VIDEO-RATE VISUAL SERVOING FOR ROBOTS

Peter I. Corke and
Richard P. Paul

MS-CIS-89-18
GRASP LAB 176

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104

February 1989

Acknowledgements: This work was supported in part by NSF IRI84-10413-AO2, MCS-8219196-CER, CCR8716975, NSF/DCR grants 8501482, NSF/DMC 8512838, U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027.

VIDEO-RATE VISUAL SERVOING FOR ROBOTS

Peter I. Corke [1]
Richard P. Paul [2]

1. Abstract

This paper presents some preliminary experimental results in robotic visual servoing, utilizing a newly available hardware region-growing and moment-generation unit. A Unix-based workstation in conjunction with special purpose video processing hardware has been used to visually close the robot position loop at video field rate, 60Hz.

The architecture and capabilities of the system are discussed. Performance of the closed-loop position control is investigated analytically and via step response tests, and experimental results are presented. Initial results are for 2 dimensional servoing, but extensions to 3 dimensional positioning are covered along with methods for monocular distance determination.

Finally some limitations of the approach and areas for further work are covered.

2. Introduction

This paper presents some preliminary experimental results in robotic visual servoing, utilizing an *off-the-shelf* hardware region-growing and moment-generation unit. This unit can process a binary scene, containing many regions, at video data rates and yields fundamental geometric parameters about each region. Such a device opens up many applications that were not previously possible due to the computational complexity involved.

The authors' aim was to develop a system capable of:

- at least frame rate (30Hz) analysis of video data
- functioning with a non-trivial scene
- determining object position as well as distance
- following a moving or stationary target
- functioning with a moving camera

A priori knowledge of the target object's shape and dimensions is assumed. A non-trivial scene is one with a complex background (not a black backdrop) and in which illumination levels may change. In this paper a distinction is made between tracking, and visual servoing. Much of the tracking literature assumes slow motion, or stop-start motion. Visual servoing however treats the camera-processing subsystem as a sensor with its own inherent dynamics, which the closed-loop position control must take into account.

There are many approaches to segmenting a complex scene so as to locate a possibly moving object. A general discussion of motion detection schemes is presented in[1]. Perhaps the simplest approach is image differencing[2] in which the difference between consecutive frames eliminates all detail from the scene except the edges of moving objects. This approach will fail if the velocity of the object is too low to differentiate motion energy from sensor noise, or if the camera is moving, and some hybrid strategy is needed to initially locate a static object.

---

[1] Research Scientist, CSIRO Division of Manufacturing Technology, Melbourne, Australia.

[2] Professor, CIS Department, University of Pennsylvania, Philadelphia.

A more sophisticated approach is optical flow analysis[3] which yields a dense velocity vector field from spatio-temporal changes in pixel intensities. This analysis is computationally expensive, particularly the smoothing operation[4] and requires special processing hardware to implement in real time. Other problems include interpretation of the velocity field for general camera motion with rotations[5] and sensitivity to camera noise[6]. The derived velocity field must still be segmented to separate background from foreground. A related technique is based on feature matching, where features are generally corner points[7, 8], edges[9] or regions. The change in position of various feature points between scenes can be used to estimate the velocity of those points. However features such as corners or lines are a sparse representation of the scene, and it is difficult to relate those features to the objects of interest in the environment without complex world models.

Sanderson[10] discusses visual servoing of robots and proposes a control scheme based on features extracted from the scene such as face areas of polyhedra, but gives no detail of the vision processing required to obtain those features. Kabuka describes two segmentationless approaches. One[11] uses an adaptive control based on parameters derived from the entire scene, while[12] is based on Fourier phase difference between consecutive frames.

Image segmentation is only the first step towards object tracking, the next problem is to identify the position of the regions within the scene, and must be done whether the input image represents intensity, velocity or range data. One approach to object location is to use a pyramid decomposition[13] so as to reduce the complexity of the search problem. Others have used moment generation hardware to compute the centroid of an object within the scene at video data rates[14, 15, 16].

This paper discusses a region feature approach; a harware unit performs segmentation of the intensity image to identify region features. The regions are then further investigated in software to identify the target, and tracking commands are then generated from its position. This approach is very general and works regardless of the state of motion of camera and target. The task of segmenting a complex scene in real-time is not trivial, but is nevertheless the basis of this work.

There is relatively little literature on visual servo implementations for robot position control. Many papers present only simulation results. Approaches to date could be classified as either stop-start (not dealt with here) or continuous. Hill and Park[17] describe a closed-loop position controller for a Unimate robot. They use binary image processing for speed, but also propose the use of structured lighting to reduce computational burden and to provide depth determination. Coulon and Nougaret[18] describe a digital video processing system for determining the location of one object within a processing window, and use this information for closed loop position control of an XY mechanism. They report a settling time of around 0.2s to a step demand. Kabuka[11] describes a two axis camera platform and image processor controlled using an IBM-PC/XT, and reports a minimum time of 30s to center on an object. Makhlin[19] discusses aspects of accuracy, tracking speed, and stability for a Unimate based visual servo system.

A number of related areas also require real-time visual object tracking. Gilbert[20] discusses automatic object tracking cameras used for rocket tracking. Andersson[21] describes a ping-pong playing robot that uses a real-time vision system to estimate ball trajectory for a subsequent paddle positioning planning algorithm. While the system described is not a closed-loop position controller many of the principles and problems are the same.

3. The experimental setup

The experimental setup is shown schematically in Figure 1. It comprises three major subsystems; robot control, image processing, and coordination.

3.1. Robot control

The robot control subsystem runs on a MicroVAX II connected to a Unimate Puma 560 robot and controller. Software on the MicroVAX can directly command robot joint angles using the RCI interface of RCCL[22]. The MicroVAX in this application is programmed to be a cartesian rate server; that is, it accepts cartesian velocity commands (in the tool reference frame) over an Ethernet using Unix datagram facilities. The server program consists of two asynchronous processes, one that accepts cartesian rate command packets from the network, and another that communicates robot joint angle measurements and commands with the Unimate controller via the RCI software interface. The robot servo interval is currently 14 or 28ms[22]. The manipulator's inverse Jacobian is computed every sample period using the method of Paul and Zhang[23]. The desired robot joint angles are simply calculated by

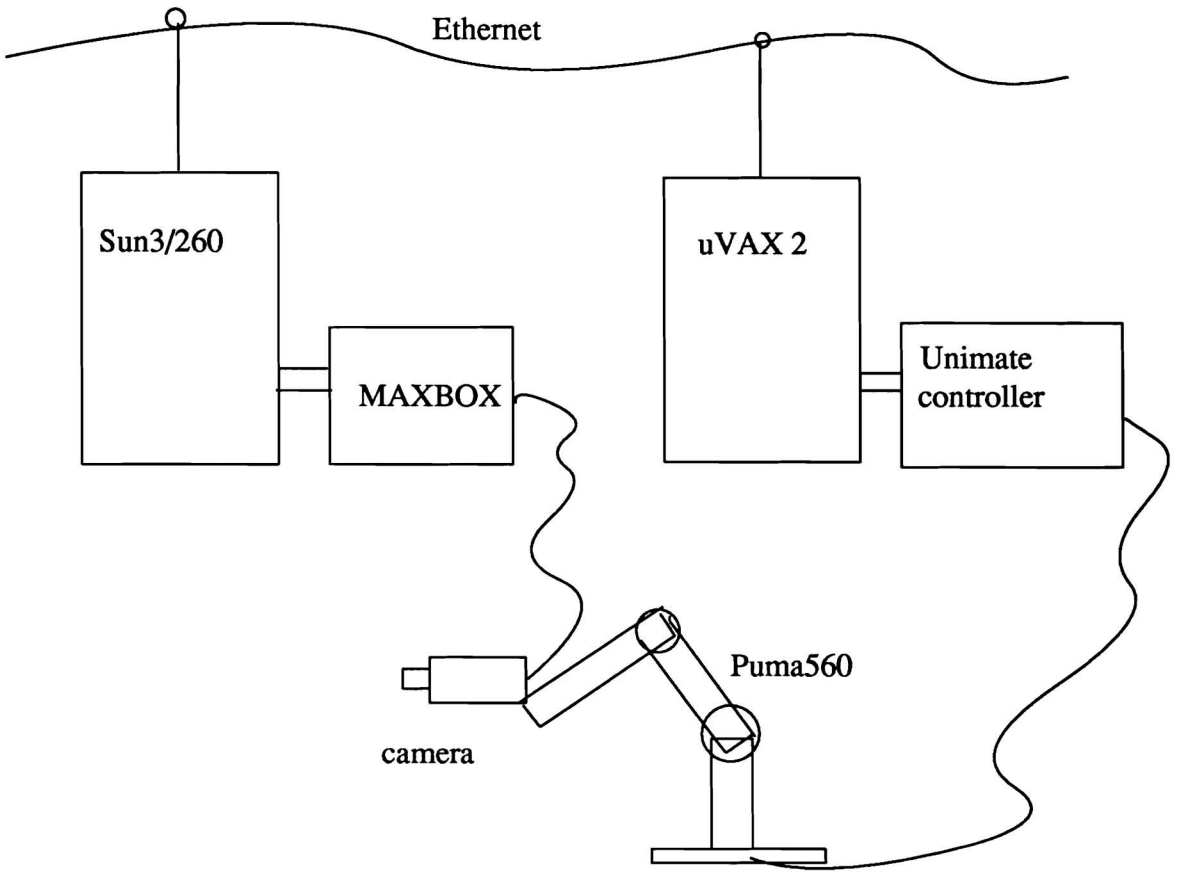

Figure 1. Experimental setup.

$$\theta_{d_{t+1}} = \theta_{d_t} + J^{-1} dx_{client}$$

where dx_{client} is the cartesian rate from the motion client. The control algorithm makes no use of the observed joint angle θ_{obs} since the robot is currently moving in free space, and it is the job of the Unimate servo subsystem to achieve desired joint angles.

Cartesian rate data is transferred in Sun's external data representation, XDR, using UDP protocol. XDR is needed since the client process, running on a Sun-3, has a different data byte orientation and floating point representation to the MicroVAX. Benchmark tests indicate that writing a command packet on the client takes $1100\mu s$ of elapsed time (kernel plus context switch time) using UDP, versus $2500\mu s$ using TCP/IP. Although UDP is an inherently unreliable protocol, and does not guarantee sequential delivery of packets, in the simple laboratory network environment these presented no problems. The server will accept datagrams from a number of different clients such as the visual tracker or a graphical teachpendant emulator.

3.2. Image processing

3.2.1. Segmentation

Robust segmentation of a scene remains one of the great problems of machine vision. Haralick[24] provides a survey of techniques applicable to static images. Unfortunately many of the algorithms proposed for segmenation are iterative and thus not suitable for real-time applications. This being the case most real-time implementations use contrived situations with dark backgrounds and simply threshold the video data. Much work has been done on automated approaches to threshold selection[25, 26].

Many suggest that incorporating edge information into the segmentation process increases robustness. Although 2D histograms of edge and intensity information have been used to extract targets from noisy FLIR[3] imagery in real time[27, 28, 29], many approaches still use simple thresholding.

An adaptive approach to segmentation is currently being investigated, using the capability of this hardware region-grower to perform many trial segmentations per second. One of the principle problems is how to rate the success of a segmentation so that an automatic system may modify or adapt its parameters[30].

3.2.2. Architecture for preprocessing

A functional representation of the image processing subsystem is shown in Figure 2, and is based on Datacube [4] pipeline processing modules. As mentioned above, one design aim was that the system be able to function robustly with respect to different scenes, non-black background and changing lightlevels. The architecure described has attempted to meet these requirements within the constraints of modules available.

The Datacube family of video processing modules are VMEbus boards that perform various operations on digital video data. The inter-module video data paths are patch cables installed by the user. The boards are controlled by a host computer via the VME bus. The video data paths run at 10Mpixels/s and are known as MAXBUS[4]. Horizontal and vertical timing is established

[3] Forward looking infrared

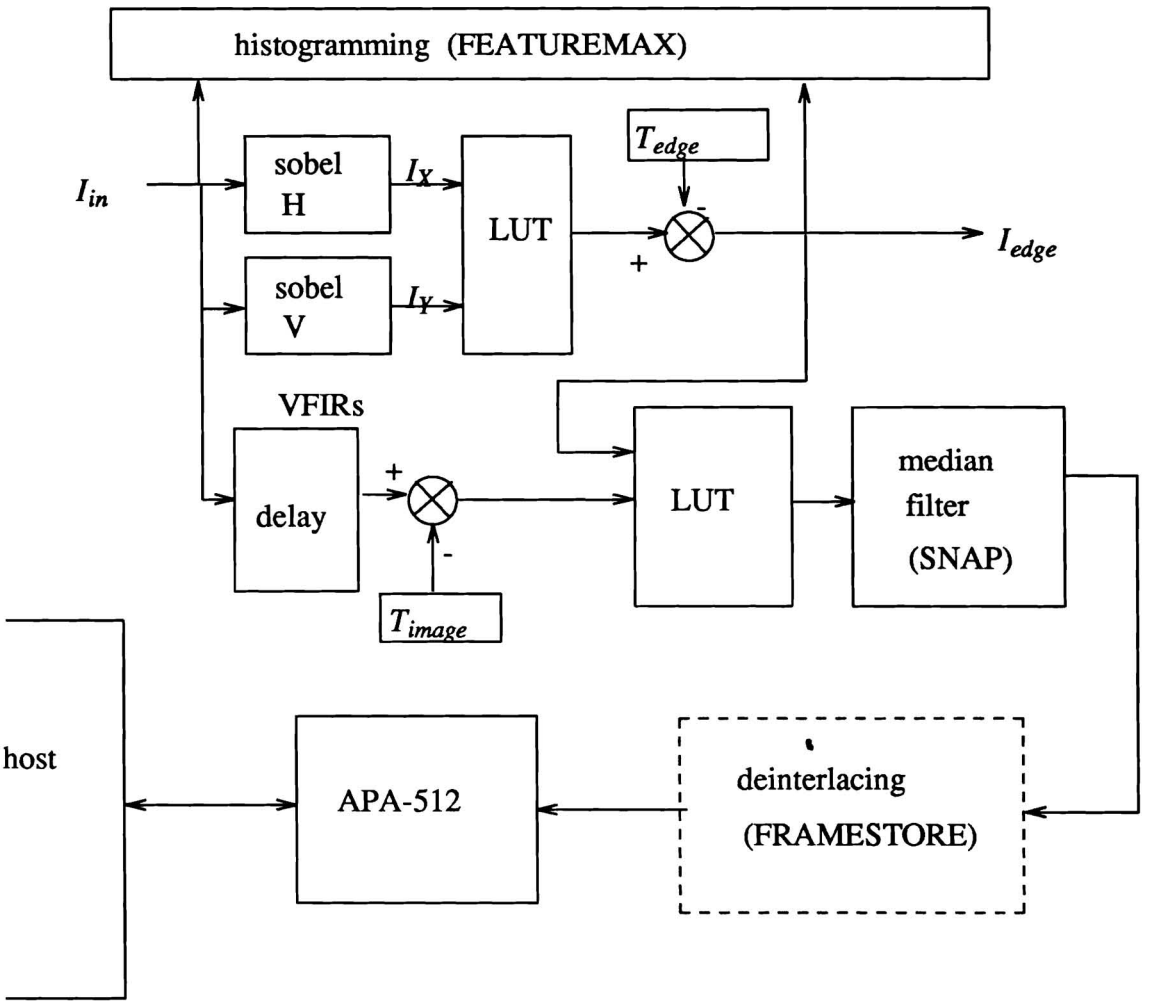

Figure 2. Image processing architecture.

---

by a separate timing bus linking all boards. All images are 512 x 512 pixels.

The two VFIR convolution modules generate horizontal and vertical Sobel gradient components

$$I_X = G_X * I_{in}$$

$$I_Y = G_Y * I_{in}$$

where G_X and G_Y are respectively the horizontal and vertical Sobel convolution kernels. A lookup table is used to compute the gradient magnitude, from which a gradient threshold, T_{edge}, is subtracted

$$Iedge = \sqrt{I_X{}^2 + I_Y{}^2} - T_{edge}$$

All arithmetic is performed in 8 bits 2's complement. The original video stream is delayed to

[4] Trademark of Datacube Inc.

synchronize it with the thresholded edge image, is offset by an intensity threshold T_{image}, and mapped through a lookup table which binarizes the pixels on the basis of both intensity and gradient magnitude. This provides some advantage in cases where regions of different intensity, but on the same side of the intensity threshold, separated, by distinct intensity gradients need to be resolved as separate regions.

Binary median filtering on a 3x3 neighbourhood is used to eliminate single pixel noise regions which may overwhelm the region-growing unit.

The Featuremax board is used to histogram edge and scene intensities periodically. The histogram data it computes is used by the host to select the thresholds T_{edge} and T_{image}.

For frame-rate processing the interlaced video data must be deinterlaced using double-buffered framestores in the Framestore module. Although only frame rate processing is discussed[31], an alternative approach which has been sucessfully tested is to process the interlaced data as two consecutive frames of half vertical resolution. The field processing approach has the advantage of eliminating the one frame time delay involved in deinterlacing, at the expense of twice as many regions to process per second.

3.2.3. APA-512

The APA-512[31] is a hardware unit designed to accelerate the computation of area parameters of objects in a scene. It was conceived and prototyped by the CSIRO Division of Manufacturing Technology, Melbourne, Australia, in 1982-4, and is now manufactured by Vision Systems Ltd. of Adelaide, Australia. The first author was a member of the design team.

The unit has some similarities to other hardware implementations of binary image processing systems[15, 32, 16]. Andersson's unit[14] computes moments of grey-scale data so as to improve accuracy when dealing with a quickly moving object. However it has no capability to

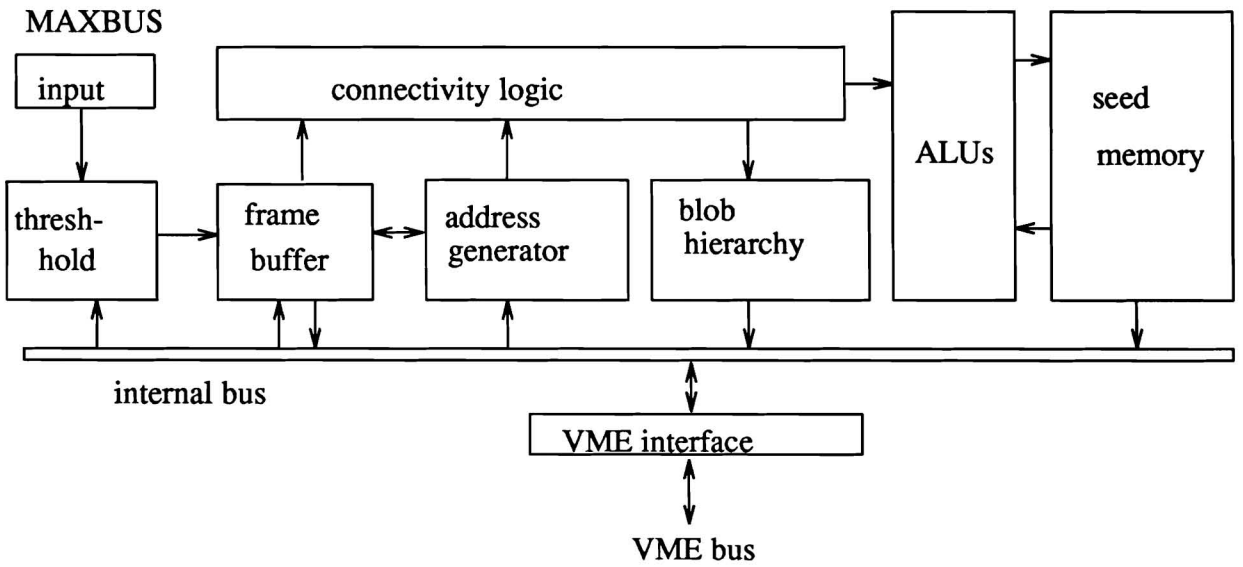

Figure 3. APA-512 block diagram.