Cost-Effective Vision Systems for Intelligent Vehicles

I. Masaki (*), S. Decker, A. Gupta, B. K. P. Horn, H-S. Lee,
D. A. Martin, C. G. Sodini, J. K. White, J. L. Wyatt, Jr.

Massachusetts Institute of Technology
(*) Room E40-159, MIT, Cambridge, MA 02139, USA
telephone (1)617-253-8532, fax (1) 617-258-7334, masaki@mit.edu

1. Introduction

The cost of machine vision systems is limiting their applications. The goal of this project is to develop cost-effective vision systems for a number of intelligent vehicle applications. We propose concurrent design of algorithms and hardware as a way of improving the cost and performance of vision systems. With this approach, we develop custom hardware rather than using off-the-shelf microprocessors. This paper describes three example systems which we have developed or are developing: car following, lane detection, and stereo vision. With the car following system, a driver pushes a button when he or she has a car in front which he/she would like to follow. The vehicle speed is controlled automatically using a vision system to keep a constant distance to the car in front. This system is described in section 2. Section 3 is devoted to the lane sensing vision system. Both the car following and lane sensing systems were implemented using custom digital hardware (described in section 4) at General Motors when the first author was working with GM. The stereo vision system in section 5 is currently being developed by a group at Massachusetts Institute of Technology. A feature of this stereo vision system is that it uses analog/digital hybrid technology to reduce system cost without sacrificing performance.

2. Car Following

Car following is useful for a number of applications, including

- adaptive cruise control
  (or intelligent cruise control),
- convoy driving for AHS (Automated Highway Systems), and
- automatic go-and-stop control on congested roads.

The vision system for car following outputs the positional offsets in three dimensional coordinates. One dimension is the distance change to the foregoing vehicle. The other two dimensions are the image frame horizontal (x) and vertical (y) positional offsets of the foregoing vehicle. The distance change information can be used to keep a constant distance to the foregoing vehicle. The horizontal offset (x) can be used to control the steering with some time delay so that the vehicle will follow the trajectory of the foregoing vehicle, if the vehicle control system is structured in this way. A block-diagram of the car-following system is shown in Figure 1, and more details are described in the following sections.

2.1 Edge Detection

With the adaptive cruise control applications, a binary edge image of the foregoing vehicle is extracted and stored, as a template, when a driver pushes a button to start car-following. The template region in the image frame is defined automatically by using two-

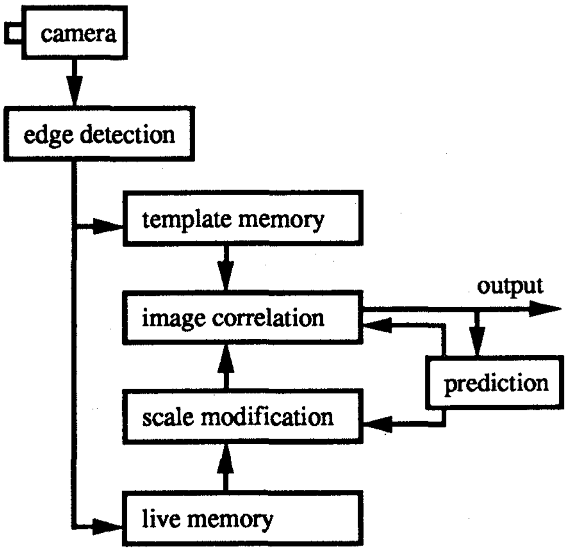

Figure 1. Car Following System

dimensional geometric knowledge of vehicles including vertical symmetry. After storing the template image, a live binary edge image is continuously updated and correlated with the stored template image ten times a second. Edge thickening is performed on the live edge image to compensate for pose uncertainty and camera quantization error. For example, the pose of the foregoing vehicle changes when it goes into a curve and rotates around the vertical axis. Edge thickness is programmable. Typical thicknesses are one pixel for the template and two pixel for the live edge image.

2.2 Prediction

The foregoing vehicle's location changes are smooth so it is possible to predict the possible location and size ranges from the results of processing the previous image. Different estimate schemes predict the ranges with different success rates and processing costs. We chose the simplest scheme because the position of the vehicle in the image changes slowly for closely-spaced frames. For example, suppose that the time interval is 100 ms, the image is 128 x 128 pixels, the distance to the foregoing vehicle is 30 m, the width of the field of view at 30 m is 10 m, and the lateral speed of the vehicle is 2 m/s. Then the positional change will be less than three pixels. Our system works by searching the image in a +/- 3 pixel range about the vehicle position in the previous frame. Our highway experiments indicate that the largest position changes of the foregoing vehicle in the image are due to pitching of the automated vehicle. The pitch motion depends heavily on the vehicle suspension.

2.3. Scale Modification

The scale modification changes the scale of the image to compensate for the change in the distance to the foregoing vehicle. For example, 200 % scale modification is required when the vehicle distance has been changed from 20 m to 40m. The current prototype is designed to handle the scale changes from 50% to 200% with 64 steps. We tried both linear and non-linear schemes to assign 64 steps to the range of 50% to 200%. In the linear scheme, we divided the scale range with an equal scale difference between adjacent steps. In non-linear scaling, we gave a higher distance resolution for the cases in which the foregoing vehicle is closer, because it is more important to measure the distance accurately when the vehicle is closer. In our non-linear scheme, the ratio of the scale factor between the adjacent scale steps is constant as shown below:

$$1 / \text{Scale-Factor(step K)} = \text{Delta-R} * (1 / \text{Scale-Factor(step K-1)}) \quad (\text{for K} = 1 \text{ through } 64)$$

$$\text{Scale-Factor(step 0)} = 0.5$$

$$\text{Delta-R} = (2/0.5)**(-63)$$

(where ** represents exponent)

It is not necessary to generate 64 differently scaled images for each process cycle. The system generates only three scales to minimize the processing time. Suppose, for example, that the system has 64 scaling steps which cover linearly from 50% through 200% of the original distance of 40m and that the foregoing vehicle makes a sudden slow down with the deceleration rate of 0.5 G when the distance is 40m. The difference between the adjacent scale steps is (80m-20m)/63. The system generates three scales for the distances of 39.4 m, 40.0 m, and 40.6 m, respectively. The distance change caused by the deceleration of 0.5 G for 100 ms (which is vision system's cycle time) is 0.05 m. The scale range of three is more than sufficient to cover the possible maximum distance change.

2.4. Image Correlation

The scaled live edge image is correlated at different offsets with the stored template edge image. The offset range is 17x17 with the current prototype because the sizes of the template and scaled live images are 64x64 and 72x72 pixels, respectively. The image correlation process calculates the probability of (3x17x17) cases. The image correlation process has two modes to choose from. The first mode accumulates the number of white-and-white pixel pairs between the live and template edge images as shown below:

$$PL = \sum L(n,m) \, (*) \, T(n,m)$$

where

PL : Probability that the hypothesis is correct.
L(n,m): Live binary edge pixel value at the image location of (n,m) in the image generated by the scale modification unit. 1 if it is an edge. Otherwise 0.
(*) : Operator defined as follows: A(*)B = 1 if A = B = 1 Otherwise, A(*)B = 0.
T(n,m): Template binary edge pixel value at the image location of (n,m) in the image generated by the horizontal translation unit. 1 if it is an edge. Otherwise 0.

Another mode is defined as follows:

$$PL = \sum L(n,m) (*) T(n,m)$$

where

[*] : Operator defined as follows: A(*)B = 1 if A=B=1 or A=B=O Otherwise 0.

3. Lane Sensing

The algorithm for lane sensing shares a number of common ideas with the car-following algorithm described above, including shape matching and temporal prediction [1]. Differences between the lane sensing algorithm and the car-following algorithm include multiple templates and a lack of scaling. This section will describe the lane sensing algorithm.

The lane-marking edges can be located by correlating a live edge image with multiple edge segment templates representing different edge orientations. In the first step, the live edge image is divided into horizontal regions. Each region is 8-pixel-high and 128-pixel-wide. In each region, the left side edges of the lane marks are located by correlating with multiple templates. The height of the region was selected to be the same as the height of the template images. The width was set at the half-width of the live image. Thirty-six template images were loaded in a memory chip. Each image is 8x8 pixels and represents a different rotation of a straight line. The rotational difference between adjacent template images is (180/31) degrees. It is small enough to avoid discontinuous changes between any adjacent straight line template images.

To minimize correlation time, only five predicted templates out of the 32 are used. The prediction algorithm chooses five templates by selecting the previous best match and its two neighbors on either side. So far, we have found no need for a more sophisticated prediction algorithm.

4. Digital Custom Hardware for Car Following and Lane Sensing

A focus of this research was to develop application-specific digital processor architectures to make the system faster, low power consumption, more compact, and less expensive in mass production. We worked on the algorithm and the hardware simultaneously rather than developing the algorithm first. The two phase method, which includes the first step for algorithm development and the second step for hardware development, accelerates algorithm development, but may not be the best way to develop highly efficient systems. Algorithm development is always guided consciously or unconsciously by the existing processor architecture. The constrains caused by the processor architecture can be changed in variety of ways depending on the type of architecture. It is important to consider tradeoffs between algorithm and hardware from the very early stages of system development. We developed some application-specific processors and implemented a portion of them into an application-specific integrated circuit chip. We also made the system flexible enough so that both car following and lane sensing algorithms can be carried out by the identical hardware with different software set-ups.

Image correlation requires the most computational power of the architecture, and therefore it was implemented in an application-specific integrated circuit chip based on a two-micron CMOS technology. The chip includes 40,000 transistors in 200 x 190 mils. The architecture of the chip is shown in Figure 2. The chip works at the speed of 6M Hz. The processing speed is more than 100 million pixel operations per second due to the application-specific parallel pipelined architecture. This is about a 170 fold speed up compared to off-the-shelf microprocessor-based systems [2].

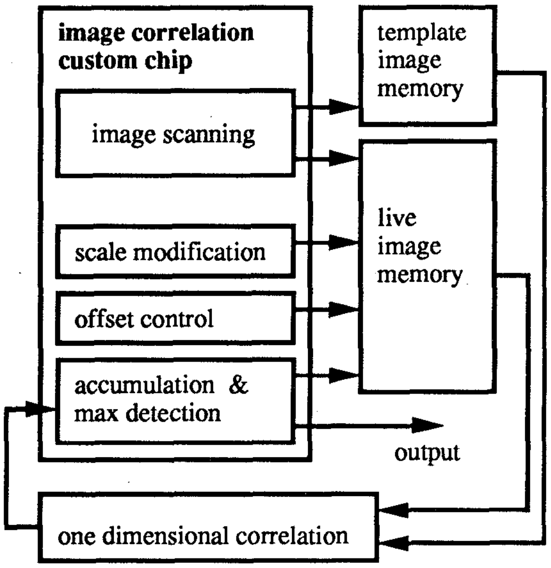

Figure 2. Digital Hardware for Correlation

5. Analog/Digital Stereo Vision

A significant technical issue which we found through the development of the car-following and lane-sensing vision systems was that we need a technical breakthrough to reduce the system cost. We propose an analog/digital hybrid architecture as a way of reducing the system cost significantly [3, 4]. Advanced analog technologies have the following merits:

- Analog circuitry requires significantly smaller silicon area for a variety of visual processes.

- Small silicon area leads to high processing speed, low cost, small size, and/or low power consumption.

The limitations of the analog approach are:

- moderate accuracy
  (1% with conservative fabrication process)

- limited data storage time.

We took a trinocular stereo algorithm as a test case to evaluate the cost-effectiveness of the analog/digital hybrid approach. The first author and his group at General Motors implemented this trinocular stereo algorithm into a custom digital hardware and tested it on highways (see Figure 3) [5]. The experimental results were encouraging but we recognized that it is important to reduce costs.

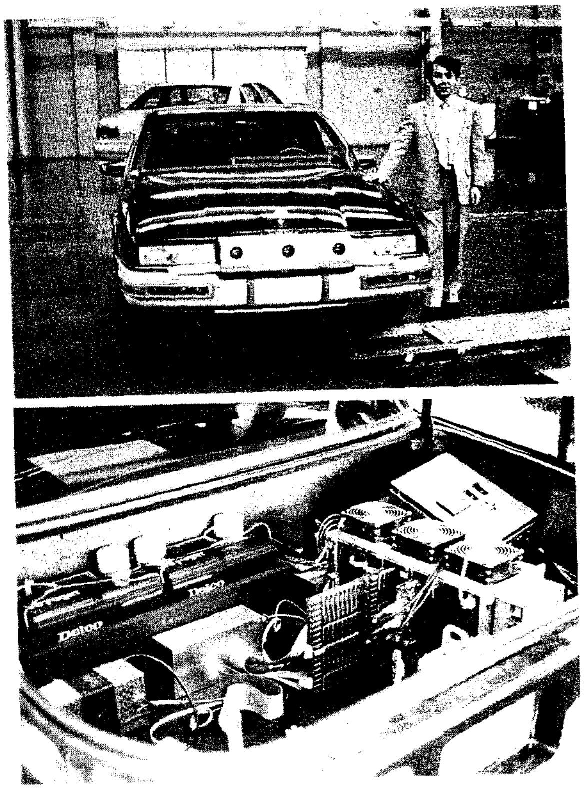

Figure 3. Digital Custom processor for Stereo

The analog/digital stereo vision system which is being developed by our MIT group has two

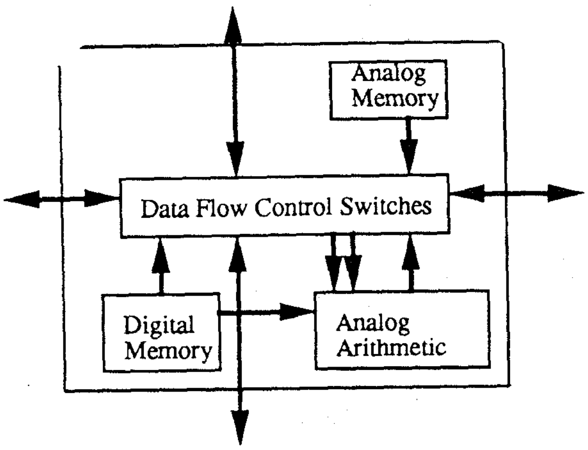

Figure 4. Processing Element

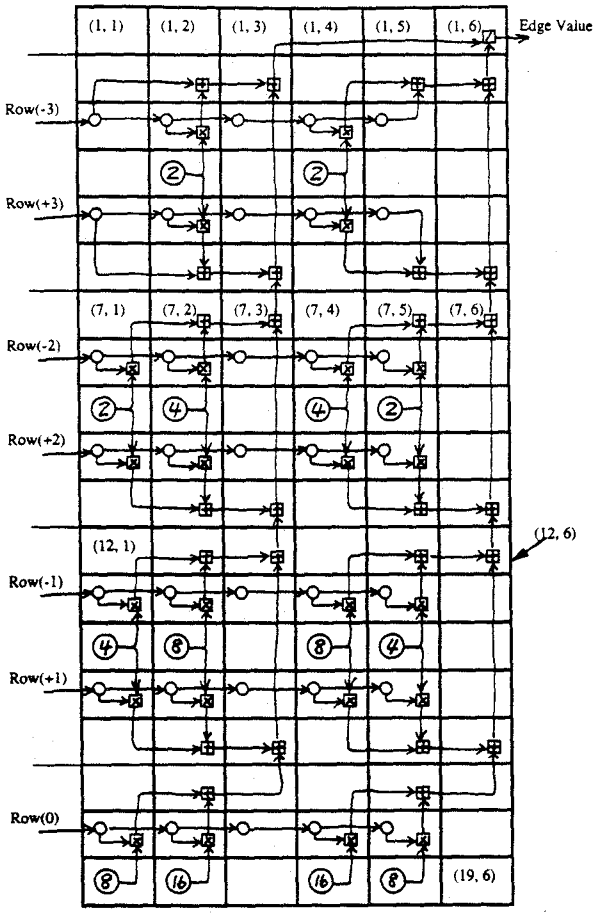

Figure 5. Example of Array Processor Programming

significant features: an analog/digital array processor and a wide dynamic range CCD imager. The analog/digital array processor chip will eventually have 64x64 processing elements on a single chip. Different processing elements will carry out different instructions at any given time because the chip uses a MIMD (Multi-Instruction Multi-Data) scheme instead of a SIMD (Single-Instruction Multi-Data) scheme. Every processing element consists of four units as shown in Figure 4. Figures 5 shows programming example. With this example, the edge detection algorithm is mapped on the array of 19x6 processing elements. The weight values of the kernel are stored in analog memory units shown in circles and the intensity image is fed to the left side of the array in parallel. The calculated edge value is output from the top right element. For stereo correlation, the connections among the elements are reconfigured and each element is re-programmed.

We are also developing a wide dynamic range CCD camera chip so we will be able to image vehicles both in the shadow and under strong sunshine. The principle is shown in Figure 6. The height of the energy barrier between each pixel well and the common drain is low in the early stage of each image acquisition period (TE) and is increased to a high level for the late stage of each period (TL). At any given time, the electrons in the well with energies that exceed the gate height are discharged from the well to the drain over the gate. The output for low-intensity light L1 does not change with this time-varying gate height. In contrast, the output for high-intensity light is suppressed from H1 to H2 because some electrons are discharged in the early stage of the acquisition period. The gate-height function can be changed dynamically based on the nature of images. Adding more steps to the energy barrier can produce even greater compression.

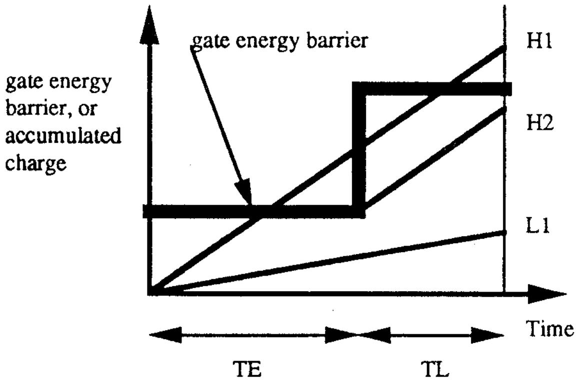

Figure 6. Wide Dynamic Range CCD

Acknowledgment

The authors are indebted to G. G. Dodd and other people at General Motors. Part of this work has been supported by NSF and ARPA under Contract No. MIP 9117724.

References

[1] I.Masaki, Vision-based Vehicle Guidance, Springer-Verlag, 1991
[2] I.Masaki, Industrial Vision Systems Based on Application-Specific IC Chips, IEICE Transactions on Electronics, vol.E74, no.6, 1728-1734, June 1991
[3] J.L.Wyatt, C.Keast, M.Seidel, D,Standley, B.Horn, T.Knight, C.Sodini, H-S.Lee, T.Poggio, Analog VLSI Systems for Image Acquisition and Fast Early Vision Processing, International Journal of Computer Vision, pp.217-230, 1992
[4] C.Mead, Analog VLSI and Neural Systems, Addison Wesley, 1989
[5] I. Masaki, ASIC Approach for Vision-Based Vehicle Guidance, IEICE Transactions on Electronics, vol. E76-C, no.12, pp 1735-1743, December 1993