

Real-Time Gray-Scale Video Processing Using a Moment-Generating Chip

R. L. ANDERSSON

Abstract—A system for performing visual processing on gray-scale images in real time (60 Hz) has been constructed. The custom VLSI moment generator chip computes area, center of gravity, orientation, and size. An image preprocessor allows separate moments to be computed for separate regions. A standard set of buses allows new processing elements to be easily added. The system is expected to find application in real-time sensor-based electronic assembly and in automated inspection and registration tasks. A simple application to a real-time visually servoed robot task is summarized.

I. INTRODUCTION

THE IMPLEMENTATION of a large class of potential robot applications is currently not feasible due to the lack of high-speed three-dimensional image-processing systems. Current assembly-task design relies on a person's manipulative skill, their ability to detect and correct for uncertainty in the environment, and the ability to concurrently inspect the work in progress. The latter two abilities, and perhaps even the first, rely on the ability of the person to sense the environment using visual and tactile perception. To automate many tasks, we must in effect emulate a person's perceptual skills.

Robotic vision systems must operate in "real time." By a real-time system, I mean one that does its job in a period of time that is at least partially determined by external constraints, not just a system that is fast. The need for real-time operation is often created by physical processes, such as when the solder has melted or when the object has moved out of reach of a robot, and by scheduling constraints such as assembly lines. The inability of a robot system to meet such constraints is an outright bar to the usability of the system.

Given that a system can do a specific job in the required time, there are important economic reasons for improving the operating rate. A system that runs twice as fast costs half as much per unit operation.

In the near term, we would like to build systems that are capable of processing simplified scenes at the rate necessary to use the data for real-time robotic control. Tasks often involve observing some feature in an image and causing some physical device to come to a specific position in relation to it for "handling." Examples include picking up objects and registration tasks. Sometimes, the position of the controlled device may not be accurately known, so it may be desirable to simultaneously monitor the position of both object and end

effector. An alternative is to put the camera on the robot and measure the relative displacement directly.

In the long term, we would like to construct systems capable of extracting information useful for manipulation and inspection from gray-scale images of three-dimensional scenes in real time. Such systems will be an important component of larger systems for monitoring an entire three-dimensional workspace for collisions, for verifying machine operation, for assuring quality, and for performing detailed assembly jobs.

A. Current Vision Systems

Commercial vision systems primarily use binary processing, typically by thresholding and run length compressing the image to reduce the amount of data before storing it in a fairly general-purpose processor. Once read in, the data is processed for many tenths of a second before some decision is made. The canonical basis for these schemes is [6]; there are many current commercial imitators.

Another class of fast computer vision algorithms is "image processing" algorithms. These algorithms take an image as input and produce an image as output. Most often, the processing is done by a convolution operator used to smooth the data or find edges. Special-purpose hardware may be built to perform the processing in pipelined fashion on the video stream (for an example, see [3]). From the perspective of robotics applications, such algorithms are not directly useful, as they do not reduce the amount of information which needs to be processed, although they may simplify subsequent feature extraction operations. Reducing the amount of data to be processed without eliminating essential image content is the fundamental problem in processing images in real time.

Advanced research-oriented systems (under the heading of "image understanding") typically read a gray-scale image into a frame buffer before processing it for several seconds, minutes, or even hours. Although these systems are slowly and steadily advancing in capability, their formidable processing requirements have severely limited their application to the robotics environment.

B. Systems Approach

The approach described here trades a lot of the flexibility of the image understanding methods for real-time operation. In the future, we would like to be able to adapt more algorithms from the world of image understanding to real-time processing.

A typical system configuration is shown in Fig. 1. The modules fall into two categories: image processors in which an

Manuscript received November 1, 1984; revised May 6, 1985.

The author is with AT&T Bell Laboratories, Room 4G608, Crawford's Corner Road, Holmdel, NJ 07733, USA.

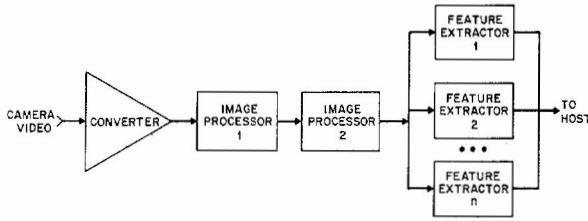


Fig. 1. Typical series/parallel system.

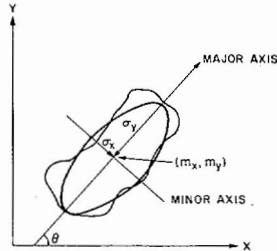


Fig. 2. Parameters of blob extracted by moments.

image is input and another image output; or "feature extractors" in which an image is input and some set of parameters output. The image processors are connected serially, whereas the feature extractors are in parallel—similarly, perhaps, to the human brain.

Each individual processing module is implemented on a single Multi-Bus board (Intel Corporation). The modules are unified by two types of buses (not including the Multi-Bus) whose signal locations and functions have been standardized. One bus contains a video signal digitized to eight bits of gray scale with 256×240 pixels per frame, 60 frames per second. The second bus is used to return a computer-selected video stream from one of the modules to the digitizer board for output on a monitor. Additional boards may be easily incorporated by designing them to utilize the busses correctly.

II. MOMENT GENERATOR

Moments have been in use in computer vision for some time [7], [8], and their use in physics and statistics goes back much farther. The equation defining the moments $M^{m,n}$ of an intensity array $a_{i,j}$ is

$$M^{m,n} = \sum_{i,j} a_{i,j} i^m j^n, \quad (1)$$

where $m + n$ ($m, n \geq 0$) is the order of the moment, i is the column, and j is the row.

The zero through second-order moments are sufficient to find the area, center of gravity, angle to major axis, and standard deviation along major and minor axes for an object, approximating the object as an ellipse, as shown in Fig. 2. These quantities are directly useful in picking up an object, or in guiding further visual processing, for example. Second and higher-order moments may be combined to form "invariants" which are used to characterize an object for purposes of discriminating among members of some set of objects.

The amount of time required to compute gray-scale moments has hindered their use. On a VAX 11/780 with floating-point accelerator, a direct calculation of the zero through second-order moments of a 256×256 image takes 6.5 s. Straightforward hardware implementations of moment calculations require large numbers of multipliers, accumulators,

registers, and supporting logic. Hybrid electrooptical approaches are possible [2], but suffer the same problems, in terms of accuracy, stability, and dynamic range, that are typical of analog computers. The moment generator system is intended to make the computation of moments easy enough for use as a new primitive for image examination.

The moment computation has been integrated onto a VLSI chip capable of computing a single zero through second-order moment of a gray-scale image in real time. Since there are six such moments, the moment processor module contains six chips. A number of techniques used to make the chip possible will be discussed below.

A. Power-Vector Generation

We consider moment generation as a dot product

$$M^{m,n} = \sum_t a_t p_t^{m,n} = (\bar{a}, \bar{p}^{m,n}), \quad (2)$$

where the elements of the vectors are in the same order as a normal TV scan, $t = i + 256j$. The element $p_{i,j}$ of \bar{p} will be referred to interchangeably with p_{i+256j} .

The equation defining $\bar{p}^{m,n}$ is

$$p_{i,j}^{m,n} = i^m j^n. \quad (3)$$

The element $p_t^{0,0}$ is one for all t . The first order moments require a counter for either x or y , depending on the moment. Apparently, a second order \bar{p} requires two counters and a multiplier. We can write the next value of each second order \bar{p} as a function of the previous one,

$$p_{i+1,j}^{2,0} = p_{i,j}^{2,0} + 2i + 1 \quad (4)$$

$$p_{i+1,j}^{1,1} = p_{i,j}^{1,1} + j \quad (5)$$

$$p_{i,j+1}^{0,2} = p_{i,j}^{0,2} + 2j + 1 \quad (6)$$

with special cases for top of screen and left margin. We can build an iterative \bar{p} generator composed of a single counter, a shifter, an adder, some "and" gates, and a small control programmable logic array (PLA).

B. Bit Decomposition

We can decompose \bar{a} as

$$\bar{a} = 2^7 \bar{a}_7 + 2^6 \bar{a}_6 + \dots + \bar{a}_0. \quad (7)$$

If we substitute equation (7) into (2) and distribute, we obtain

$$M^{m,n} = 2^7 (\bar{a}_7, \bar{p}^{m,n}) + 2^6 (\bar{a}_6, \bar{p}^{m,n}) + \dots + (\bar{a}_0, \bar{p}^{m,n}). \quad (8)$$

Computation of the dot products in (8) requires only $1 \times n$ bit multiplication, which may be implemented by n "and" gates, where n is the number of bits in \bar{p} .

At the end of each frame, we must compute

$$M^{m,n} = 2^7 F_7 + 2^6 F_6 + \dots + F_0 \quad (9)$$

where

$$F_k = (\bar{a}_k, \bar{p}^{m,n}). \quad (10)$$

Equation (9) can be evaluated only once per frame (60 times per second) using Horner's method of polynomial evaluation. The calculation is performed by the host processor which controls the system.