Stark, 1963]. The system tries to match the velocity of the target, so consider a target that changes its speed to a new constant velocity. The controller needs to respond to a step input. An internal model of the delays in the oculomotor plant and visual processing feeds back the anticipated error signal at the same time as the negative feedback loop (through the world and vision) delivers the observed error signal. If the internal model is correct, it exactly balances the negative feedback signal, and no error is observed. Thus the controller needs only to generate changes in camera velocity to respond to observed retinal slip. Thus, if the new camera velocity exactly compensates for retinal slip, the system will

exhibit only a lag of a single time interval while the visual system passes on the observed error in camera velocity. Note that the gains can therefore be increased since the system will be stable.

Although internal positive feedback can make the system stable despite delays, the response will remain late. The only way to avoid being late is to predict the target signal's behavior in advance. This is possible for signals that are predictable (*e.g.*, because a model of the signal's behavior is available), or for signals that can be approximated by linear predictors (*e.g.*, a Kalman filter [Bar-Shalom and Fortmann, 1988]). It is important to make the predictions in a coordinate system that is not perturbed by the control system so the target signal will be stable in that space. For instance, the target trajectory may be relatively smooth in head-centered coordinates although the target's motion is confounded with camera movements in retinotopic visual space.

The remainder of this chapter considers these concerns at greater length, and it is based on work reported in [Brown and Coombs, 1991].

We gather together some introductory and tutorial material on control systems with delay. Delay is especially pernicious in feedback systems, and some form of modeling and prediction is essential to overcome its effects. After a short introduction to the issues, we present four basic techniques for overcoming delay and compare their performance through simulation. The four techniques are the following.

- Predictive techniques to cancel the delay within the loop.
- Cancel negative feedback to obtain open-loop characteristics.
- Smith prediction, which uses a model of the plant.
- System inversion techniques, which use a model of the controller and the plant.

In each of these techniques, predictive filters can be used to overcome latency by providing approximate dynamic predictions of waveforms within the system, such as input and control signals.

6.1 Feedback Control and Delay

We are interested in control systems with delay. Feedback control has several familiar advantages—one important one is the decreased sensitivity of a closed-loop negative feedback system to variations in its parameters. Since the open- and closed-loop systems are not directly comparable, instead we repeat the familiar argument (*e.g.*, [Dorf, 1980]) about the decreased sensitivity of feedback systems to parametric variation. Clearly in an open loop system with plant transfer function $G(s)$ and input and output $X(s)$ and

$Y(s)$, the change in the transform of the output due to a parameter variation $\Delta G(s)$ is clearly

$$\Delta Y(s) = \Delta G(s)X(s). \tag{6.1}$$

For a closed loop system, find $Y(s) + \Delta Y(s)$ by substituting $G(s) + \Delta G(s)$ into the familiar formula $Y = XG/(1 + GH)$, assume $GH(s) >> \Delta GH(s)$, and obtain

$$\Delta Y(s) = \frac{\Delta G(s)}{(1 + GH(s))^2} X(s). \tag{6.2}$$

Defining the system sensitivity to be the ratio of the percentage change in the system transfer function to that of the process transfer function, we see that open loop systems have sensitivity unity, but closed loop feedback systems have sensitivity $1/(1 + GH(s))$, the denominator of which is usually much greater than one.

We bring up this point here only because one of the schemes we shall examine later chooses to sacrifice closed-loop advantages in order to deal with delay. Given that ideally we desire feedback, then why is delay a problem in feedback control systems?

We illustrate with the constant-gain feedback system illustrated in Fig. 6.1(a). Here the transfer function is simply $K/(K+1)$, and the response to a step function is a scaled step function. Thus this system tracks the input perfectly. Fig. 6.1(b) shows the same system with a delay of τ seconds added.

The responses of these systems to a unit step are shown in the next few figures. Not shown is the perfect step function output of height 0.4117 for $K = 0.7$ that exactly tracks the input in the continuous controller, zero delay case. Fig. 6.2 shows the response of the first order *discrete* feedback loop

$$y(kT) = K[u(kT) - y((k-1)T)] \tag{6.3}$$

to a step input. This discrete controller has an implicit sample-and-hold circuit in the feedback branch, and this characteristic gives the discrete realization of the controller some of the aspects of delayed control. That is, there is an inherent minimum delay equal to the sampling interval that must exist somewhere in any closed-loop discrete system, even if no excess delay is present. These qualitative differences motivate the use of the Z transform in discrete system analysis. Thus, discrete systems are often written in the manner of Fig. 6.1(b) with the Laplace transform delay box relabeled as a Z-transform delay of z^{-1}, indicating the inherent delay implied by stepwise operation. This system is described by

$$y(kT) = K[u((k-1)T) - y((k-1)T)],$$

with delay in the feedforward branch, in contrast to system 6.3, whose delay is located in the feedback. Fig. 6.3 shows the effect of delay in the *continuous* version of the controller for $K = 0.7$, and Fig. 6.4 shows the unstable response that results when the gain exceeds unity ($K = 1.1$).

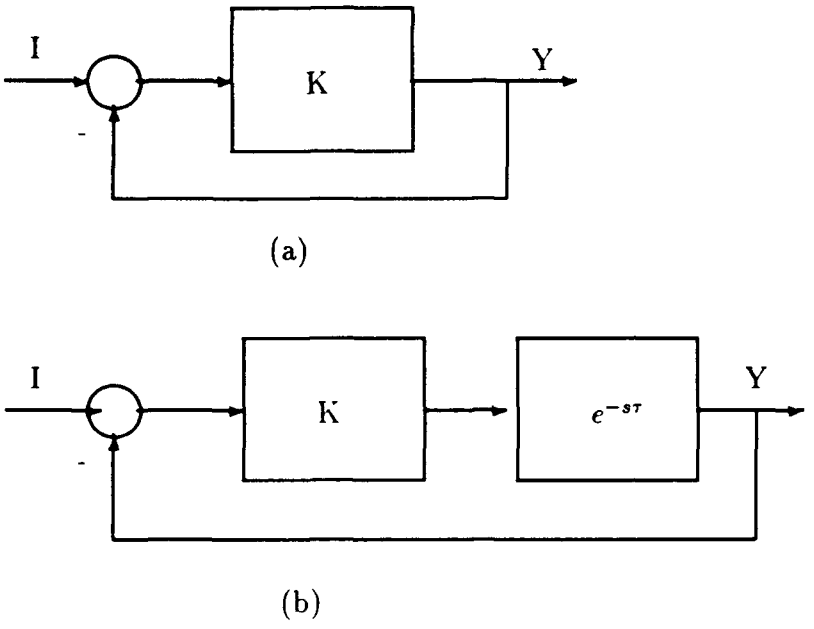


Figure 6.1: Constant gain feedback system: (a) without delay, (b) with delay.

The decidedly discontinuous performance of the continuous system in the presence of delay requires some explanation. The following explanation appears in [Marshall, 1979], but we repeat it here for completeness. We have the following situation, where the delay is τ seconds and we let the time t vary continuously from 0. A step input occurs at time $t = 0$.

$$y(t) = \begin{cases} 0 & t < \tau \\ K & \tau < t < 2\tau \\ K(1-K) = K - K^2 & 2\tau < t < 3\tau \\ K(1-K(1-K)) = K - K^2 + K^3 & 3\tau < t < 4\tau \\ \vdots & \end{cases}$$

Solving for the closed form of the series, we find that

$$y(t) = \frac{K + (-K)^{n+1}}{1+K} \quad \text{for} \quad n\tau < t < (n+1)\tau. \tag{6.4}$$

The steady state gain for the delayed system (if it stabilizes) is the same as for the continuous or discrete delayed system steady-state gain: the exponential term in (6.4) vanishes if the magnitude of K is less than unity.

One can approach the behavior of the delayed system using Laplace transforms. This method puts the problems caused by delay in terms of the delay of the output

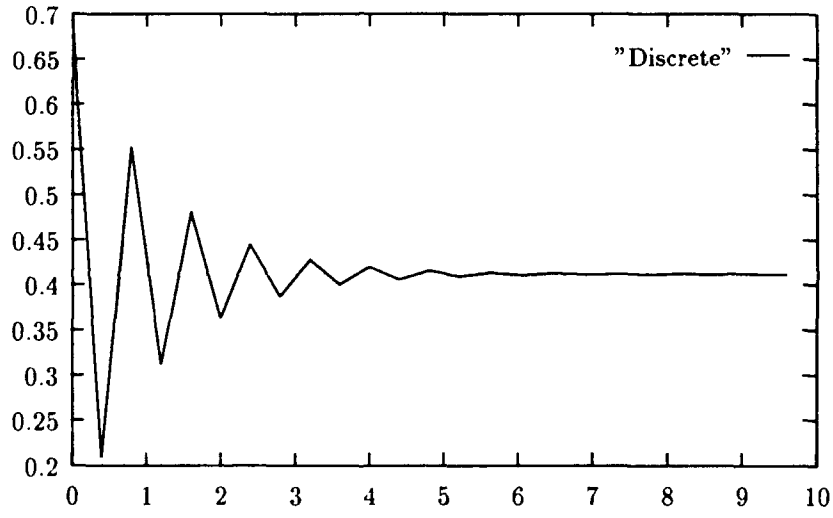


Figure 6.2: Output of constant gain feedback system for step input and discrete control with K = 0.7. Continuous control yields a perfect step.

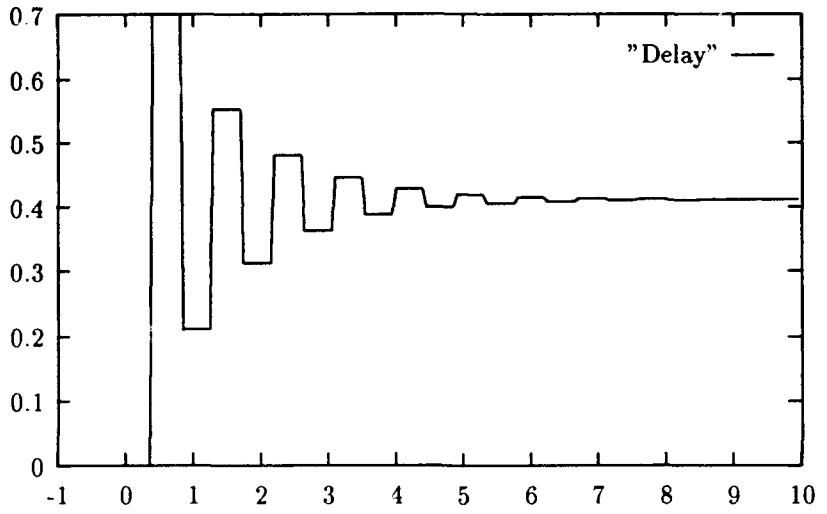


Figure 6.3: Output of delayed constant gain feedback system with step input, $K = 0.7$.

signal and the poles introduced into the system by the delay. (For an introduction to basic concepts such as "poles", see [Dorf, 1980] or any other introduction to control theory.) In the following sections of this report we shall investigate different methods for eliminating the unpleasant effects of the delay and the poles.

The Laplace transform of the output $y(t)$ given input $x(t)$ with Laplace transform $X(s)$ is

$$\mathcal{L}(y(t)) = Y(s) = \frac{Ke^{-s\tau}}{1 + Ke^{-s\tau}}X(s) = \frac{K}{K + e^{s\tau}}X(s). \tag{6.5}$$

In the latter form it can be seen that the characteristic equation is not algebraic, and that it has an infinite number of closed-loop poles. In fact, the poles are the roots of

$$e^{s\tau} = -K = Ke^{j(\pi \pm 2\pi q)}, q = 0, 1, 2... \tag{6.6}$$

and so

$$s\tau = \ln K + j(\pi \pm 2\pi q), q = 0, 1, 2... \tag{6.7}$$

Taking $K = 1$ gives poles falling along the imaginary axis, spaced by $2\pi/\tau$, the two principal ones (closest the origin) at $\pm j(\pi/\tau)$ representing the fundamental oscillatory

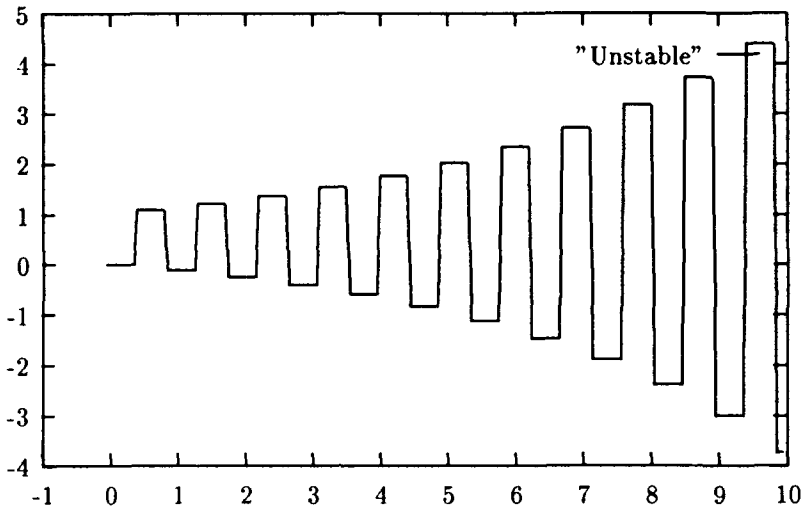


Figure 6.4: Output of delayed constant gain feedback system with step input, $K = 1.1$.

frequency of period 2π visible in the output. This unstable system then diverges for $K > 1$, since poles move over into the right half-plane. The extension of eq. (6.5) to a system with a controller (transfer function C), a plant (G) and delay ($e^{-s\tau}$) is

$$\frac{CGe^{-s\tau}}{1 + CGe^{-s\tau}} X(s), \tag{6.8}$$

which we shall see again.

In an actual situation, delay again manifests its presence with oscillations. In a straightforward tracking application with the Rochester robot head, one camera on the head is to track a moving object. With the camera moving, the spatial positional error of the camera's axis is calculated using information about the camera's position (obtained from reading back angles from the camera's motors) and the image-coordinate error calculated as the distance of the target's image from the origin of the image coordinate system. If the time between reading the two necessary data (camera position and image position) is an unmodelled delay, the system performs as shown in Fig. 6.5.

Table 6.1 sums up these generalities. Compared with open-loop control, a closed loop system can be more resistant to variations in behavior induced by plant parameter variations. Introducing delay into an open loop system simply delays the output, while in a closed loop system it can also introduce instability. In the sections that follow

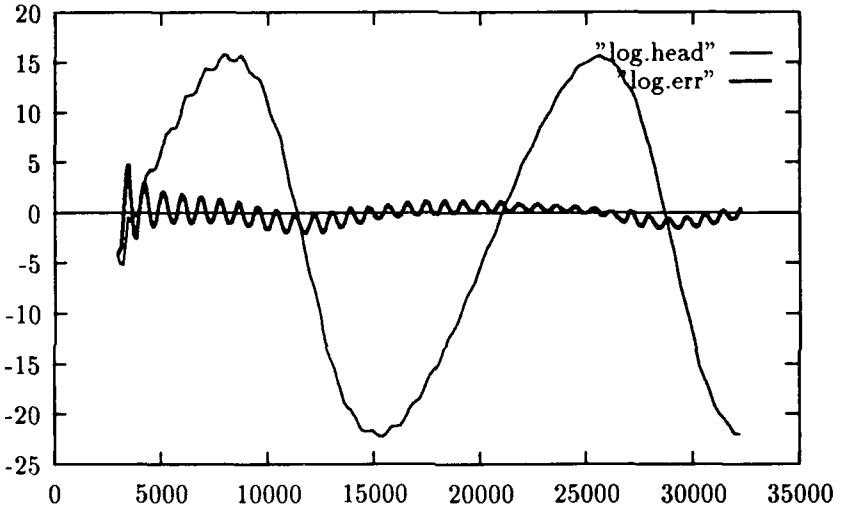


Figure 6.5: Tracking a sinusoidally moving target. Delay in calculating camera's true angular position results in an oscillating error that is superimposed on a generally correct sinusoidal head motion. Here the head position graph is from direct readout from the motors, and the error is the target's retinal position measured with vision.

	Open Loop	Closed Loop
No Delay	basis for comparison	more robust
Delay	output delayed	output delayed, poles affect performance

Table 6.1: Characteristics of undelayed and delayed systems.

we present five different ways to cope with the effects of delay and try to make some comparisons between them. We shall investigate the following techniques. In every case the starting point is a closed loop system with delay, having the output delay and multiple-pole characteristic equation discussed above.

- Cancel negative feedback—achieve open loop performance, delayed.
- Smith prediction—attain closed loop performance, delayed.
- Signal synthesis adaptive control—attain closed loop performance, undelayed.
- Smith and input prediction—attain closed loop performance, undelayed.
- Predictive techniques to estimate and then predict delayed signals within the loop—with perfect prediction, attain closed loop performance, undelayed.

6.2 Opening the Loop

In studying primate gaze control, Young [1977] wanted to explain how smooth pursuit avoided instability if tracking is modeled as a pure negative feedback system. There are two problems with this model. First, the error, and thus control, signal is zero when accurate tracking is achieved; this should send eye velocity transiently to zero. Second, tracking performance is better than it should be given the delays in the control loop and the time constants of the processes. His proposal is that the system tracks not the retinal image, but a neural signal that corresponds to target motion (in the world).

Robinson [1987] describes a mechanism that implements Young's idea: as Robinson says, "if negative feedback bothers you, get rid of it". In the negative feedback system the eye velocity is fed back and subtracted from the target velocity (with some delay). If the eye is in the process of tracking, then the target velocity is the sum of the eye velocity (with respect to the head) and the target's retinal velocity (its velocity with respect to the eye). But the latter is just the error signal resulting from negative feedback. Thus an estimated target velocity signal can be constructed by positively feeding back the commanded eye motion into the control loop, delayed to arrive at the proper time to combine with the error term produced by negative feedback. This mechanism not only provides a signal based on the target's true motion, but it cancels the negative feedback and thus removes the possibility of oscillations.

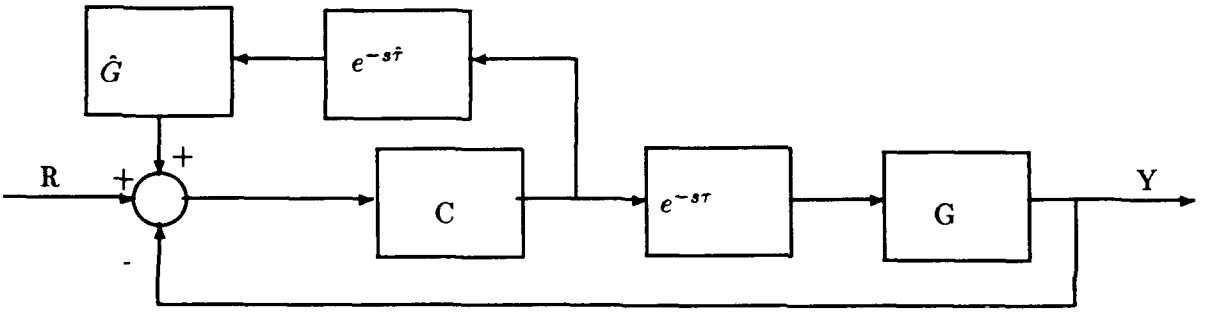


Figure 6.6: Feedback cancellation.

In all our examples we shall have a controller in the loop with the plant. Thus the block diagram of the feedback-cancellation idea is shown in Fig. 6.6. Applying the positive feedback idea in this context involves constructing a model plant $\hat{G}$. If the designer's model of the plant and delay is correct, the system is changed to open-loop, losing all the well-known advantages of closed-loop control. Further the open-loop response is in fact delayed. Given accurate plant and delay models, the designer can do better, as we shall see.

6.3 Smith Prediction

Smith prediction [Smith, 1957; Smith, 1958] is a by-now classical technique, and it is the basic idea behind most modern methods. The treatment in [Marshall, 1979] is especially readable. Smith prediction was one of the main tools for managing cooperating delayed controls in the simulation studies of the Rochester Robot [Brown, 1990a; Brown, 1990b; Brown, 1990c].

Smith's Principle is that the desired output from a controlled system with delay τ is the same as that desired from the delay-free system, only delayed by τ. Let the delay be τ, the delay-free series controller be $C(s)$, the desired delay controller be $\tilde{C}(s)$ and the plant be $G(s)$. The delay-free system transfer function will be

$$\frac{CG}{1+CG}.$$

The delay system with its desired controller has transfer function

$$\frac{\tilde{C}Ge^{-s\tau}}{1+\tilde{C}Ge^{-s\tau}}.$$

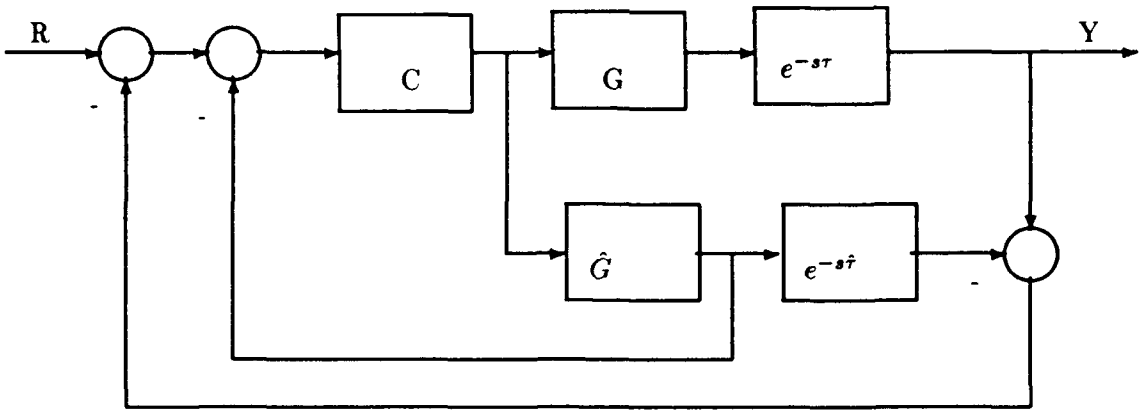


Figure 6.7: Smith prediction control.

But Smith's Principle is

$$\frac{\tilde{C}Ge^{-s\tau}}{1+\tilde{C}Ge^{-s\tau}} = \frac{CG}{1+CG}e^{-s\tau}.$$

This quickly leads to the specification for the controller $\tilde{C}$ in terms of C, G, and $e^{-s\tau}$:

$$\tilde{C} = \frac{C}{1+CG(1-e^{-s\tau})}. \tag{6.9}$$

This simple principle has spawned a number of related controllers, often arising from each other by simple block-diagram manipulation. It is worth noting that Smith did not take the next step and demand that the output should not be delayed. To take this step requires either a non-physical or a prescient component in the system, as we shall see.

Fig. 6.7 shows the Smith predictor as applied to the standard situation we use in this chapter, i.e. with continuous control.

This diagram can be rewritten to show the relation to the feedback cancellation technique (Fig. 6.8).

Assuming exact estimates of G and τ, i.e. that $\hat{G} = G$ and $\hat{\tau} = \tau$, then the positive feedback coming to summation point $S1$ and the negative feedback coming to summation point $S2$ in Fig. 6.8 cancel, and removing them from the diagram yields a simplified diagram in which Smith's principle clearly holds—the transfer function is

$$\frac{Y}{I} = \frac{C}{1+CG}Ge^{-s\tau} = \frac{CGe^{-s\tau}}{1+CG},$$

simply a delayed version of the undelayed closed-loop transfer function (Fig. 6.9).

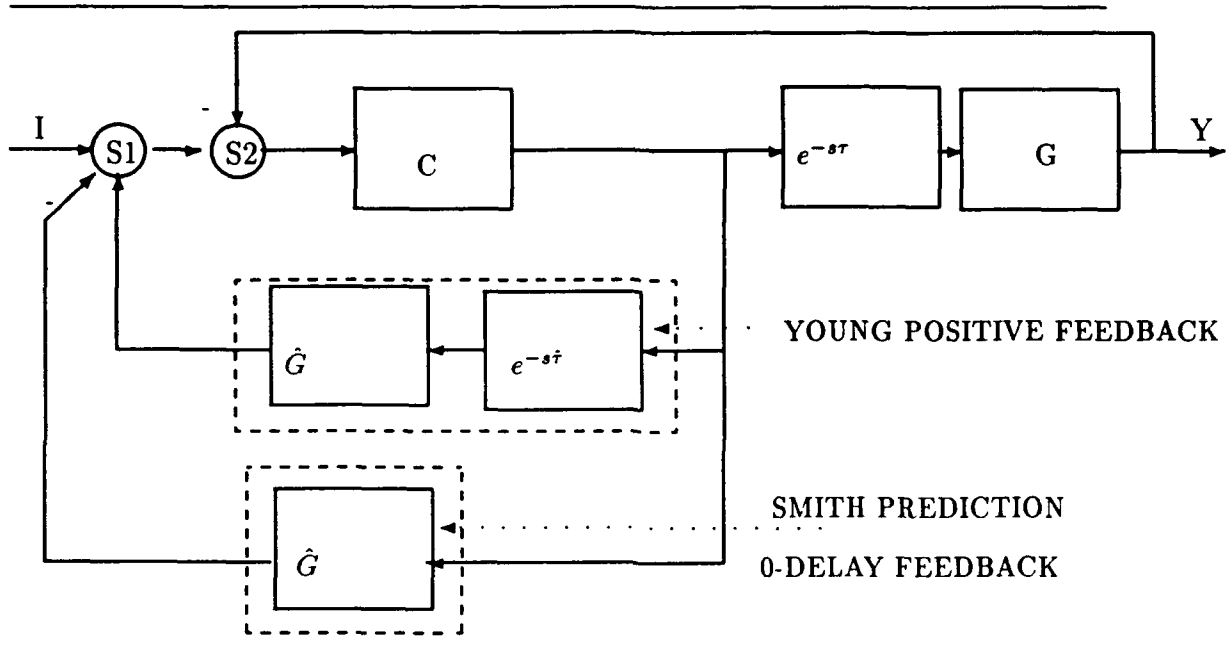


Figure 6.8: Smith prediction and feedback cancellation.

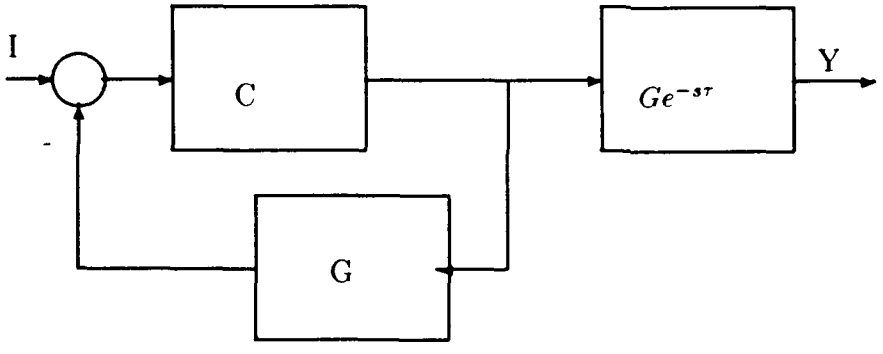


Figure 6.9: Cancellation yields Smith's principle.

To sum up, Smith prediction uses the model of the plant in a negative feedback scheme—the controller controls this model. If the model is good, one gets a delayed version of the closed-loop control. That is, the transfer function of the delayed system is changed from

$$\frac{CGe^{-sT}}{1+CGe^{-sT}} \text{ to } \frac{CGe^{-sT}}{1+CG}.$$

The delayed closed-loop negative feedback is there also, and the hope is that it would tune up the infelicities in the performance due to inaccurate temporal or parametric (plant) modeling. A result of the delay institutionalized in Smith control is non-zero latency, inducing a steady-state error that can vary depending on the input signal. For position-error tracking of a constant velocity target, the latency would cause a steady-state constant positional error with the tracker trailing behind the target.

6.4 Signal Synthesis Adaptive Control

6.4.1 Background

Humans alone are able to track periodic targets without latency. Primates do not follow any targets without latency, and humans are not able to anticipate non-periodic targets. (Target signals described by the summation of even a few sine waves of appropriately varied frequencies appear "unpredictable" enough to prevent humans from tracking them without latency.) Bahill and his coworkers [Bahill and Harvey, 1986; McDonald and Bahill, 1983] propose a scheme called signal synthesis adaptive control to achieve the advantages of undelayed feedback control with zero latency. As we shall see, this requires predicting the future, something that control theorists are not fond of doing. The scheme is called "signal synthesis" control because the controller is predicting the input signal. It is called "adaptive" because in the published formulation there was a scheme to switch between predicted signals to keep the prediction in line with reality (if a trajectory velocity curve changed from square wave to sine wave, for example).

The derivation of this type of control presents us with a powerful technique. In fact, on paper it is possible to achieve many different sorts of control, including canceling the latency but leaving the poles, canceling the poles but leaving the latency, canceling both, and canceling both and substituting the effect of an entirely new controller for the existing controller C. As described, the technique simply uses an inverse plant. These are hard to engineer, since for a plant consisting of integrators one obtains an inverse consisting of differentiators. Still, we shall see that we can rewrite the resulting inverse plant to be realizable and in fact we shall see it has a close relation to the techniques we have seen so far.

6.4.2 Inverting a Delayed Feedback System

As a warm up, let us get some intuition by computing the inverse of our standard delayed feedback system (Fig. 6.10(a)). The system is

$$Y(s) = \frac{C(s)G(s)e^{-s\tau}}{1 + C(s)G(s)e^{-s\tau}} I(s) \tag{6.10}$$

The inverse of the system, remembering that Laplace transforms compose by multiplication, is just the reciprocal of this expression:

$$Y^{-1}(s) = \frac{1 + C(s)G(s)e^{-s\tau}}{C(s)G(s)e^{-s\tau}} I(s), \tag{6.11}$$

which implies that

$$Y^{-1}(s) = I(s)(\frac{1}{C(s)G(s)e^{-s\tau}} + 1) = I(s)(\frac{e^{s\tau}}{C(s)G(s)} + 1). \tag{6.12}$$

Translating the rightmost expression into a block diagram yields Fig. 6.10(b), in which certain characteristic features appear including a predictive (non-physical, non-causal) time-advance component, inverses of the individual components of the system, and a positive feed-forward of the input signal.

6.4.3 The System-inverting Controller

The basic block diagram of the system-inverting controller (SIC) is given in Fig. 6.11. The block labeled SIG. SYNTH. CTRL. does the work. We need to derive what this block should do (i.e. how to characterize $B(s)$ in terms of the rest of the system and its desired function). To see how the method goes, let us start by imposing the requirement that we want to synthesize a zero-latency version of the undelayed closed loop original system. This takes us a step beyond Smith prediction since we will now reduce the latency to zero. We thus want

$$Y(s) = I(s).$$

For this equation to hold, we have (selectively dropping the Laplace transform variable s):

$$Y = \frac{CGe^{-s\tau}}{1 + CGe^{-s\tau}}(I + B). \tag{6.13}$$

Since

$$Y = I = \frac{CGe^{-s\tau}}{1 + CGe^{-s\tau}}(I + B),$$

$$B = I(1 - \frac{CGe^{-s\tau}}{1 + CGe^{-s\tau}})(\frac{1 + CGe^{-s\tau}}{CGe^{-s\tau}}).$$

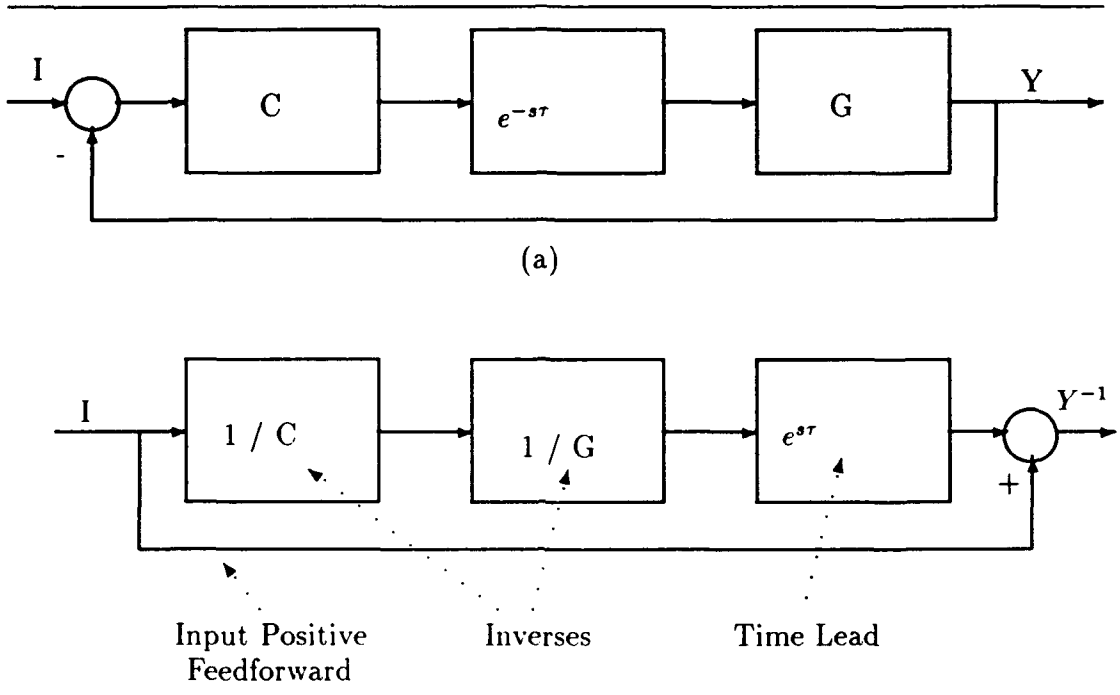


Figure 6.10: A delayed system and its inverse, with inverse characteristics labeled.

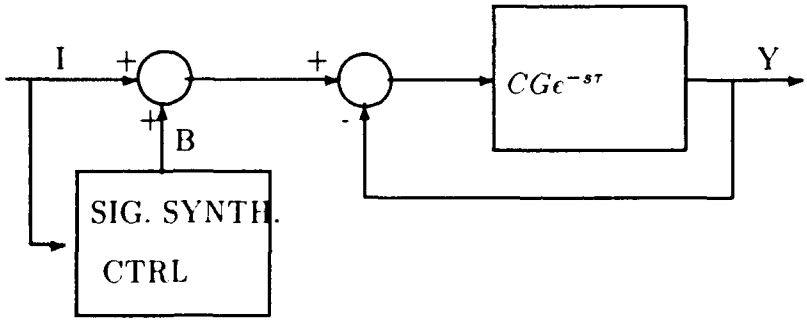


Figure 6.11: The signal synthesis controller.

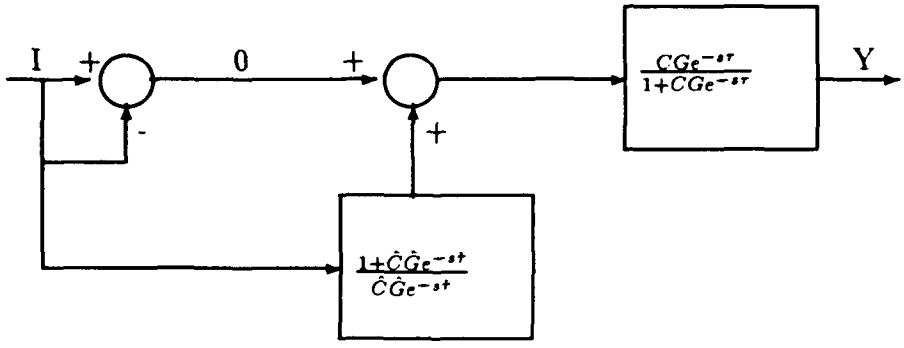


Figure 6.12: The signal synthesis controller for zero latency realization of closed-loop control.

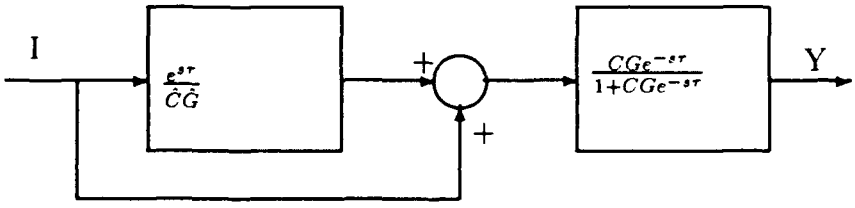


Figure 6.13: Another realization of the signal synthesis controller for zero latency realization of closed-loop control.

Finally,

$$B = I(\frac{1 + CGe^{-s\tau}}{CGe^{-s\tau}} - 1). \tag{6.14}$$

In eq. (6.14), the first term inside the parentheses is the inverse of the system, that is of the delayed plant and feedback controller. The second term (-1) simply cancels the input! In this form the SIC block diagram appears as in Fig. 6.12.

The transfer function of the SIC producing $B(s)$ can be rewritten (multiply the first term in parentheses by $e^{s\tau}$, divide out, simplify)

$$\frac{e^{s\tau}}{CG} + 1,$$

which we saw in Section 6.4.2. Written like this, the SIC block diagram is as shown in Fig. 6.13.

A last rewriting of this particular system shows that it can be realized without any explicit inversions of the components. Thus it can be implemented with the same com-

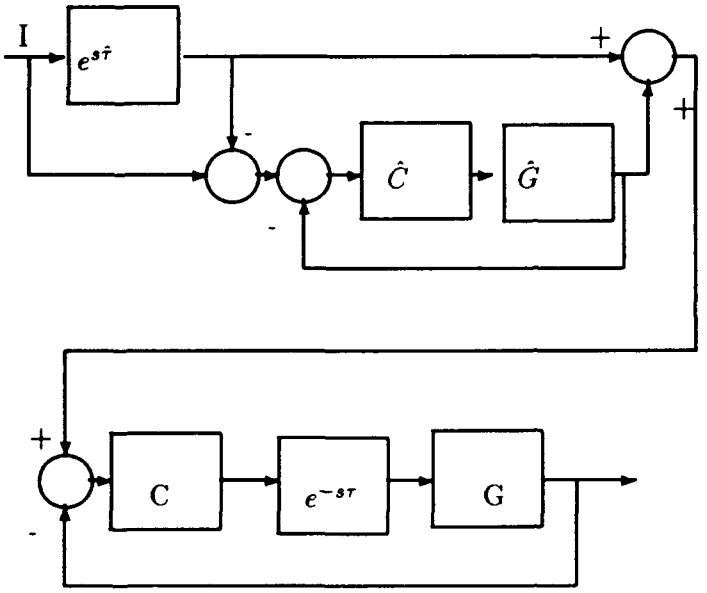


Figure 6.14: Block diagram manipulation yields this realization of the system of Fig. 6.12 or 6.13.

ponents that were used to make the Smith controller, as long as the signal is predictable (Fig. 6.14).

This basic technique of writing down the desired transfer function of the system and then solving for B is clearly quite general. For instance, to synthesize a new delay-free system with controller D and plant H, simply invert the existing plant, controller, and delay and substitute the desired ones as follows.

We want the system response to be

$$\frac{Y}{I} = \frac{DH}{1+DH}$$

Using the block diagram defining the SIC (Fig. 6.11) we equate what the system will do to what we want it to do:

$$(B+I)(\frac{CG\epsilon^{-s\tau}}{1+CG\epsilon^{-s\tau}}) = I(\frac{DH}{1+DH})$$

Solving for B we get

$$B = I\left[\frac{DH}{1+DH}\frac{1+CG\epsilon^{-s\tau}}{CG\epsilon^{-s\tau}} - 1\right]. \tag{6.15}$$

Here again we see the input being Subtracted and the inverse of the unwanted system composed with the desired system (Fig. 6.15).

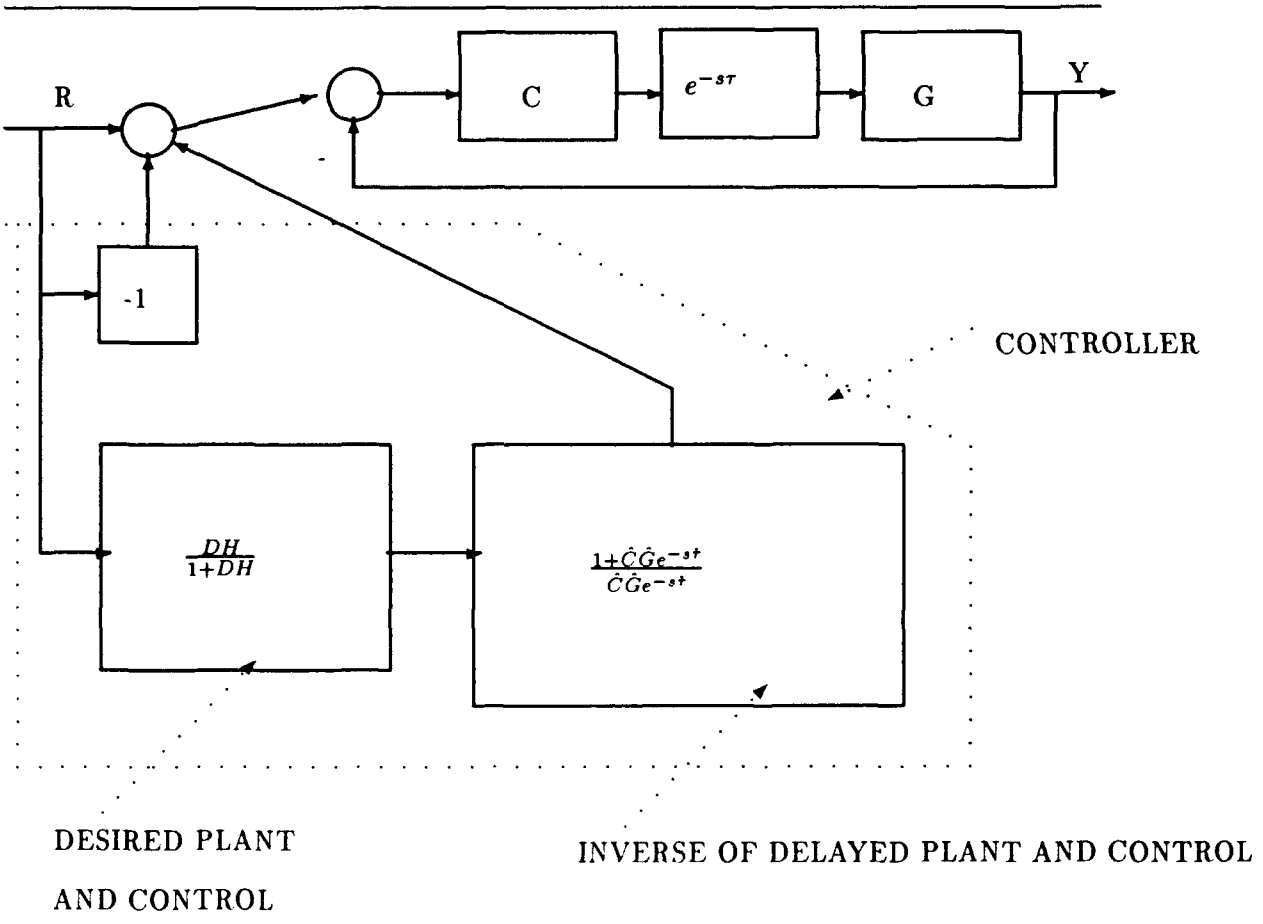


Figure 6.15: System inversion allows any controller to be substituted for the original.

6.5 Enhancing Smith Prediction with Input Prediction

As we have seen, pure Smith prediction removes the delay-induced poles from the controller but leaves the latency. To remove the latency in the Smith controller, it only remains to predict the input, as in the SIC.

A step in this direction is taken in [Brown, 1989; Brown, 1990a], in which an explicit kinematic simulation used to predict the system state, and an optimal (i.e. variance-minimizing) filter is used to smooth the position and velocity estimation of the world object. Extended Kalman filters, (linear) Kalman filters, and time-invariant filters were investigated as input estimators. However, in this work the filters were not used in their predictive capacity (Fig. 6.16).

Our goal in the current work is to predict the signal for the purpose of compensating for delays. The predictor will be placed early in the system, before the Smith predictor. The pure predictor of the SIC is an "oracle" usually dismissed as unphysical or noncausal. In this section we introduce the idea of a predictive filter to supply the necessary $e^{s\hat{\tau}}$ to remove the Smith predictor's latency. It is standard practice with optimal filtering to use statistical techniques to see if the current dynamic model fits the data, and if not to substitute another model [Brown *et al.*, 1989; Bar-Shalom and Fortmann, 1988]. This "variable dimension" approach is the predictive filtering equivalent of the signal synthesis adaptive control scheme [Bahill and McDonald, 1981].

One example of a system incorporating Smith prediction and signal estimation is shown in Fig. 6.16.

6.5.1 The α-β filter

Linear dynamical systems with *time-invariant* coefficients in their state transition and measurement equations lead to simpler optimal estimation techniques than are needed for the time-varying case. The state estimation covariance and filter gain matrices achieve steady-state values that can often be computed in advance. Two common time-invariant systems are constant-velocity and constant-acceleration systems.

Let us assume a *constant velocity* model: starting with some initial value, the object's velocity in LAB evolves through time by *process noise* of random accelerations, constant during each sampling interval but independent. With no process noise the velocity is constant; process noise can be used to model unknown maneuverings of a non-constant velocity target. The cumulative result of the accelerations can in fact change the object's velocity arbitrarily much, so we model a maneuvering object as one with high process noise. For this work we assume position measurements only are available, subject to measurement noise of constant covariance. Clearly the more that is known *a priori* about the motion the better the predictions will be. Some sensors or techniques can provide retinal or world velocity measurements as well.

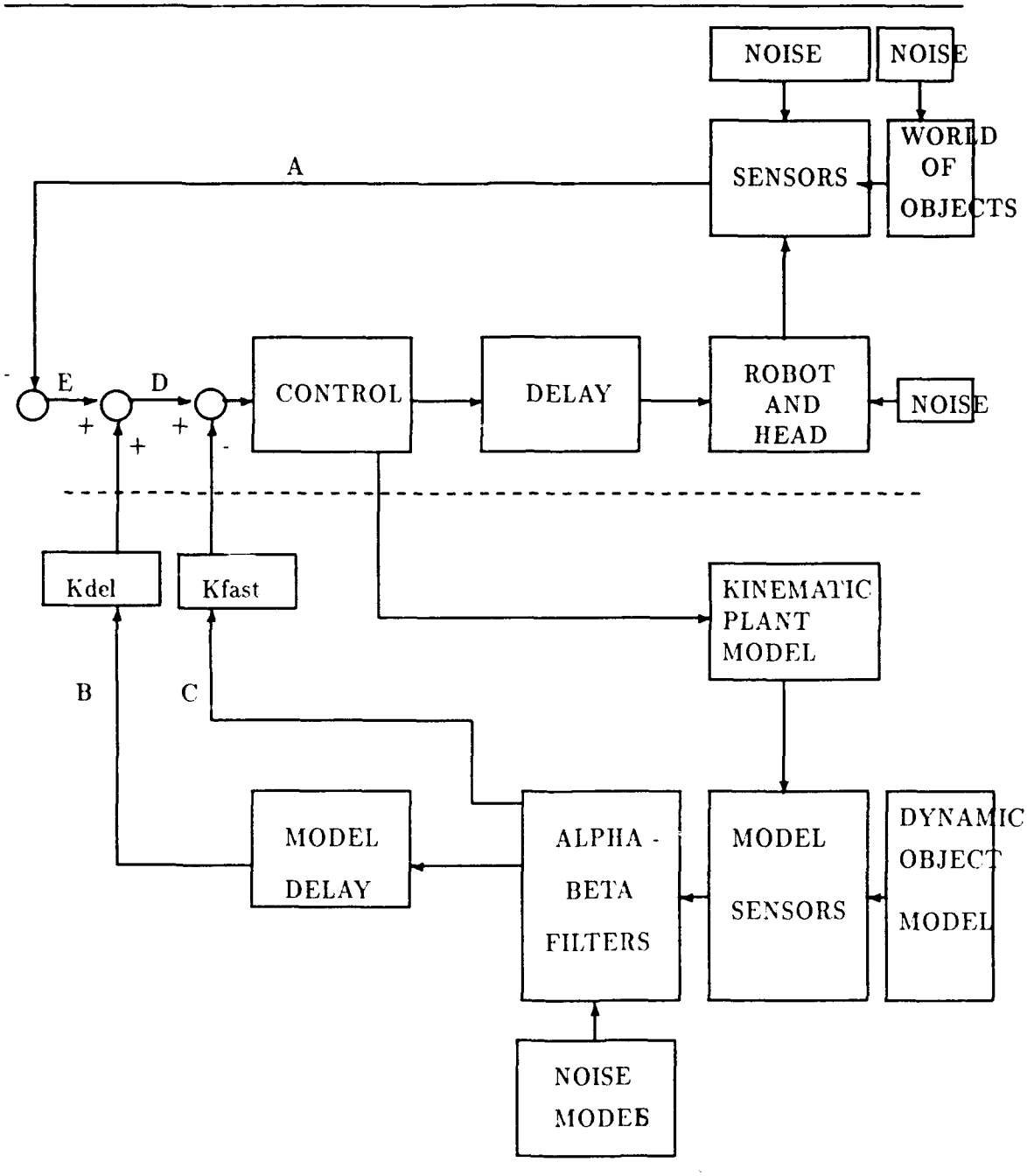


Figure 6.16: Smith prediction with kinematic model, signal smoothing and estimation with α-β filter. This system exhibits latency.

Assume the object state (its position and velocity) evolves independently in each of the (X, Y, Z) dimensions. For instance, in the Y dimension, it evolves according to

$$\mathbf{y}(k+1) = \mathbf{F_y}\mathbf{y}(k) + \mathbf{v}(k), \tag{6.16}$$

where

$$\mathbf{F_y} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{6.17}$$

for sampling interval Δt, error vector $\mathbf{v}(k)$, and $\mathbf{y} = [Y, \dot{Y}]^T$. The equations for the other two spatial dimensions are similar, and in fact have identical $\mathbf{F}$ matrices. Thus for the complete object state $\mathbf{x} = [X, \dot{X}, Y, \dot{Y}, Z, \dot{Z}]^T$, $\mathbf{F}$ is a (6×6) block-diagonal matrix whose blocks are identical to $\mathbf{F_y}$. The error vector $\mathbf{v}(k)$ can be described with a simple covariance structure: $E(\mathbf{v}(k)\mathbf{v}^T(j)) = \mathbf{Q}\delta_{kj}$.

The α-β filter for state prediction has the form

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \begin{bmatrix} \alpha \\ \beta'\Delta t \end{bmatrix} [\mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)], \tag{6.18}$$

where $\hat{\mathbf{x}}(k+1|k+1)$ is an updated estimate of $\mathbf{x}$ given $\mathbf{z}(k+1)$, the measurement at time $k+1$. Here we assume that $\mathbf{z}(k+1)$ consists of the three state components (X, Y, Z) (but not $(\dot{X}, \dot{Y}, \dot{Z})$). The state estimate is a weighted sum of a state $\hat{\mathbf{x}}(k+1|k)$ *predicted* from the last estimate to be $\mathbf{F}\hat{\mathbf{x}}(k|k)$ and the *innovation*, or difference between a predicted measurement and the actual measurement. The predicted measurement $\hat{\mathbf{z}}(k+1|k)$ is produced by applying (here a trivial) measurement function to the predicted state.

The α-β filter is a special case of the Kalman filter. For our assumptions, the optimal values of α and β can be derived (see [Bar-Shalom and Fortmann, 1988], for example) and depend only on the ratio of the process noise standard deviation and the measurement noise standard deviation. This ratio is called the object's *maneuvering index* λ, and with the piecewise constant process noise we assume,

$$\alpha = -\frac{\lambda^2 + 8\lambda - (\lambda+4)\sqrt{\lambda^2+8\lambda}}{8} \tag{6.19}$$

and

$$\beta = \frac{\lambda^2 + 4\lambda - \lambda\sqrt{\lambda^2+8\lambda}}{4}. \tag{6.20}$$

The state estimation covariances can be found in closed form as well, and are simple functions of α, β, and the measurement noise standard deviation.

6.5.2 The α-β-γ Filter

The α-β-γ filter is like the α-β filter only based on a uniform acceleration assumption. Thus it makes a quadratic prediction instead of a linear one. Broadly, it tends to be more

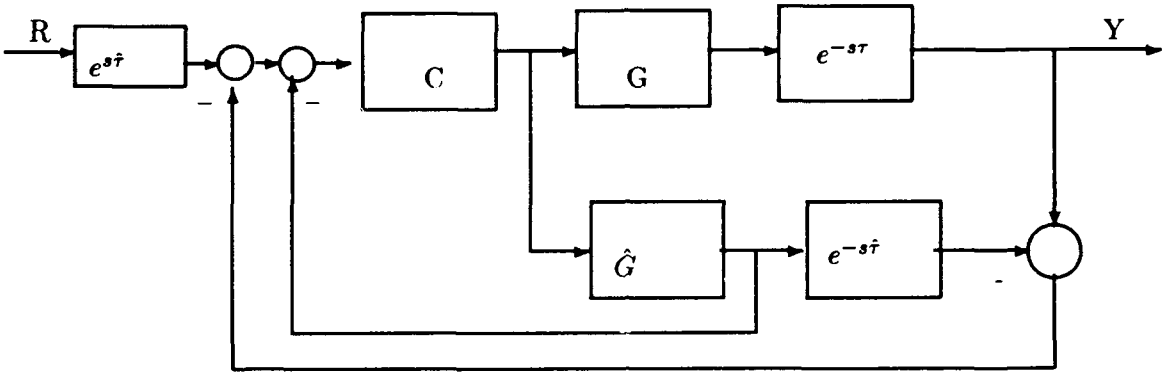


Figure 6.17: Smith prediction control with input prediction to overcome latency.

sensitive to noise but better able to predict smoothly varying velocities. Its equation is the following.

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \begin{bmatrix} \alpha \\ \beta/\Delta t \\ \gamma/\Delta t^2 \end{bmatrix} [\mathbf{z}(k+1) - \hat{\mathbf{z}}(k+1|k)], \tag{6.21}$$

With the maneuvering index λ defined as before, the optimal α and β for the case that the target experiences random small changes in acceleration (random jerks) are the same as before and the optimal $\gamma = \beta^2/\alpha$.

6.5.3 Input Prediction

The smoothing effect of the estimation process improved the performance of the system in noise [Brown, 1989; Brown, 1990a]. The predictive version of the filter was not used to try to remove latency, however. The classical Smith predictor can be enhanced with some form of input prediction in order to ameliorate the latency built into Smith's principle.

The resulting block diagram is shown in Fig. 6.17.

The predictive element $e^{s\hat{\tau}}$ may be realized nonphysically by a "prescience filter" or oracle, by actual prior knowledge of the signal, or (approximately) by a predictive filter. By the same token, the predictive element in the signal synthesis controller shown in Fig. 6.14 may be realized by any of these mechanisms. In later sections we shall do some experimental work to assess the relative efficacy of the resulting controllers.

6.6 Experiments

6.6.1 Simulator and Example System

To explore the effects of noise and systematic error on the control schemes described here we wrote a control system simulator. The user specifies the characteristics of blocks in a system diagram, as well as sample rates, input function, and length of run. The system can be continuous, discrete, or hybrid. A continuous system (or block) is specified by its differential equation (a variable step Runge-Kutta method is used for solutions). Discrete components include integrators (with various options of anti-windup, integration time, saturating or not, and leak), sample and hold, differentiators, gains, inputs (impulse, step, ramp, cosine, and squarewave), additive noise (uniform or Gaussian), summers, and output blocks. Each block can induce a delay or achieve a time advance. Continuous blocks in use are an integrator, a leaky integrator, a cascaded integrator and leaky integrator, and a spring, mass, damper system. There are also predictors of several varieties. Currently, there are nonphysical "prescient" oracles that look into the future of their input as well as α-β and α-β-γ filters—the predictors have a provision to produce noisy predictions in which the time to look ahead is noisy. New blocks are easy to add and the functionality of blocks is easy to change.

Using this tool, the systems illustrated in Figs. 6.6, 6.14, and 6.17 were simulated. Several examples were tried, to establish for instance that in the ideal case of perfect modeling both the Smith and SIC controllers achieved the same effect as the undelayed controller. For this report we ran experiments in which noisy predictions (rather, accurate predictions but for noisy values of the future time) and noisy plant models (simulated simply by adding noise into the system). Since this experimental approach is really no substitute for an analytical understanding of what is happening, we limited our experiment to a typical case that has enough complexity to produce interesting behavior. We chose a PID discrete controller applied to a continuous spring, mass, damper system.

The action of this controller acting alone on the system is illustrated, first for a step input and then for a "sinusoidal" input (actually $u(t)(1-\cos(t\omega 2\pi))$, with $u(t)$ the unit step). Figs. 6.18 and 6.19 show the response of the system and the necessary control signal to a step input, using a good setting of P,I, and D (0.4, 0.4, and 0.8). Fig. 6.20 shows the step response of the system for a bad setting of P,I, and D (0.9, 0.4, 0.1). Fig. 6.21 shows the sinusoidal input. Figs. 6.22 and 6.23 show the sinusoidal response of the system for the good and bad settings, respectively.

6.6.2 The PID-MSD System in Noise

The goal of these experiments is to compare the performance of three approaches overcoming delay. Each approach depends on a system model, and two approaches use

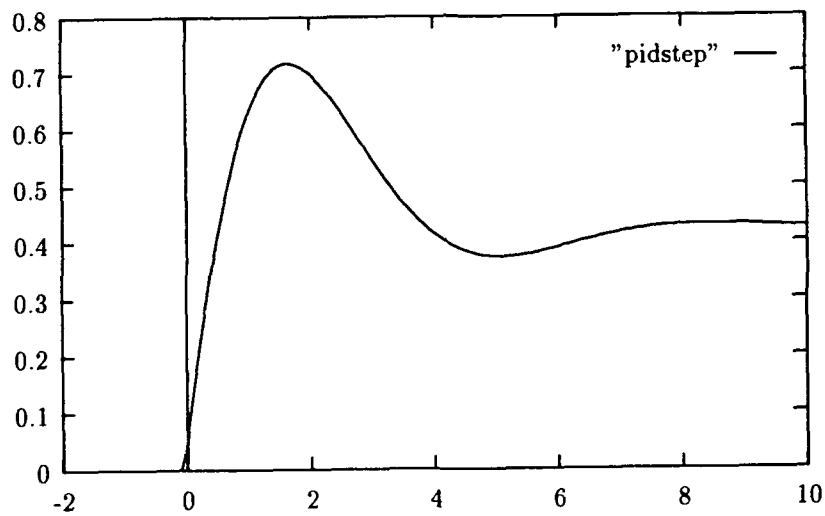


Figure 6.18: Output of mass, spring, and dashpot system to step input using (P,I,D) = (0.4, 0.4, 0.8): the result is underdamped response.

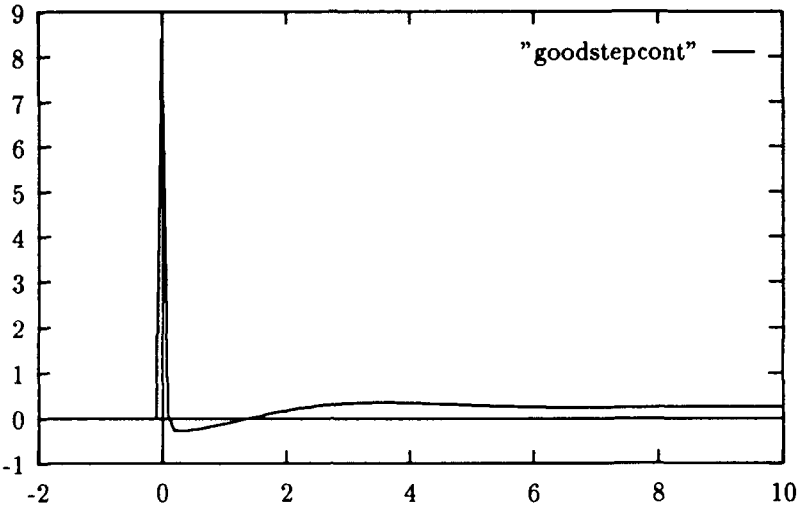


Figure 6.19: Control signal producing the output of Fig. 6.18.

models of the input. In the experiments to be described, we concentrate on stochastic errors rather than systematic errors. In particular we consider additive error in the output of the controller that we intend to model the effects of parametric mis-estimation of the system. For temporal sensitivity, we consider stochastic mis-estimation of the correct delay.

To set a baseline for these experiments, we first consider the performance of the pure PID controller for the MSD (mass, spring, dashpot) system described in the previous section. The system diagram is that of Fig. 6.24.

Clearly there are many parameters that can be interestingly manipulated in experiments on control systems, and many measurements that can characterize the effects. First, to get some idea of the effects of delay in the feedback loop, consider the system response in the case of a delay of 0.4 compared with no delay (besides the inherent one-step discrete controller delay) (Fig. 6.25).

For this set of experiments, the sampling rate of the discrete controller is a constant 0.1 second. The noise added to the controller output is normally distributed, with a mean of 0.0 and a standard deviation of 0.2. The stochastic time perturbation is calculated as the integral number of ticks that results from rounding a normally distributed random variable with mean 0.0 and standard deviation of 0.1. The control input is $u(t)(1 - \cos(2\pi t\omega))$. The performance of the perturbed systems can be compared to

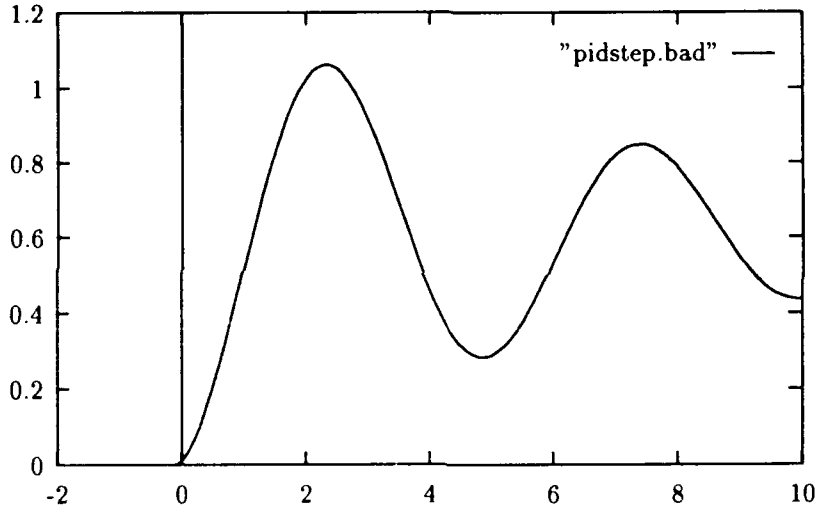


Figure 6.20: Output of mass, spring, and dashpot system to step input using (P,I,D) = (0.9, 0.4, 0.1): the result is a quite underdamped response.

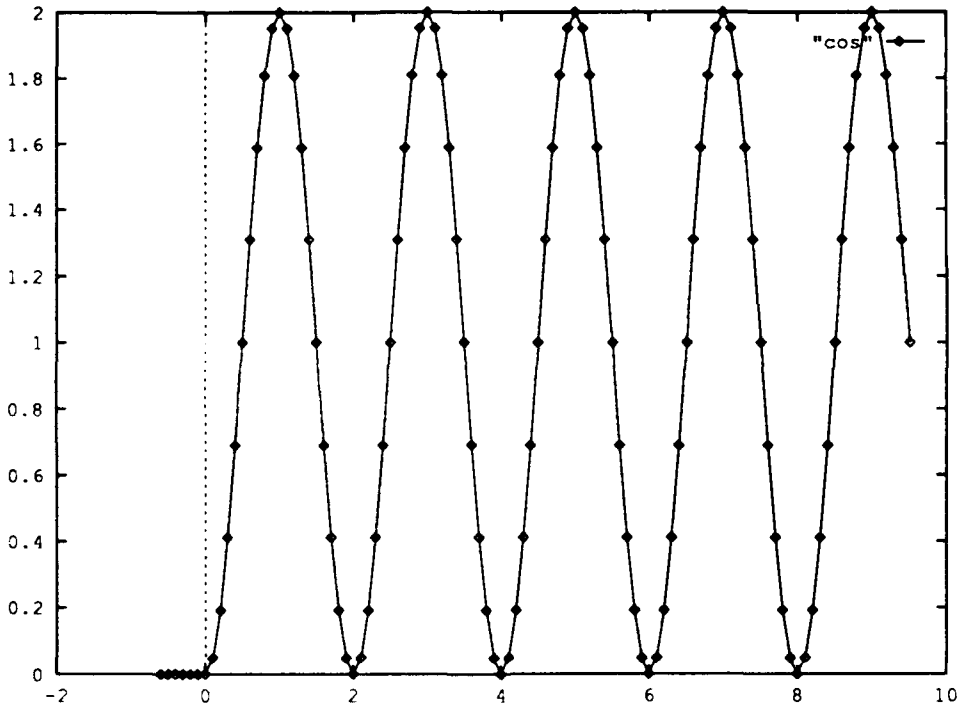


Figure 6.21: Sinusoidal reference input $u(t)(1 - \cos(t\omega 2\pi))$, with $u(t)$ the unit step, $\omega = 0.5$. This input is used throughout rest of work.

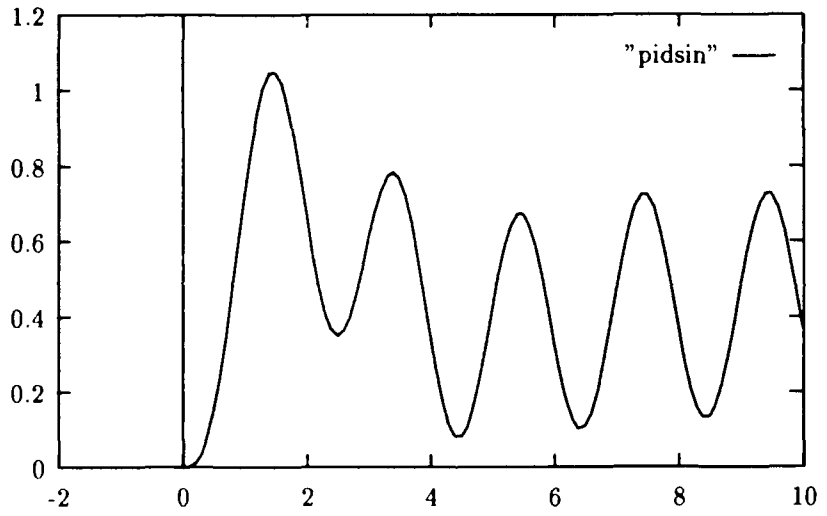


Figure 6.22: Output of mass, spring, and dashpot system to $u(t)(1 - \cos(2\pi\omega t))$ input for $\omega = 0.5$ using (P,I,D) = (0.4, 0.4, 0.8).

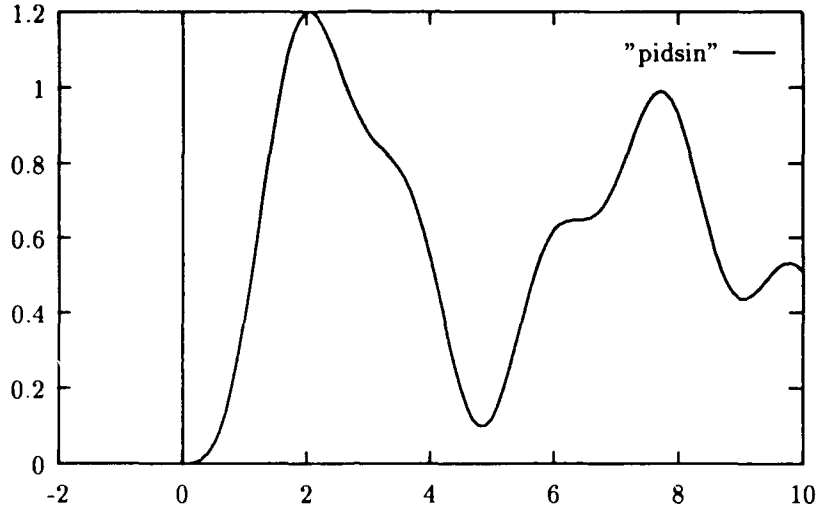


Figure 6.23: Output of mass, spring, and dashpot system to sinusoidal input using (P,I,D) = (0.9, 0.4, 0.1).

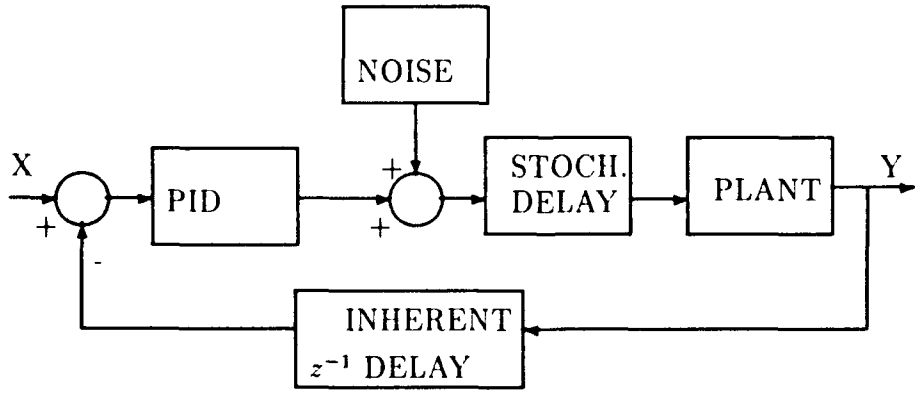


Figure 6.24: Perturbations in the PID controller. The controller output has additive noise and there is mis-estimation of the "zero" delay, implemented by stochastic non-zero delay or advance. In fact the delay is one time step, since this is actually a discrete controller—we have indicated that inherent delay with the z^{-1} box.

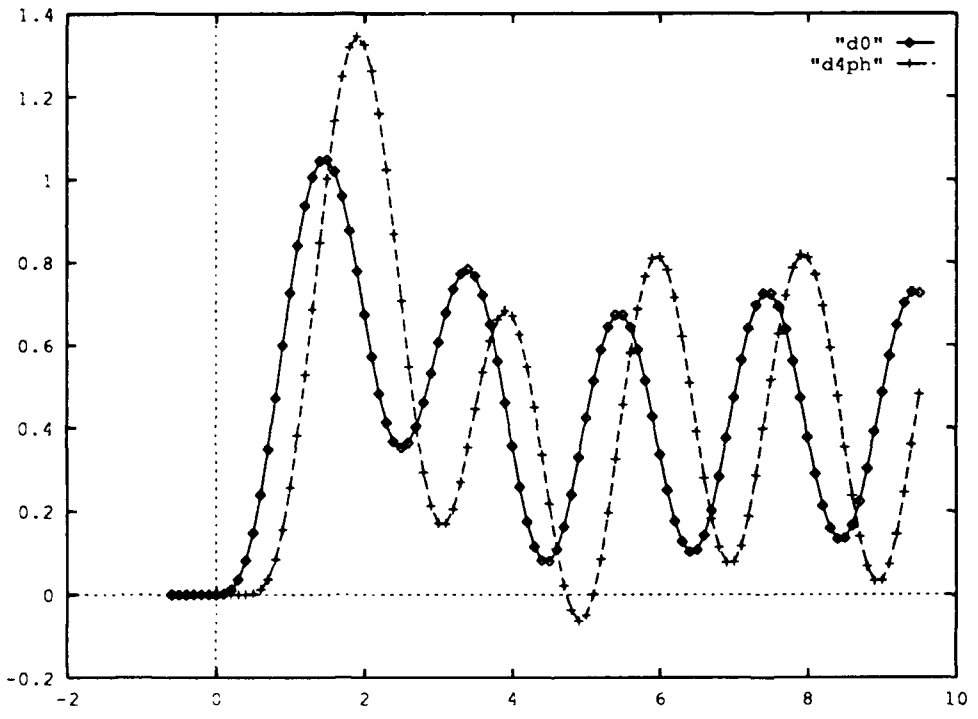


Figure 6.25: Zero delay case compared to a constant delay of 0.4 in the feedback loop of the system. The delay causes overshoot and a phase lag.

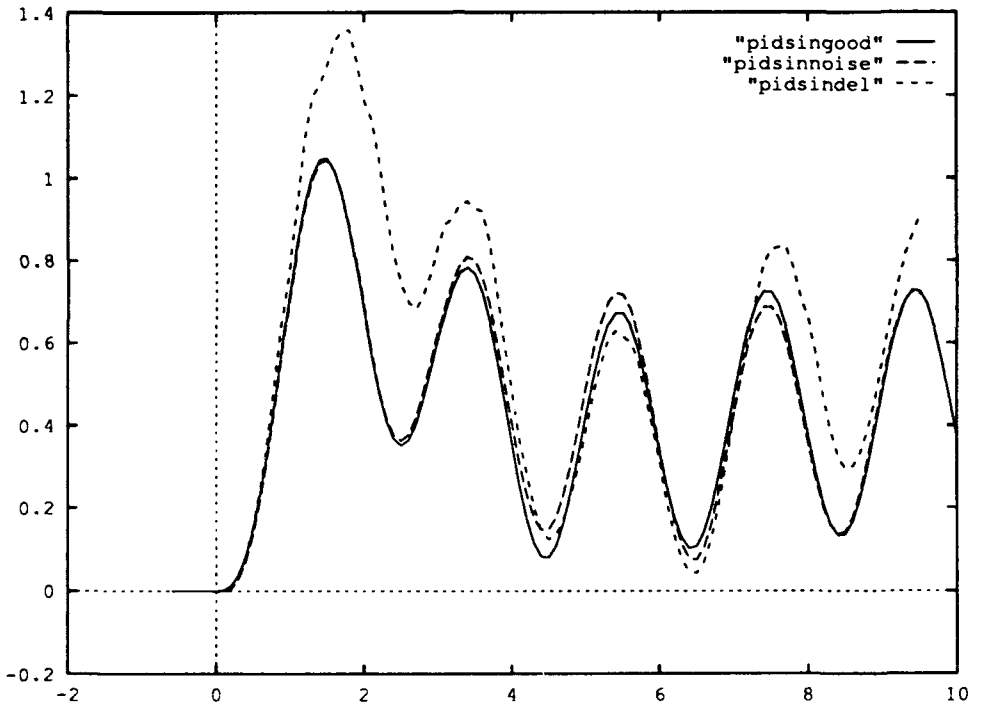


Figure 6.26: The conditions of Fig. 6.22 shown with the results of noise added to controller output and with noisy signal delay and advance.

that of the basic PID controller and to noisy versions of the PID controller by the metrics of phase lag and gain (Bode plots) if the frequency of the input sinusoid is varied. Within a frequency, such metrics as overshoot and the sum of absolute differences of the relevant system state (position) suggest themselves.

Fig. 6.26 shows the system state (position of the mass) under the noisy conditions described above compared with the noiseless case.

6.6.3 Delay and Simple Prediction

The PID-MSD system is relatively resistant to delay, but its effects are significant. One idea is to insert an estimator-predictor in the loop to cancel the effects of delay. Fig. 6.27 shows the system.

Here the delay is compensated by an α-β filter that estimates and predicts the output of the controller. Using an α-β-γ filter has advantages and disadvantages: the advantages are less overshoot, the disadvantage is a longer initialization time (three

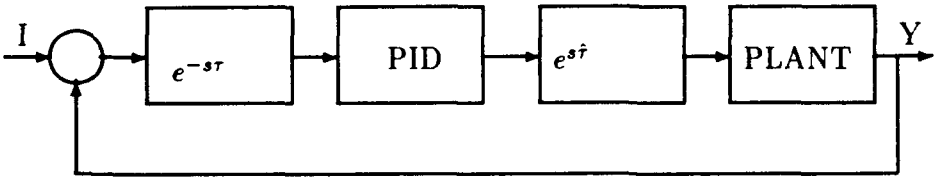


Figure 6.27: The PID MSD system with delay and predictor.

readings instead of two) that can make a significant difference in performance when control is active. Fig. 6.28 shows the zero delay case, the case where delay is 0.3 sec. and no prediction is used, and the case of 0.3s delay and the prediction is for 0.3 sec. ahead. In this system the prediction eliminates the phase lag of the system induced by the delay, but the overshoot of the output in the delayed and prediction-compensated systems is similar.

6.6.4 Open Loop Control

It is difficult to compare the performance of the open-loop control approach to the other methods because this approach calls for an entirely new system design: the open loop controller in general looks nothing like the closed loop controller. What we show here is simply the degradation of the performance of the controller whose loop has been opened by positive feedback under the two noise conditions of this section: the standard sinusoidal input, additive noise of 0.2 in the positive feedback path, and normally distributed stochastic variation of the delay from its mean of 0.0 with a standard deviation of 0.1. For comparison we simply give the noiseless performance, which should be (and in fact is, in our simulation) the same as the output of the PID controlled MSD system with negative feedback disabled. This output is not related to the closed-loop PID-MSD system performance, of course. Fig. 6.29 gives the system configuration and Fig. 6.30 the three responses.

As mentioned in Section 6.2, the process of opening the control loop by positive feedback does not remove the delay, or latency, from the output. As with all our schemes, the solution is simply to cascade an input predictor with the system.

Since the open- and closed-loop systems are not directly comparable, we simply invoke the argument of Section 6.1 concerning the decreased sensitivity of feedback systems to parametric variation. If we can afford to give up the advantages of feedback systems, then an open-loop controller can of course be used, and then delay will only cause a latency in output that can be more or less compensated by predictive filtering somewhere in the system.

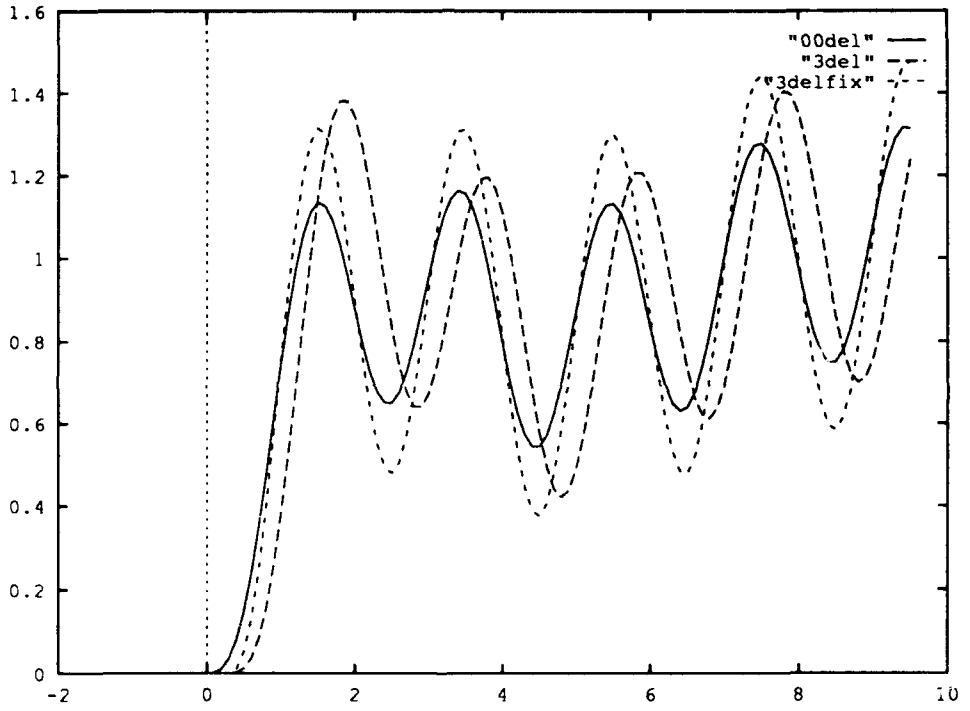


Figure 6.28: The undelayed response (solid line) is perturbed by a delay of 0.3s in the control loop (dashed line). Using the estimator to predict the control signal reduces the phase lag but overshoot remains (dotted line).

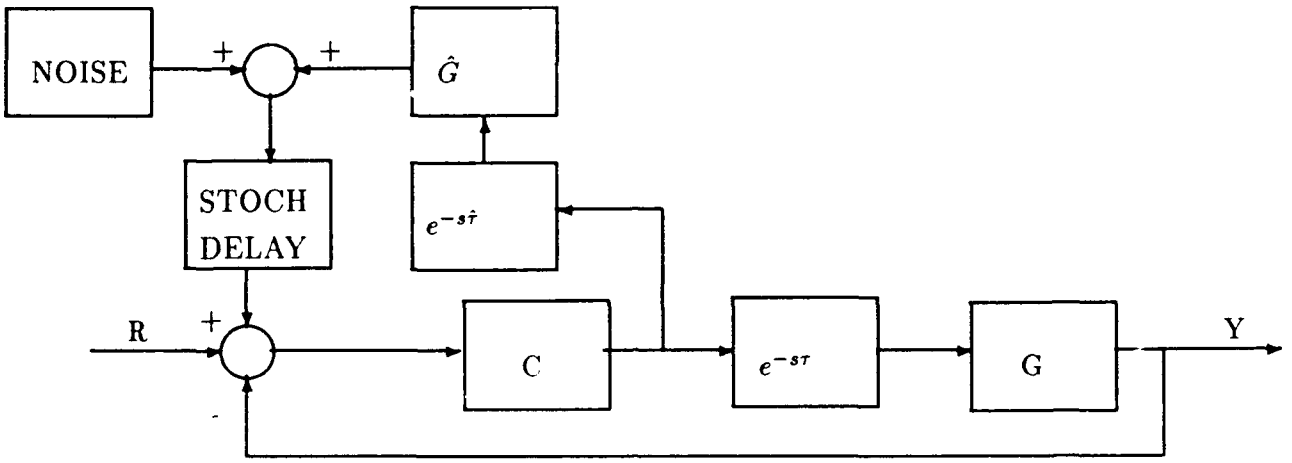


Figure 6.29: The positive feedback loop-opening system, showing where noise and stochastic delay were injected.

6.6.5 Smith and Signal Prediction

The block diagram of the system used in this section is given in Fig. 6.31. As before, the controller is the PID controller of earlier sections and the system is the same mass, spring, damper system.

This system diagram is interesting because it brings up the following question. Suppose that mis-modeling the plant actually is modeled as additive noise emanating from the plant model. In the case of zero modeled delay, this noise is both added to and subtracted from the error signal, and thus cancels itself completely. This argument extends to any behavior of the plant model at all, of course: it is all subtracted out in the zero-delay case.

Consider two cases, the zero delay case with noises as we have had previously, and a case in which the plant delay is 0.2, which is accurately reflected in the model delay except when that modeled delay is perturbed by our standard stochastic perturbations. In this case the effect of the noise is doubled, since a shifted version is subtracted from itself at the input. We obtain the graphs of Figs. 6.32 and 6.33. As expected, the Smith predictor avoids the deleterious effects of systematic delay (Fig. 6.25) on the system output, but retains a delay or latency in its output equal to that of the system delay.

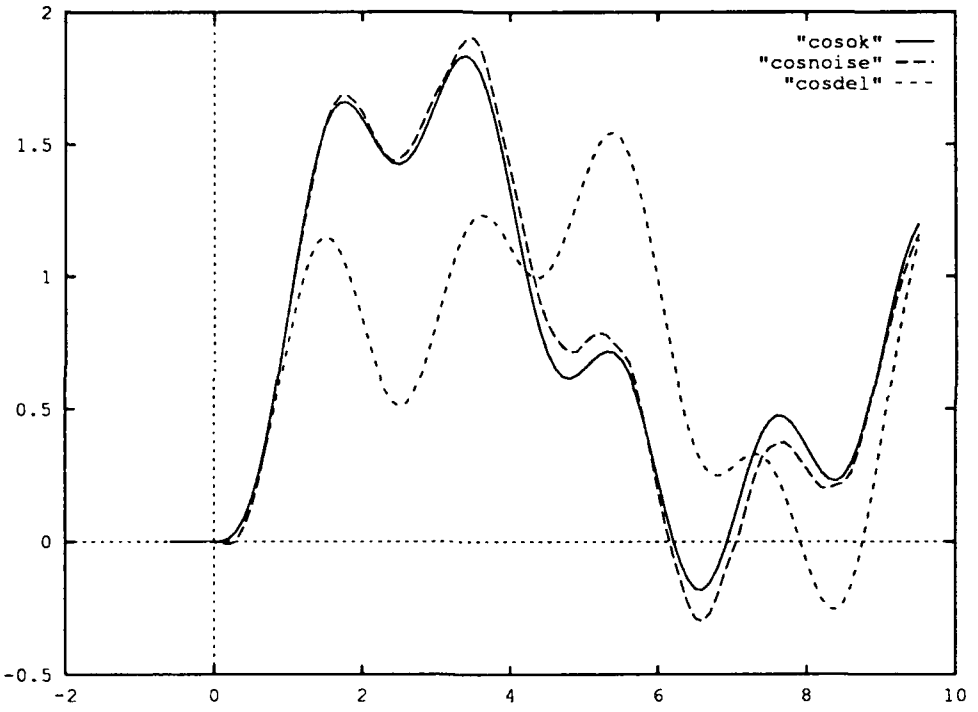


Figure 6.30: The open loop PID-MSD system output for $u(t)(1 - cos(2\pi\omega t))$ input. Ideal output is shown along with output with G(0.0, 0.2) noise in the positive feedback loop and G(0.0, 0.1) stochastic delay in the positive feedback loop.

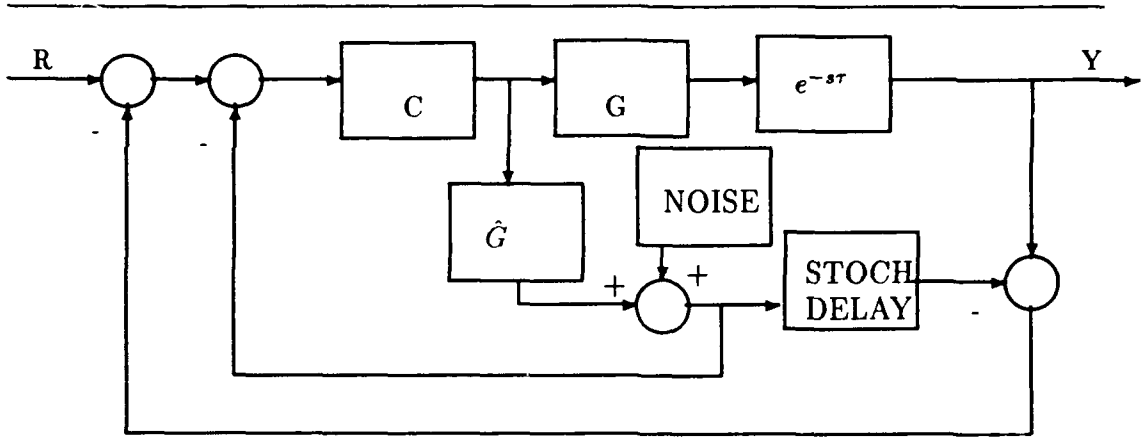


Figure 6.31: The Smith predictor with noise added to simulate plant and delay mismodeling. The STOCH. DELAY box represents a stochastic variation about the modeled delay, which itself is ideally the same as the true plant delay.

6.6.6 Signal-Inverting Control

Although this controller is inspired by, and in its most general form (Fig. 6.15) demands the inversion of the following system, here we use the rewriting that merely calls for a duplication of the controller and a model of the plant (along with a nonphysical signal advancer). Fig. 6.34 shows the system used in the simulations. Fig. 6.35 shows the results with the standard noise parameters of this section.

6.7 Input Prediction

In the experiments in Section 6.6, the Smith and SIC controllers needed to be able to predict the input in order to overcome all the effects of delay. So far we have assumed an oracle that actually knows the future. This is an impractical general solution (although in some cases one can make an argument that very precise expectations are downloaded into the controller—this is a central idea in the signal synthesis adaptive controller [Bahill and Harvey, 1986; McDonald and Bahill, 1983].

In this section we examine the effectiveness of the α-β and α-β-γ filters for input prediction. Recall that the former makes linear predictions based on past input and its inherent beliefs about the reliability of its sensors versus the predictability of the target. The latter makes quadratic predictions.

As an example, Fig. 6.36 shows our standard sinusoidal input $u(t)(1 - 2\pi\omega\cos(t))$ and the predicted value for 0.2 seconds into the future yielded by the α-β and α-β-γ

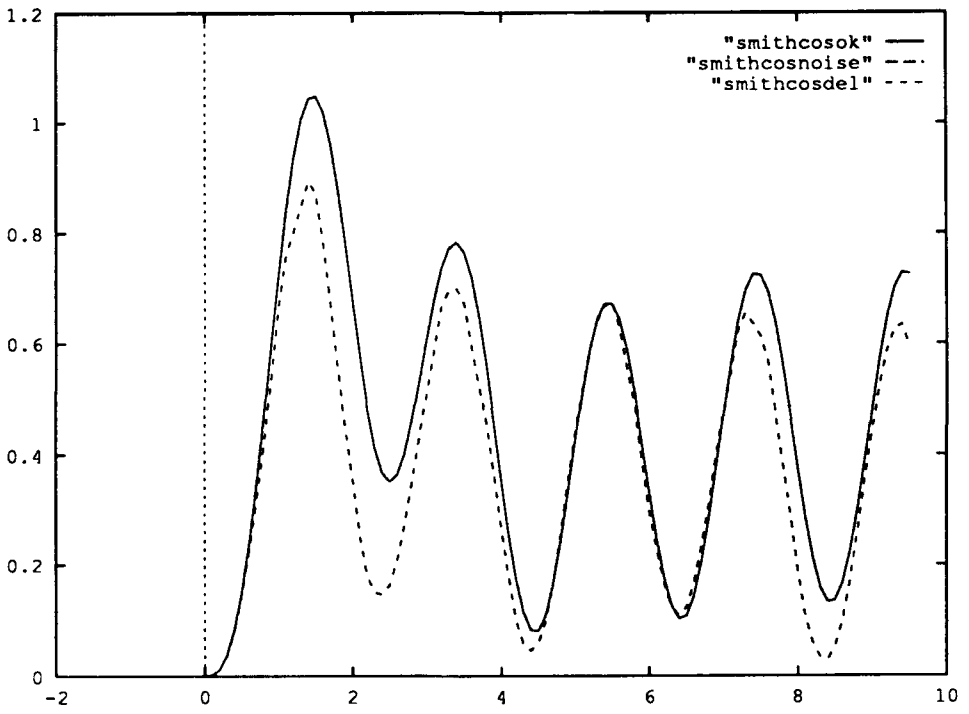


Figure 6.32: The Smith controller with zero modeled delay. Ideal output is shown (solid line) along with output with G(0.0, 0.2) noise in the feedback loop (long dashes) and G(0.0, 0.1) stochastic delay in the feedback loop (short dashes). The controller cancels the noise in this case (see text), so the long dashed line is the same as the solid one.

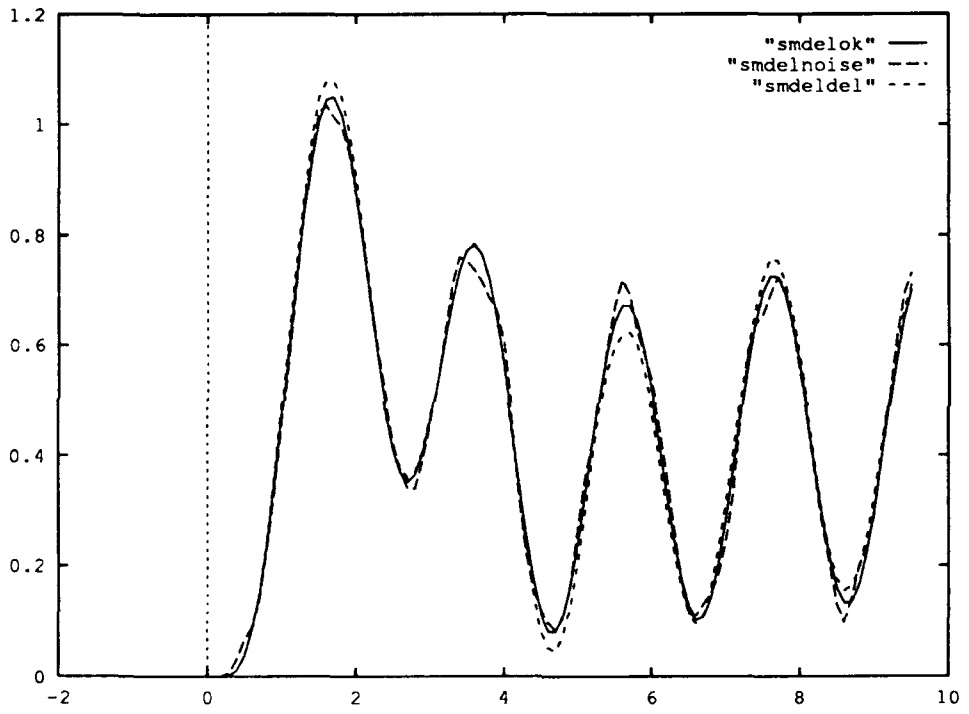


Figure 6.33: As in the previous figure, only with 0.2 system delay. Compared with the last figure, note the similarly-shaped, but delayed, ideal output, which is shown along with output with G(0.0, 0.2) noise in the feedback loop and G(0.0, 0.1) stochastic delay in the feedback loop.

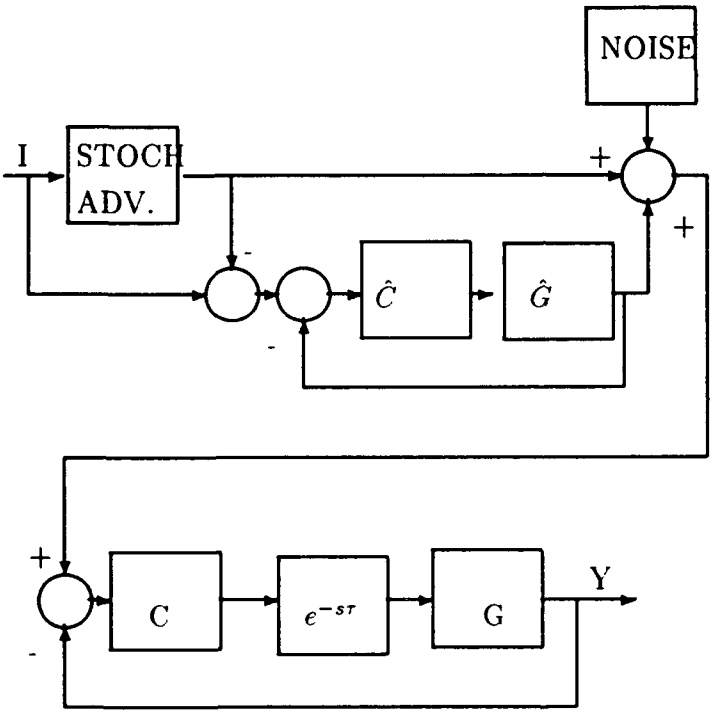


Figure 6.34: The SIC realization, with noise added to simulate plant and delay mismodeling. The STOCH. ADV box represents a stochastic variation about the modeled necessary advance, which itself is ideally the same as the true plant delay.

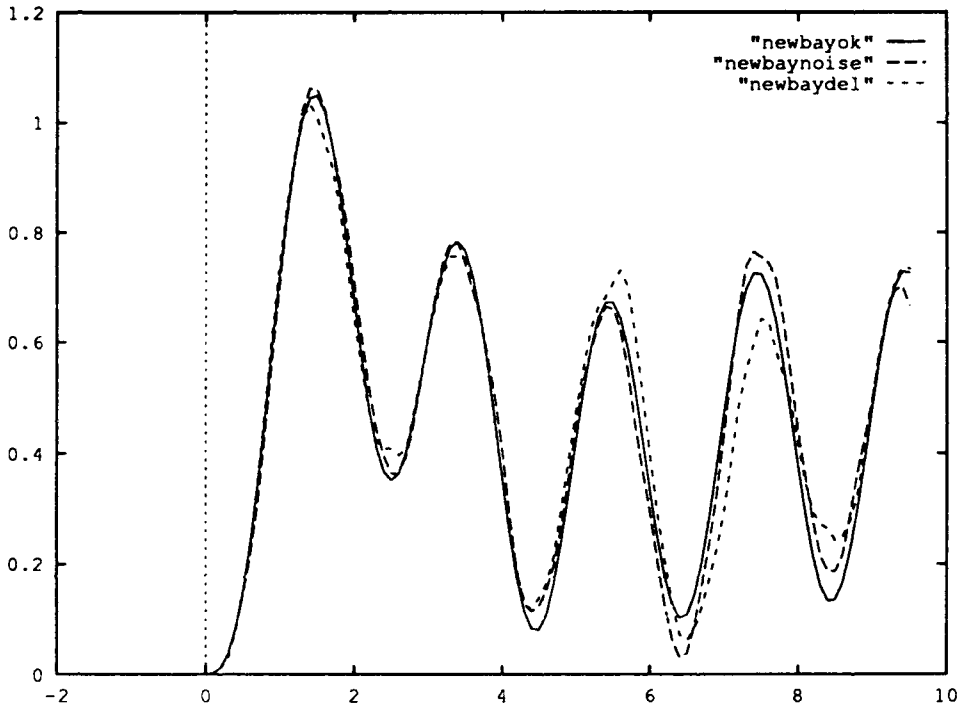


Figure 6.35: The SIC controller dealing with a delay of 0.4. The delay is not present in the output since the (nonphysical) signal advancer presents the controller with its future value, thus removing the delay. The ideal output is shown along with output with $G(0.0, 0.2)$ noise added to the predictive section and $G(0.0, 0.1)$ stochastic advance noise added to the true value.

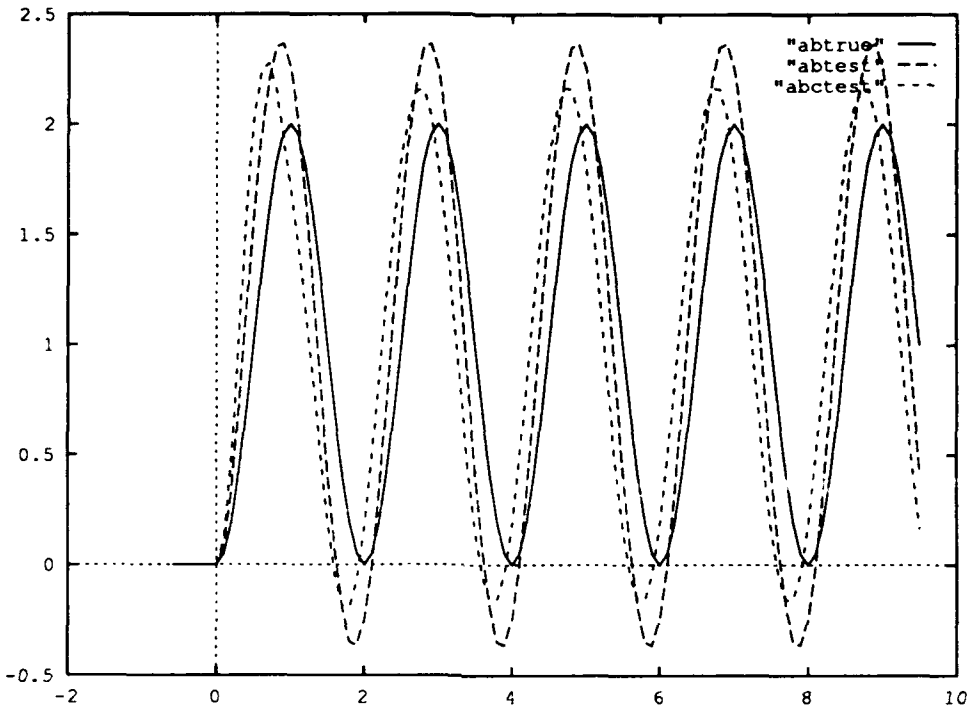


Figure 6.36: α-β and α-β-γ filters predicting sinusoidal input. The quadratic predictor performs better in noiseless data.

filters with $\lambda = 1$ (equal confidence in data and prediction). Neither filter's assumptions about constant velocity or acceleration is met. The α-β-γ filter delivers reasonable predictions with less overshoot. The α-β filter is rather less satisfactory.

The α-β-γ filter may be substituted for the input predictor in the Smith and SIC schemes. For the Smith predictor and a delay of 0.3, the results are shown in Fig. 6.37. The output is close to the ideal output, but shows the characteristic overshoot engendered by the predictive filter.

When the α-β-γ filter is substituted for the input predictor in the SIC scheme, the results are not quite as good. Fig. 6.38 shows the ideal and SIC outputs and the corresponding one from the Smith predictor (identical to the predictive filter output of Fig. 6.37).

A closer approximation to the desired output is actually possible by setting the prediction to 0.4 instead of the correct 0.3. Not only does this change result in less overshoot but in a much closer following of the desired curve between the peaks. This sort of tradeoff is also noticed in the laboratory, where the delay and λ (maneuvering

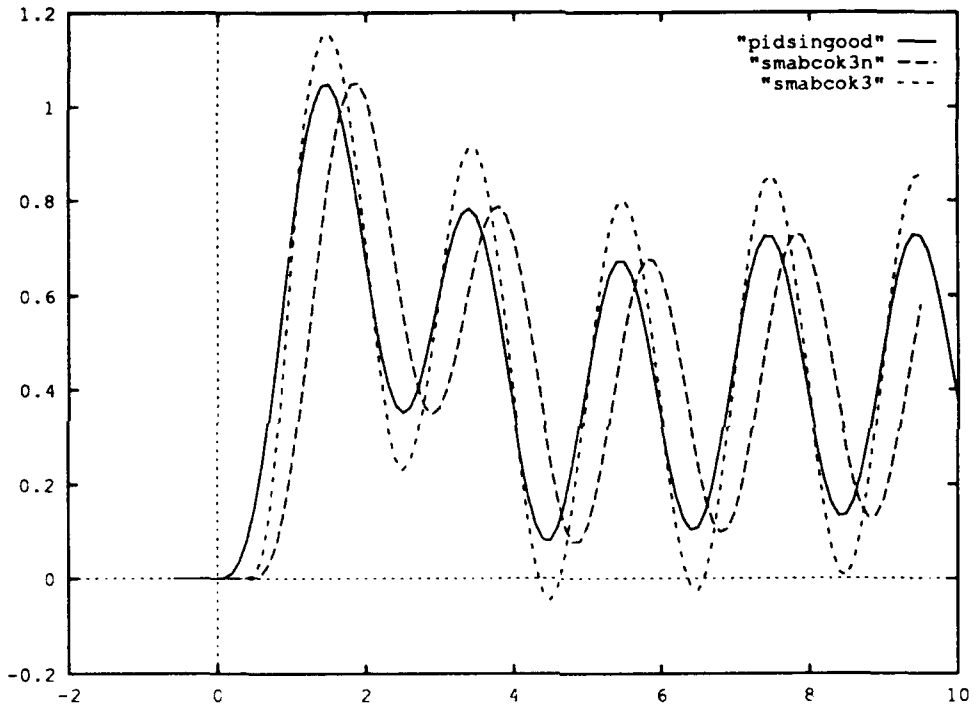


Figure 6.37: The α-β-γ filter predicts the sinusoidal input to the Smith controller (noiseless conditions). Shown are the ideal performance, the output of the Smith predictor (identical to the ideal with a latency equal to the system delay of 0.3), and the output with the predictive filter. The predicted case quickly moves to approximate the ideal except for overshoot.

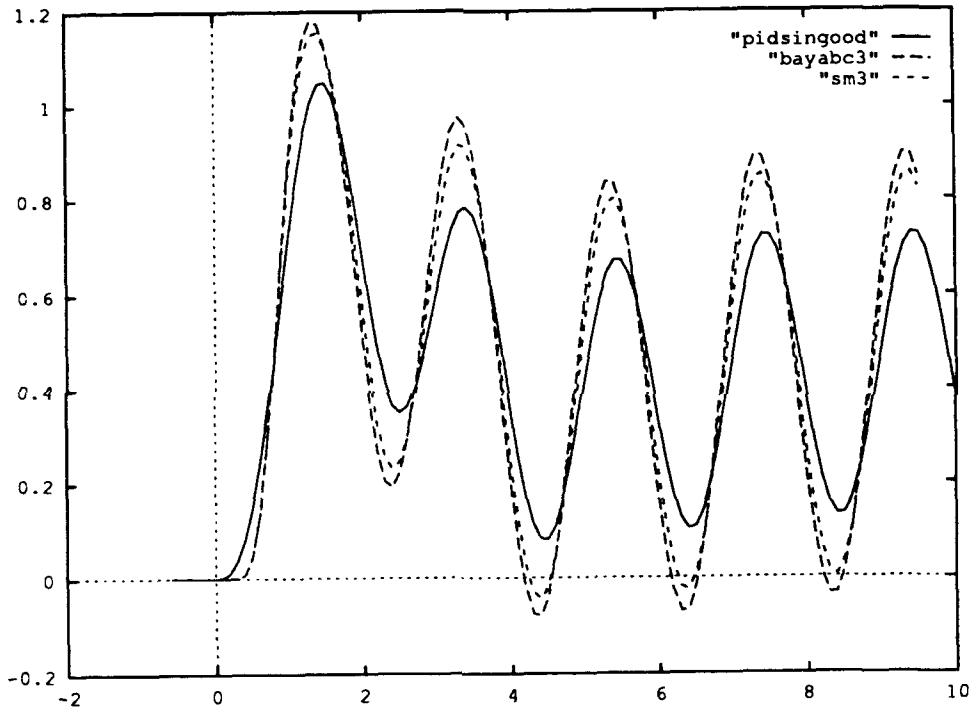


Figure 6.38: The α-β-γ filter predicts the sinusoidal input to the SIC controller (noiseless conditions). Shown are the ideal performance, the output of the SIC controller for the delay of 0.3, and the comparable output of the Smith predictor shown previously.

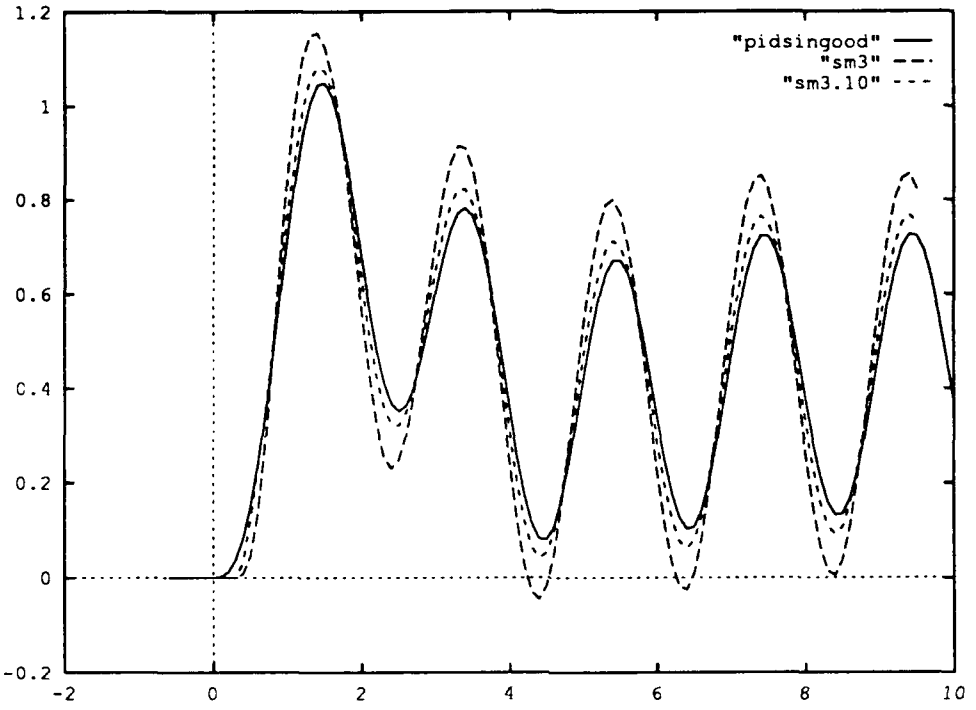


Figure 6.39: Ideal performance and the Smith predictor with its α-β-γ predictive filter predicting the sinusoidal input. The performance of the filter with noiseless input data is much better with $\lambda = 10$ than with $\lambda = 1$.

index) parameters may be traded off to improve performance.

For noiseless data, the maneuvering index (reflecting our faith in our sensors over that of our predictions) may be increased. Substantially improved performance results with $\lambda = 10.0$ (Fig. 6.39).

6.8 Summary

Our goals here were two: understand classical work on control with delay, and apply the most promising techniques in our laboratory to cope with the real delays we experience.

A summary of our evaluation of several approaches to control of time-delay systems is as follows.

1. Ignore the delay or compensate by lowering controller gains: Infeasible in general, this solution might work for some applications because of the relatively short

delays that arise with some of our equipment.

2. Opening the loop through positive feedback: This solution does not appeal, since it loses all the advantages of feedback control and seems sensitive to delay.

3. SIC control: Canceling various parts of the downstream system by inversion is a general and powerful, if practically difficult, idea. This approach requires modeling of the controller as well as of the controlled plant. For a software controller that is not infeasible, but complicates matters somewhat. There are useful cases where its realization does not call for computing the inversion of the controlled plant (generally a bad idea because it would call for differentiators). We have not seen this rewriting of the SIC controller presented elsewhere. The SIC seems slightly more sensitive to noise and delay mismatches than does the next scheme, but probably not enough to worry about.

4. Smith with input prediction control: This classical scheme is elegant and has some practical advantages (not modeling the controller, just the plant).

It is worth mentioning that predictive filters have more uses than predicting the signal: they also estimate the signal. The latter facility is useful when the signal is noisy or drops out. Our $\alpha-\beta$ and α-β-γ filters continue predicting and estimating until a certain amount of time (a filter parameter) elapses during which the filter sees no input data. We expect the estimation aspect of the filters to be as useful as their predictive aspect in practical situations, since the delays confronting us with the MaxVideo equipment and robot head camera controllers is relatively small (on the order of 100 ms or less) compared say to the delays we see with the Puma robot and its VAL controller (on the order of 500 ms).

7 Conclusion

Eye movements are pervasive in the animal kingdom, and they have recently begun to play a prominent role in computer vision as well. Robots, like animals, inhabit a world of moving and stationary objects, and robots and animals themselves move about. Consequently, the ability to hold gaze on an object is crucial to seeing it clearly. Binocular foveal vision requires that the robot hold its foveae simultaneously on the visual target. In addition, motion blur degrades spatial resolution if the image is not prevented from slipping across the retina.

This dissertation examines the problem of holding a robot's gaze on an object while both are moving using *only precategorical* visual processing (*i.e.*, without requiring the ability to recognize the target). The approach is based on the premise that the control of camera movements should be considered an integral part of visual perception. By exploiting constraints that can be maintained by active control of camera movement, simplified visual processing is sufficient to hold the robot's gaze.

A system implemented on the Rochester Robot demonstrates the idea, holding the binocular gaze of a moving robot on an object that moves through a cluttered scene. The vergence and pursuit components of the system cooperate to simplify the visual processing required, as illustrated by Figure 1.2. The vergence system controls the vergence angle between the cameras to minimize the stereo disparity of the foveated target. A fast correlation-based technique estimates the most prominent disparity in foveal stereo images. The pursuit system controls the pan and tilt angles of the cameras to center them on the foveated object that has no stereo disparity. A simple zero-disparity filter locates features that have no stereo disparity. Thus the vergence system maintains zero disparity of the target for pursuit, and pursuit keeps the target foveated for vergence. The system is able to maintain these invariants in the retinal images by its active control of the camera angles.

The chief contributions of this dissertation are:

- That localizing visual attention in 3D space enables simple precategorical processing to suffice for holding gaze on a visual target. It is demonstrated (in Section 4.5) that simple binocular cues are able to distinguish a relatively near target in the

presence of clutter. Other precategorical cues with physical 3D interpretations should prove useful in situations that do not provide binocular cues.

- A demonstration of harnessing the interaction between sensing and control to support sensory-motor behavior, which is also exemplified by the ability of binocular fixation segmentation to pick out the target during gaze holding.
- A demonstration of the symbiotic cooperation of simple subsystems, *e.g.*, pursuit and vergence (in Section 4.6). The contribution of selected components is illustrated by disabling each of them in turn.

It important to note that is easier to detect the tracking signals for active visual following than for tracking an object in passive stereo-motion image sequences. First, motion blur emphasizes the signal of target over the background. In passive visual following, the target's image slips across the retina and may thus be degraded by motion blur. During active pursuit, however, the eyes move to follow the target and stabilize the retinal image. Thus the image of the surrounding scene rather than the target moves across the retina and suffers from motion blur. The result is that image of the target is emphasized over the image of the background. Second, maintaining vergence isolates the target by disparity filtering. Holding vergence on the target enables the object to be isolated by simple zero-disparity filtering that detects objects at the fixation distance. Thus maintaining vergence on the target makes it possible to locate the target for pursuit control with simple precategorical visual processing. Third, active visual following also enables localized visual processing. The target's retinal location is roughly known because the pursuit system is keeping it near the center of view. This permits spatially localized visual processing.

Also note that *active binocular following* has advantages over:

monocular following. Vergence and binocular disparity help pick out the target. Monocular visual following must find ways to maintain invariants in the monocular images. This is difficult in images that are view-dependent, and it is also difficult to parse the complex optical flows that result from visually following an object, especially while the observer is also moving. Active binocular following, however, can use binocular cues to exploit physical properties that maintain invariants in the images. For instance, holding vergence on an object permits the object to be located in visual space by disparity filtering that is much simpler than stereoscopic interpretation. This provides the object's location to the pursuit system so gaze can be directed at the object.

binocular following without vergence. Image interpretation is simplified because the target has zero disparity (thanks to vergence control) and its retinal image is foveated (due to pursuit). Consequently, it is sufficient simply to ignore portions of the stereo images that contain non-zero disparities in order to locate the target. In

contrast, visual following with a binocular system of fixed vergence angle requires processing of potentially large disparities as the disparity of the target varies with its distance. (It is probably sufficient to be able to shift images quickly as an alternative to verging the camera system. The difference between these approaches is the difference in the viewpoint of an object that is observed by parallel and by verged binocular camera systems, and it is probably inconsequential in most cases.)

The limitations of the system described here should be clear. First, neither is any attempt is made to identify an interesting target for initiation of tracking, nor is the system able to acquire fixation of a non-foveated moving target. Both of these activities are crucially important for the completeness of a gaze control and tracking system. Second, binocular cues provide reliable information only for near targets (with, say, two or three "arm's lengths") because there must be sufficient range discontinuities to permit disparity differences to distinguish the target. Obviously, biological pursuit systems have capabilities of these kinds that the demonstration system lacks. Further, the normal mode of pursuit operation for animal systems is not likely to be precategorical; rather, the creature probably nearly always is able to recognize any object that it is tracking, and it is not clear how much and what sort of influence this has on the process of tracking an object. Certainly any advantages in robustness and performance that can be gained by the use of such knowledge should be sought by robot visual systems as well.

Nevertheless, the central claim of this thesis is that localizing attention in 3D space makes simple precategorical visual processing sufficient to hold gaze. The precategorical nature of the visual sensing means the algorithms are simple and do not require delicate tuning. Vergence control is not even required since the disparity filter can be designed to respond to any finite disparity. This basic idea can be used to advantage when holding gaze in dynamic situations.

7.1 Future Work

Future work could take several directions. Gaze holding itself can be improved by more sophisticated visual processing, and non-visual cues can be employed to increase performance. In addition, prediction can improve performance in tracking targets whose motion is predictable by mitigating the effects of delays in the system. Second, gaze holding can be incorporated in a larger repertoire of gaze controls, including saccades and control of head movements. Third, gaze control can be applied in the context of specific types of behavior, including visual perception, locomotion, and manipulation.

Gaze holding ability can improved considerably by using expectations to help distinguish the target's signal from the signals of distracting objects and clutter. Prediction

can be used to concentrate visual processing in the local area where the target signal is expected to be found. This applies to both motion and stereo disparity spaces as well as retinotopic space. Essentially, the system would model the trajectory of the target in three-dimensional head coordinates and project its expectations onto the retinotopic and disparity spaces. The expectation could be expressed as a Gaussian surface, say in retinotopic space, that emphasizes signals that fall within the expected envelope.

The non-uniform resolution of the primate retina could act as a fixed size and location version of this sort of mechanism. (In fact, the windowed image processing used by the demonstration system is just an extreme form of non-uniform resolution in visual processing.) A foveal-peripheral visual system with a central area of high-resolution surrounded by increasingly lower resolution emphasizes the central portion of the visual field. Thus a foveated target would be emphasized over peripheral surrounding objects. Similarly, small disparities would be more difficult to detect detect reliably in the periphery than the fovea due to the Nyquist limits of sampling frequency, so disparity estimation for vergence and disparity filtering for pursuit would be influenced less by objects that were in the horopter but were not foveated.

Another expectation that may help distinguish the target's signal is enabled by the nature of active visual following. When the cameras are following the target, its signal (*e.g.*, zero-disparity image) can be expected to be fairly persistent. This enables the use of age filters to help distinguish the target signal from distracting signals. This is a simplified form of coherent motion segmentation that relies on the target's optical flow being nearly zero. *I.e.*, if the target were being followed perfectly, its image should not slip across the retina at all, while the images of the surrounding scene should be slipping a relatively large amount. Even if the target is not being followed perfectly, its slip should still be relatively small compared with the background slip, and this could be used to segment and locate the target. If the robot and camera motions are smooth enough, it may be possible to anticipate the background flow and detect the target if it is moving inconsistently with the constraint determined by the robot and camera motions.

Concentrating on visual cues alone is an interesting constraint, but we have not lost sight of the fact that in successful systems visual cues probably are combined into a model of target motion which may be used to control gaze [Brown, 1990a; Coombs and Brown, 1990]. In addition, non-visual cues (such as head motions sensed by accelerometers or rate-gyroscopes) can inform gaze stabilization systems. Using such non-visual cues calls for models of the observer's physical plant so that the proper compensating movements may be made. Furthermore, the appropriate camera movements to compensate for head motion depend on the gaze location, and so a representation of the location of objects in three dimensions (or at least depth) is needed. There is increasing evidence that human and other primate gaze control systems do indeed make use of such cues [Paige, 1990; Snyder *et al.*, 1990].

The performance of any system that contains delays can be improved by using prediction to overcome the effects of predictable delays. Chapter 6 discusses some techniques for coping with delays in feedback systems, but obviously there remains considerable room for further study.

As well as improving gaze holding performance itself, the repertoire of gaze controls can also be expanded to include saccades and head movements as well as camera movements. As the number and complexity of gaze control components increase, so does the need for more sophisticated coordination among the control systems. For instance, if the head is turning to follow an object, the cameras need to be rotated less with respect to the head. Interaction of controls can be simple (say by preemption), or more complex (with controls aware of and cooperating with the actions of other controls [Brown, 1990b; Brown, 1990a; Brown, 1990c]). The former approach requires either breaking down the controls into orthogonal, non-interacting primitives or being content to have one control acting at a time. The latter approach requires more sophisticated modeling of the effects of interaction.

Another area for future exploration concerns the effect of gaze controls such as gaze shifts and gaze holding on visual perception. One obvious application is the support of stereo systems with limited fusional ranges [Nishihara, 1984; Olson, 1991]. More generally, systems that fixate must choose appropriate targets for the task they are performing. Thus gaze control at the highest level can be viewed as a resource management problem, in which limited sensory and computing hardware must be allocated so as to maximize the usefulness of the recovered information [Rimey and Brown, 1990].

Similarly, camera movement and fixation strategies could have strong implications for visual perception for locomotion. Some results [Bandopadhay, 1986; Aloimonos *et al.*, 1988; Raviv and Herman, 1991; Nelson, 1991] already suggest the potential impact of having a better understanding how specific kinds of eye movements influence the the observed optical flow (and consequently the visual processing that is necessary to achieve the desired behavior). For instance, does it help to keep the observer's cameras at a fixed angle in head space, or is it better to hold gaze on objects? Where is it best to look while traveling on a curvilinear path?

Gaze control and visual feedback can have considerable impact on robust visually-guided behavior in the form of indexical behavior. For instance, developing the capability for a robot to `go-to-the-fixated-object` could result in robust navigation based on visual feedback even in the presence of other sources of error (*e.g.*, slippage errors in dead-reckoning). Thus the need for precise estimates of the target and precise calibrations and visuomotor transformations is reduced simply by the use of feedback. An example of robotic behavior that notoriously requires high accuracy in visual perception, transformation and positioning is the task of picking up an object with a robot hand. However, with only coarse calibration, the hand could be carried most of the way toward the target in an initial open-loop reaching phase. A second, visually-servoed,

reaching and grasping phase can use visual perception of the hand and the tool in visual coordinates to make relative adjustments of hand position and orientation without requiring transformation to absolute motor coordinates. There is evidence suggesting that this two-stage reaching and grasping describes the way primates reach and grasp objects. This approach will be robust with less precise calibration than a completely open-loop system would require.

Bibliography

[Abbott and Ahuja, 1988] A. Lynn Abbott and Narendra Ahuja, "Surface Reconstruction by Dynamic Integration of Focus, Camera Vergence, and Stereo," In *Proc. of ICCV'88, the Second International Conference on Computer Vision, (Tampa, FL, December 5–8, 1988)*, 1988.

[Agre and Chapman, 1987] Philip E. Agre and David Chapman, "Pengi: an implementation of a theory of activity," In *AAAI*, pages 268–272, 1987.

[Allen, 1989] Peter Allen, "Real-Time Motion Tracking Using Spatio-Temporal Filters," In *Proc. of the DARPA Image Understanding Workshop*, 1989.

[Aloimonos *et al.*, 1988] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay, "Active Vision," *International Journal of Computer Vision*, 1(4):333–356, January 1988.

[Analog Devices, 1987] Analog Devices, Inc., Norwood, Massachussetts, *DSP Products Databook*, 1987.

[Bahill and Harvey, 1986] A. T. Bahill and D. R. Harvey, "Open-loop Experiments for Modeling the Human Eye Movement System," *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-16(2):240–250, 1986.

[Bahill and McDonald, 1981] A. T. Bahill and J. D. McDonald, "Adaptive Control Models for Saccadic and Smooth Pursuit Eye Movements," In A. F. Fuchs and W. Becker, editors, *Progress in Oculomotor Research*. Elsevier, 1981.

[Bajcsy, 1988] R. Bajcsy, "Active Perception," *Proceedings of the IEEE*, 76:996–1005, 1988.

[Ballard, 1989] Dana H. Ballard, "Reference Frames for Animate Vision," In *Proc. of the International Joint Conference on Artificial Intelligence*. AAAI, 1989.

[Ballard, 1991] Dana H. Ballard, "Animate Vision," *Artificial Intelligence*, 48:57–86, 1991.

[Ballard and Ozcandarli, 1988] Dana H. Ballard and Altan Ozcandarli, "Eye Movements and Visual Cognition: Kinetic Depth," In *Proc. of ICCV'88, the Second International Conference on Computer Vision, (Tampa, FL, December 5–8, 1988)*, 1988.

[Bandopadhay, 1986] Amit Bandopadhay, *A Computational Study of Rigid Motion Perception*, PhD thesis, University of Rochester, Computer Science Department, Rochester, New York 14627 USA, December 1986, (Also Technical Report # 221).

[Bar-Shalom and Fortmann, 1988] Yaakov Bar-Shalom and Thomas E. Fortmann, *Tracking and data association*, Academic Press, 1988.

[Barnard and Fischler, 1982] Stephen T. Barnard and Martin A. Fischler, "Computational Stereo," *ACM Computing Surveys*, 14(4):553–572, 1982.

[Bogert *et al.*, 1963] B. P. Bogert, M. J. R. Healy, and J. W. Tukey, "The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo-autocovariance, Cross-Cepstrum, and Saphe Cracking," In M. Rosenblatt, editor, *Proc. Symp. Time Series Analysis*, pages 209–243, New York, 1963. John Wiley and Sons.

[Brown, 1989] C. M. Brown, "Kinematic and 3D Motion Prediction for Gaze Control," In *Proceedings: IEEE Workshop on Interpretation of 3D Scenes*, pages 145–151, Austin,TX, November 1989.

[Brown, 1990a] C. M. Brown, "Gaze controls cooperating through prediction," *Image and Vision Computing*, 8(1):10–17, February 1990.

[Brown, 1990b] C. M. Brown, "Gaze controls with interactions and delays," *IEEE Transactions on Systems, Man, and Cybernetics*, IEEE-TSMC20(3), May 1990.

[Brown, 1990c] C. M. Brown, "Prediction and cooperation in gaze control," *Biological Cybernetics*, May 1990.

[Brown *et al.*, 1989] C. M. Brown, H. Durrant-Whyte, J. Leonard, and B. S. Y. Rao, "Centralized and Noncentralized Kalman Filtering Techniques for Trackingand Control," In *DARPA Image Understanding Workshop*, pages 651–675, May 1989.

[Brown *et al.*, 1988] Christopher Brown, Dana Ballard, Timothy Becker, Roger Gans, Nathaniel Martin, Thomas Olson, Robert Potter, Raymond Rimey, David Tilley, and Steven Whitehead, "The Rochester Robot," Technical Report 257, University of Rochester, Computer Science Department, 1988, Christopher M. Brown (Editor).

[Brown and Coombs, 1991] Christopher Brown and David Coombs, "Notes on Control with Delay," Technical Report 387, University of Rochester, Computer Science Department, Rochester, New York 14627 USA, August 1991.

[Burt *et al.*, 1989] P. Burt, J. Bergen, R. Hingorani, R. Kolczinski, W. Lee, A. Leung, J. Lubin, and H. Shvaytser, "Object Tracking with a Moving Camera: An Application of Dynamic Motion Analysis," In *Proceedings of the Workshop on Visual Motion*, 1989.

[Burt and Adelson, 1983] Peter J. Burt and Edward H. Adelson, "The Laplacian Pyramid as a Compact Image Code," *IEEE Transactions on Communications*, 31(4):532–540, April 1983.

[Carpenter, 1988] Roger H. S. Carpenter, *Movements of the eyes*, Pion, 2nd ed., revised and enlarged edition, 1988.

[Clark and Ferrier, 1988] James Clark and Nicola Ferrier, "Modal Control of an Attentive Vision System," In *Proc. of ICCV'88, the Second International Conference on Computer Vision, (Tampa, FL, December 5–8, 1988)*, 1988.

[Coombs, 1989] David J. Coombs, "Tracking Objects with Eye Movements," In *Proc. of the Topical Meeting on Image Understanding and Machine Vision*, North Falmouth, Cape Cod, MA, June 1989. Optical Society of America.

[Coombs and Brown, 1990] David J. Coombs and Christopher M. Brown, "Intelligent Gaze Control in Binocular Vision," In *Proc. of the Fifth IEEE International Symposium on Intelligent Control*, Philadelphia, PA, September 1990. IEEE.

[Coombs and Brown, 1991] David J. Coombs and Christopher M. Brown, "Cooperative Gaze Holding in Binocular Vision," *IEEE Control Systems*, June 1991.

[Coombs *et al.*, 1990] David J. Coombs, Thomas J. Olson, and Christopher M. Brown, "Gaze Control and Segmentation," In *Proc. of the AAAI-90 Workshop on Qualitative Vision*, Boston, MA, July 1990. AAAI.

[Corke and Paul, 1989] Peter I. Corke and Richard P. Paul, "Video-Rate Visual Servoing for Robots," Technical Report MS-CIS-89-18, Department of Computer and Information Science, University of Pennsylvania, GRASP Lab, Philadelpha, PA 19104, February 1989.

[Datacube, 1987] Datacube, Inc., Peabody, Massachussetts, *MaxVideo Installation and Software Manual*, 1987.

[Devore and Peck, 1986] Jay Devore and Roxy Peck, *Statistics The Exploration ond Analysis of Data*, West Publishing Company, 1986.

[Dorf, 1980] Richard C. Dorf, *Modern control systems*, Addison-Wesley, 3rd edition, 1980.

[Erkelens *et al.*, 1989a] C. Erkelens, J. Van der Steen, R. Steinman, and H. Collewijn, "Ocular Vergence Under Natural Conditions I: Continuous Changes of Target Distance Along the Median Plane.," *Proceedings of the Royal Society of London*, 1989.

[Erkelens *et al.*, 1989b] C. Erkelens, R. Steinman, and H. Collewijn, "Ocular Vergence Under Natural Conditions II: Gaze Shifts Between Real Targets Differing in Distance and Direction.," *Proceedings of the Royal Society of London*, 1989.

[Erkelens and Regan, 1984] C. J. Erkelens and D. Regan, "Human Ocular Vergence Movements Induced by Changing Size and Disparity," *Journal of Physiology*, 379:pp. 145–169, 1984.

[Fleet *et al.*, 1989] David J. Fleet, Allan D. Jepson, and Michael R. M. Jenkin, "Phase-based Disparity measurement," Research in Biological and Computational Vision RBCV-TR-89-29, Department of Computer Science, University of Toronto, November 1989.

[Geiger and Yuille, 1987] Davi Geiger and Alan Yuille, "Stereopsis and Eye-movement," pages 306–314, 1987.

[Girod and Kuo, 1989] Bernd Girod and David Kuo, "Direct Estimation of Displacement Histograms," In *Proc. of the Topical Meeting on Image Understanding and Machine Vision*, North Falmouth, Cape Cod, MA, June 1989. Optical Society of America.

[Heeger and Hager, 1988] D J Heeger and G Hager, "Egomotion and the Stabilized World," In *Proc. of ICCV'88, the Second International Conference on Computer Vision, (Tampa, FL, December 5-8, 1988)*, pages 435–440, December 1988.

[Heeger and Simoncelli, 1989] D J Heeger and E P Simoncelli, "Sequential Motion Analysis," In *Symposium on Robot Navigation*, pages 24–28, Stanford, CA, March 1989. AAAI.

[Heeger and Jepson, 1990] David J. Heeger and Allan D. Jepson, "Simple Method for Computing 3D Motion and Depth," In *Proc. of ICCV'89, the Third International Conference on Computer Vision, (Osaka, Japan, December 4-7, 1990)*, 1990.

[Heeger and Jepson, 1991] David J. Heeger and Allan D. Jepson, "Subspace Methods for Recovering Rigid Motion I: Algorithm and Implementation," *International Journal of Computer Vision*, 1991, in press.

[Horn, 1986] Berthold K. P. Horn, *Robot Vision*, The MIT Press, Cambridge, Massachusetts, 1986.

[Howard and Simpson, 1989] Ian Howard and W. Simpson, "Human Optokinetic Nystagmus Is Linked to the Stereoscopic System," *Experimental Brain Research*, 1989.

[Jenkin, 1991] Michael R. M. Jenkin, "Using Stereomotion to Track Binocular Targets," In *Proc. of CVPR'91, the Conference on Computer Vision and Pattern Recognition (Maui, Hawaii, June 3-6, 1991)*, 1991.

[Jepson and Jenkin, 1989] Allan D. Jepson and Michael Jenkin, "The Fast Computation of Disparity from Phase Differences," In *Proc. of CVPR'89, the Conference on Computer Vision and Pattern Recognition (San Diego, CA, June 4-8, 1989)*, pages 398–403, 1989.

[Krauzlis and Lisberger, 1989] R. J. Krauzlis and S. G. Lisberger, "A Control Systems Model of Smooth Pursuit Eye Movements with Realistic Emergent Properties," *Neural Computation*, 1989.

[Krotkov, 1989] Eric Paul Krotkov, *Active computer vision by cooperative focus and stereo*, Springer-Verlag, 1989.

[Kuglin and Hines, 1975] C. D. Kuglin and D. C. Hines, "The Phase Correlation Image Alignment Method," In *Proc. IEEE Int'l Conf. on Cybernetics and Society*, pages 163–165, 1975.

[Land, 1975] Michael F. Land, "Similarities in the Visual Behavior of Arthropods and Men," In Michael S. Gazzaniga and Colin Blakemore, editors, *Handbook of Psychobiology*, pages 49–72. Academic Press, 1975.

[Lee and Wohn, 1988] Sang Wook Lee and K. Wohn, "Tracking Moving Objects by a Mobile Camera," Technical Report MS-CIS-8-97, University of Pennsylvania, Computer and Information Science Department, November 1988.

[Lisberger, 1990] S. G. Lisberger, 1990, personal communication.

[Lisberger *et al.*, 1987] S. G. Lisberger, E. J. Morris, and L. Tychsen, "Visual Motion Processing and Sensory-Motor Integration for Smooth Pursuit Eye Movements," *Annual Review of Neuroscience*, 10:97–129, 1987.

[Marshall, 1979] J. E. Marshall, *Control of Time-Delay Systems*, Peter Peregrinus Ltd., 1979.

[Matthies *et al.*, 1989] L. Matthies, T. Kanade, and R. Szeliski, "Kalman Filter-Based Algorithms for Estimating Depth From Image Sequences," *International Journal of Computer Vision*, 3:209–236, 1989.

[Maxwell and King, 1990] J. S. Maxwell and W. M. King, "Disjunctive Saccades contribute to vergence in Rhesus Monkeys," In *Abstracts of Society for Neuroscience*. Society for Neuroscience, 1990.

[McDonald and Bahill, 1983] J. D. McDonald and A. T. Bahill, "Zero-Latency Tracking of Predictable Targets by Time-Delay Systems," *Int. Journal of Control*, 38(4):881–893, 1983.

[Miles *et al.*, 1991] F. A. Miles, U. Schwarz, and C. Busettini, "The Parsing of Optic Flow by the Primate Oculomotor System," In A. Gorea, editor, *Representations of Vision: Trends and Tacit Assumptions in Vision Research*, pages 185–199. Cambridge University Press, Cambridge, 1991.

[Nelson, 1991] Randal C. Nelson, "Qualitative Detection of Motion by a Moving Observer," In *Proc. of CVPR'91, the Conference on Computer Vision and Pattern Recognition (Maui, Hawaii, June 3-6, 1991)*, pages 173–178, 1991.

[Nishihara, 1984] H. K. Nishihara, "Practical real-time imaging stereo matcher," *Optical Engineering*, 23(5):536–545, September/October 1984.

[Olson, 1991] Thomas J. Olson, "Stereopsis for Fixating Systems," In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, Charlottesville, Virginia, October 1991.

[Olson and Coombs, 1991] Thomas J. Olson and David J. Coombs, "Real-Time Vergence Control for Binocular Robots," *International Journal of Computer Vision*, 7(1):67–89, November 1991.

[Paige, 1990] G. D. Paige, "Modulation of the linear vestibulo-ocular reflex (LVOR) by vergence," *Investigative Ophthalmology and Visual Science*, 31(4):121, 1990, Annual ARVO Meeting Abstract Issue.

[Papanikolopoulos *et al.*, 1991] N. Papanikolopoulos, P. Khosla, and T. Kanade, "Vision and Control Techniques for Robotic Visual Tracking," In *International Conference on Robotics and Automation*. IEEE, 1991.

[Pearson *et al.*, 1977] J. J. Pearson, D. C. Hines, Jr., S. Golosman, and C. D. Kuglin, "Video-rate Image Correlation Processor," In *SPIE v. 119, Applications of Digital Image Processing*, pages 197–205, San Diego, 1977.

[Raviv and Herman, 1991] Daniel Raviv and Martin Herman, "A New Approach to Vision and Control for Road Following," NISTIR 4476, National Institute of Standards and Techonology (NIST), Robot Systems Division, Bldg 220, Rm B124, Gaithersburg, MD 20899 USA, January 1991.

[Reading, 1983] R. W. Reading, *Binocular Vision: Foundations and Applications*, Butterworth, Boston, 1983.

[Rimey and Brown, 1990] R. D. Rimey and C. M. Brown, "Selective attention as sequential behavior: Modelling eye movements with an augmented hidden Markov

model," In *Proc. of the DARPA Image Understanding Workshop*, Pittsburgh, PA, September 1990. DARPA.

[Robinson, 1987] David Robinson, "Why Visuomotor Systems Don't Like Negative Feedback and How They Avoid It," In Michael Arbib and Allen Hanson, editors, *Vision, Brain and Cooperative Computation.* MIT Press, 1987.

[Rojer and Schwartz, 1990] Alan S. Rojer and Eric L. Schwartz, "Design Considerations for a Space-Variant Visual Sensor with Complex-Logartihmic Geometry," In *Proc. of the International Conference on Pattern Recognition*, Philadelphia, PA, June 1990.

[Shvaytser, 1988] Haim Shvaytser, "Detecting Motion in Out-of-Register Pictures," In *Proc. of CVPR'88, the Conference on Computer Vision and Pattern Recognition (Ann Arbor, MI, June 5-9, 1988)*, pages 696–701, 1988.

[Smith, 1957] O. J. M. Smith, "Closer Control of Loops With Dead Time," *Chemical Engg. Prog. Trans.*, 53(5):217–219, 1957.

[Smith, 1958] O. J. M. Smith, *Feedback Control Systems*, McGraw-Hill, 1958.

[Snyder *et al.*, 1990] Lawrence H. Snyder, Diane M. Pickle, and W. Michael King, "Does instantaneous vergence angle modify the vestibulo-ocular reflex in monkeys?," *Investigative Ophthalmology and Visual Science*, 31(4):121, 1990, Annual ARVO Meeting Abstract Issue.

[Swain, 1990] Michael J. Swain, *Color Indexing*, PhD thesis, University of Rochester, Computer Science Department, 1990, (Also Technical Report # 360).

[Thorpe, 1983] Charles E. Thorpe, "An Analysis of Interest Operators for FIDO," Technical Report CMU-RI-TR-83-19, Carnegie-Mellon University, December 1983.

[Tölg, 1991] Sebastian Tölg, "A Biological Motivated System to Track Moving Objects by Active Camera Control," In O. Simula, editor, *Proceedings of the International Conference on Artificial Neural Networks*, Espoo, Finland, June 1991. ICANN-91, Elsevier.

[Triesman, 1985] Anne Triesman, "Preattentive Processing in Vision," *Computer Vision, Graphics, and Image Processing*, 31:156–177, 1985.

[Tsotsos, 1988] John K. Tsotsos, "A 'Complexity Level' Analysis of Vision," *International Journal of Computer Vision*, 1(4), January 1988.

[Van der Spiegel *et al.*, 1989] J. Van der Spiegel, G. Kreider, C. Claeys, I. Debusschere, G. Sandini, P. Dario, F. Fantini, P. Bellutti, and G. Soncini, "A Foveated Retina-Like Sensor Using CCD Technology," In C. Mead and M. Ismail, editors, *Analog VLSI Implementation of Neural Systems.* Kluwer, 1989.

[von Kaenel *et al.*, 1991] Peter von Kaenel, Chris Brown, and Dave Coombs, "Detecting Regions of Zero Disparity in Binocular Images," Technical Report 388, University of Rochester, Computer Science Department, Rochester, New York 14627 USA, August 1991.

[Waxman *et al.*, 1988] A. M. Waxman, W. L. Wong, R. Goldenberg, S. Bayle, and A. Baloch, "Robotic Eye-Head-Neck Motions and Visual Navigation Reflex Learning Using Adaptive Linear Neurons," *Neural Networks Supplement: Abstracts of 1st INNS Meeting*, 1:365, 1988.

[Whitehead and Ballard, 1990] Steven D. Whitehead and Dana H. Ballard, "Active Perception and Reinforcement Learning," *Neural Computation*, 2(4), 1990, (Also In the Proceedings of the Seventh International Conference on Machine Learning, Morgan Kaufmann, June 1990).

[Woodfill and Zabih, 1991] John Woodfill and Ramin Zabih, "Real-Time Motion and Stereo Tracking," In *Proc. of the National Conference on Artificial Intelligence.* AAAI, July 1991.

[Yarbus, 1967] A. L. Yarbus, *Eye Movements and Vision*, Plenum Press, New York, 1967.

[Yeshurun and Schwartz, 1989] Yehezkel Yeshurun and Eric Schwartz, "Cepstral Filtering on a Columnar Image Architecture: A Fast Algorithm for Binocular Stereo Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7), July 1989.

[Young, 1971] L. R. Young, "Pursuit Eye Tracking Movements," In P. Bach y Rita, C. C. Collins, and J. E. Hyde, editors, *Control of Eye Movements.* Academic Press, 1971.

[Young, 1977] L. R. Young, "Pursuit Eye Movement - What is Being Pursued?," *Dev. Neurosci.: Control of Gaze by Brain Stem Neurons*, 1:29-36, 1977.

[Young and Stark, 1963] L. R. Young and L. Stark, "Variable Feedback Experiments Testing a Sampled Model for Eye Tracking Movemnts," *IEEE Trans. Professional Tech. Grp. on Human Factors in Electronics*, 4:38-51, 1963.

A Understanding the Cepstral Filter

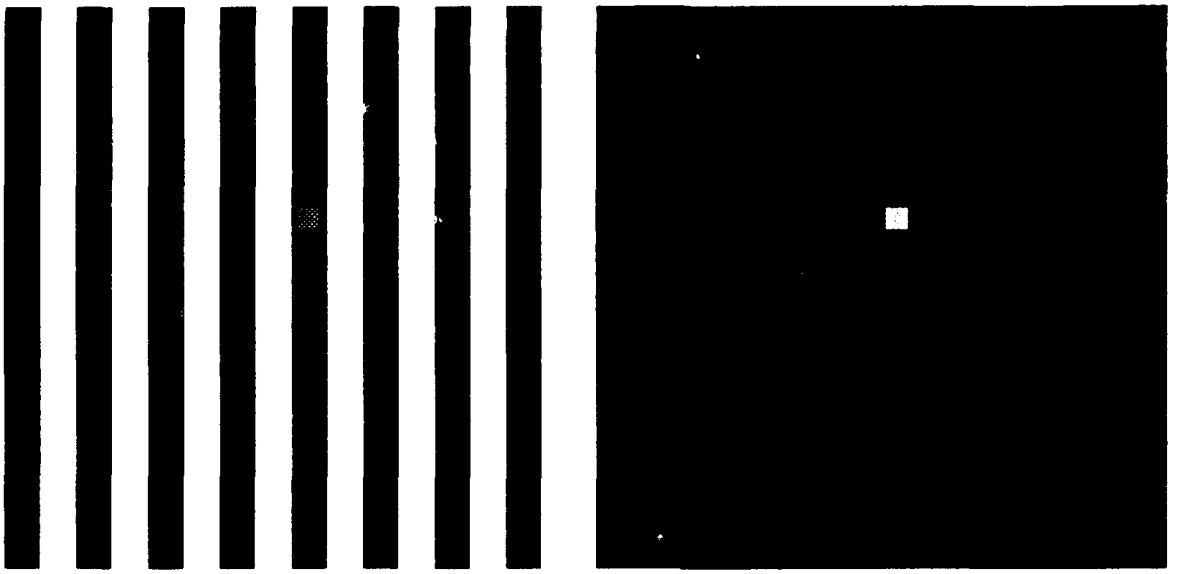
Our experience with the cepstral disparity estimator confirms Yeshurun and Schwartz's observation that the method is remarkably robust. The standard analysis (summarized in Section 3.5.1) explains what the algorithm does, but does not yield much insight into why it works so well. We feel that the algorithm is better understood by exploring its relation to autocorrelation; this view also suggests an alternative algorithm. The argument is as follows:

The cepstrum of a signal is computed by forming the power spectrum, taking the logarithm of each pixel, and Fourier transforming the result. Note that the power spectrum is just the Fourier *transform of* the autocorrelation function of the signal, and (like the autocorrelation function) is both real-valued and even symmetric. The forward and inverse Fourier transforms are equivalent for even, real-valued input functions. Therefore, the second Fourier transform in the cepstrum is equivalent to an *inverse* transform. The cepstrum, then, is the inverse transform of the log of the forward transform of the autocorrelation function. Without the logarithm step, therefore, the algorithm would simply compute the autocorrelation function.

The effect of taking the logarithm before the inverse Fourier transform can be seen by rewriting the log power spectrum as

$$\log|F|^2 = \frac{\log|F|^2}{|F|^2} F F^* = \left| \frac{\sqrt{\log|F|^2}}{|F|} F \right|^2$$

(where F^* is the complex conjugate of F). The right-hand side of this equation can be recognized as the power spectrum (*i.e.*, the Fourier transform of the autocorrelation) of a filtered version of the original function. In other words, the cepstrum can be thought of as autocorrelation with an adaptive non-linear prefilter. The prefilter is compressive in the frequency domain—it tends to make the power spectrum more nearly uniform, reducing the contribution of narrowband signals while leaving broadband signals relatively unaltered. Narrowband signals include such things as periodic patterns and large smooth blobs, both of which are poor correlation targets. By suppressing narrowband signals, therefore, the prefilter makes the input a better, less ambiguous correlation



(a) (b)

Figure A.1: Correlation target enhancement. (a) is an image $f(x, y)$ containing both good and bad correlation targets. Both types of targets have adequate high frequency content, but the periodic grid is subject to false matches both horizontally and vertically. (b) is the same image after application of the cepstral-equivalent filter. This image is a much clearer correlation target.

target. The effect can be seen by applying the appropriate prefilter to an image that contains both good and bad autocorrelation targets, as shown in Figure A.1. The periodic part of the signal has been largely suppressed, while parts of the image that have unique matches have been enhanced.

This view of the cepstrum suggests that any non-linear compressive function applied to the power spectrum should have a similar sharpening effect. Informal experiments suggest that this is indeed the case. For example, replacing the log step in the cepstral algorithm with a fourth root or arc tangent produces results that do not differ greatly from the standard cepstrum.

The ultimate compressive operator would be one that takes all input values to a constant. The Fourier transform of a constant is an impulse at $(0, 0)$, so this operator would provide the unhelpful information that the image matches itself perfectly at a disparity of zero. In order to get useful disparity information by this method one must find a way to preserve the phase information that is normally destroyed by formation of the power spectrum. For example, one might compute the transformed cross-correlation of the left and right images by multiplying the transform of one times the conjugate of

the transform of the other, and then rescale so that all entries in the resulting complex array have the same magnitude. This intuitively derived algorithm can be rigorously justified as a type of deconvolution, as follows:

The cepstral disparity estimator depends on the assumption that the right and left images differ only by a shift of d_h horizontally and d_v vertically. Given this assumption, however, a more direct approach is possible. The stated assumption is equivalent to the formula

$$R(x,y) = L(x,y) * \delta(x - d_h, y - d_v)$$

where $*$ represents convolution. Fourier transforming and solving for the disparity term gives

$$e^{-j2\pi(ud_h + vd_v)} = \frac{F_R(u,v)}{F_L(u,v)}$$

or

$$\delta(x - d_h, y - d_v) = \mathcal{F}^{-1}\left(\frac{F_R(u,v)F_L^*(u,v)}{|F_L(u,v)|^2}\right).$$

By hypothesis, however, F_L and F_R have identical magnitude spectra—they differ only in phase, because the left and right images differ only by a shift. Under this assumption, the division can be approximated by

$$\frac{F_R(u,v)F_L^*(u,v)}{|F_R(u,v)F_L^*(u,v)|},$$

which is the Fourier transform of the cross-correlation of the right and left images, rescaled so that all entries have magnitude one. That is, it is exactly the procedure suggested above by intuition. What was described there as a peak-sharpening operation turns out to undo the convolution of $L(x,y)$ with the disparity delta function.

Like the cepstrum, the deconvolution disparity estimator can be understood as correlation with an adaptive prefilter. In this case the effect of the prefilter is to obliterate the magnitude spectra of each image, so that the images differ only in phase. Deconvolution is thus equivalent to correlating phase images, and the technique is well known under the name of *phase correlation*. Kuglin and Hines [1975] first described the algorithm and showed that the height of the correlation peak and the distribution of the background values can be used to estimate the extent to which the two images do in fact differ by a shift. Pearson *et al.* [1977] describe a cleverly optimized hardware implementation of the algorithm that transforms 128 × 128 sample windows at 30 frames per second.

In theory, phase correlation should be somewhat faster than the Yeshurun and Schwartz cepstral disparity estimator for a given sample window size. This is because the cepstral estimator is based on Fourier transforms of windows of size $h \times 2w$, while phase correlation replaces those with fewer than twice as many transforms of windows of size $h \times w$. Since the running time of the Fast Fourier transform (FFT) rises more

than linearly with increasing sample window size, converting to phase correlation should reduce the time needed to estimate the disparity. However, this neglects the problem of wraparound. Like all Fourier-based approaches to discrete correlation, phase correlation (and the cepstrum) compute wrapped correlations, which can lead to ambiguities in the sign of the disparity. In the case of the cepstrum, the ambiguity can be resolved without padding by a strategy described in [Olson and Coombs, 1991]. For phase correlation, however, this strategy fails. The padding required to prevent wraparound overwhelms the apparent speed advantage of the phase correlation estimator.