Integrated Optical Motion Detection

Thesis by

John Edward Tanner

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

5223:TR:86

CALIFORNIA INSTITUTE OF TECHNOLOGY

Computer Science Department

Pasadena, California

1986

(Submitted May 15, 1986)

Acknowledgments

I thank Carver Mead for guiding me to this problem area, providing useful suggestions all along the way yet giving me enough flexibility for it to be fun. Carver helped me over some stumbling blocks I had erected and helped me begin to learn how to do research. Carver provided the intellectual and physical environment of the research group. He encouraged the richness of interaction so critical to progress in understanding.

Many members of the research group helped directly and indirectly through the sharing of ideas and program support. Special thanks to Massimo Sivilotti for supporting the chip design tools, Mary Ann Maher for the instrument control and data plotting software, Dave Gillespie for schematics capture and system support.

This thesis was prepared using LaTeX and printed on an Apple LaserWriter. Figures were drawn using a graphical editor written by Glenn Gribble. I thank Glenn and Calvin Jackson for supporting TeX and the surrounding system software.

Thanks to the people who reviewed early versions of this thesis and improved it substantially by their comments, my committee—Carver Mead, Richard Lyon, John Hopfield, David Van Essen and especially Al Barr—and other friends and colleagues who proofread this thesis—John Wawrzynek, Massimo Sivilotti, Lars Nielsen, Bill Dally, Michelle Mahowald, and Calvin Jackson.

The System Development Foundation provided support for this research.

My wife, Linda, gently kept me working through the times that weren't so much fun and shared with me the times that were. She drew some of the figures for this thesis and provided support on the home front above and beyond the call of duty. Linda, I owe you a lot of dish washing and I *will* pay you back.

I thank my parents for always encouraging but never pushing.

Abstract

Two systems for detecting the motion of a scene are described. For both, an image is projected directly onto an integrated circuit that contains photosensors and computing circuitry to extract the motion. The first system, which has been reported earlier, correlates the analog image with a digitized version of the image stored from the previous cycle. The chip reports the motion that corresponds to the maximum analog correlation value. This system represents an advance from previous designs but exhibits some shortcomings.

A second completely analog design surpasses the first. The mathematical foundation is derived and the CMOS circuits used in the implementation are given. Test results and characterization of the working chips are reported. The new motion detector is not clocked and exhibits collective behavior. The use of local information extensively avoids the correspondence problem. The system can be thought of as a Hopfield neural net with one important extension—input driven synapses. The motion detector also meshes nicely with the existing computational vision work. Extensions to handle more complex motions are proposed. The suitability of the motion extraction algorithm as a biological vision model is explored.

Contents

List of Figures

Chapter 1

Introduction

Future machines that interact flexibly with their environment must process raw sensory data and extract meaningful information from them. Vision is a valuable means of gathering a variety of information about the external environment. The extraction of motion in the visual field, although only a small part of vision processing, provides biological systems with knowledge about their surroundings. In addition to providing signals useful in tracking moving objects, visual motion can give clues about an objects extent and distance away.

Although a reliable motion detection sub-system, such as the one described in this thesis, may become an integral part of future machine vision systems, an integrated motion detector by itself has immediate application. A graphical pointing device, the mouse, reports its motion to the host computer. Existing mouse designs use either mechanical or optical means of detecting their motion over a fixed surface. Existing optical mouse systems require a specific surface pattern for reliable operation. The goal of building an optical mouse with a relaxed restriction on the surface pattern serves as a concrete goal for the design and implementation of a motion detector.

1.1 Readers Road Map

This thesis describes two approaches to the theory and implementation of integrated systems that report the uniform motion of a visual scene. Both implementations are VLSI integrated circuits that include an on-chip photosensor array, and report the motion of an image focused directly on the chip (Figure 1.1). Both systems contain integrated photosensors to sense the image and have closely coupled custom circuits to perform computation and data extraction on chip.

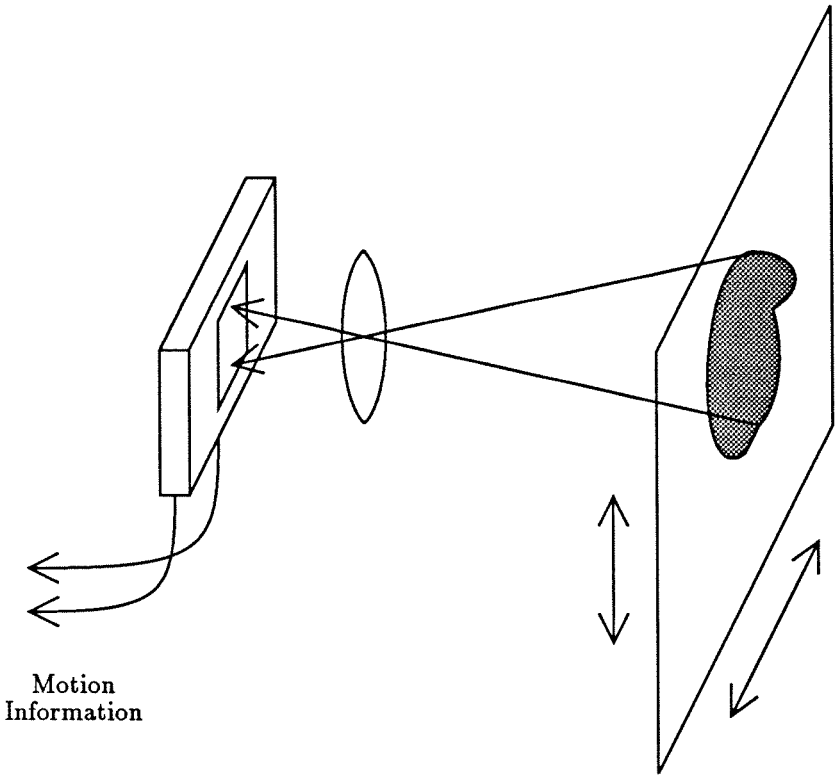


Figure 1.1: A scene is projected directly onto an integrated circuit by means of a lens. A custom chip senses the intensities of the moving image, performs a computation to extract the movement and reports the motion off-chip.

The first design, described in Chapter 2, utilizes a clocked photosensor similar to Lyon's [10]. On each cycle of the self-timed system, the image is quantized and stored digitally. The analog image is correlated with the digital image stored from the previous cycle. The maximum analog correlation is found and the corresponding motion reported. The correlating motion detector fabrication and test results are reported. This sensor system represents an advance over previous integrated sensors in the wide range of input images for which it successfully reports motion. Its shortcomings are that it cannot fully utilize the information present in the image due to its coarse quantization and the global nature of its imaging and comparing.

The second generation of integrated optical motion detector emerged from the goal of using local analog image intensity information as much as possible to extract image motion. This design combines a new high performance photosensor with analog computation elements and uses a novel approach to extracting velocity information from a uniformly moving image. The new motion sensor has a number of features that address the shortcomings of previous designs:

- The chip uses a continuous, non-clocked analog photosensor that has been demonstrated to operate over more than four orders of magnitude of light intensity [12].
- The design makes use of information in analog light intensity variations in the image. Sharp edges can be utilized but are not required.
- Local image gradients are utilized extensively. Our sensor avoids the problem of dealing with global gradients that plagues some sensors. The locality property also means that global notions such as object boundaries are not needed.
- The analog nature of our sensor and computation circuitry prevents the information loss inherent in thresholding or digitization and thus increases the use of the information that exists in a moving image.

Chapter 3 presents the physical motivation and mathematical basis for the one-dimensional version of the velocity calculation and proposes a suitable method to implement it. Chapter 4 extends the theory to handle the ambiguities of two-dimensional motion and introduces an architecture to perform the motion extraction computation. A simple analog mechanical model corresponds to this computation circuitry. Chapter 5 describes some of the CMOS circuits used in the implementation of the two-dimensional detector array and shows the characterization of these circuits individually. Chapter 6 presents test results for the working integrated motion system and extends the motion equations for good behavior even for images with

no information. Test data demonstrates that parameter variations within a chip are significant. The effects of these variations on the motion detector are analyzed.

Chapter 7 discusses the relation of the optical motion detector to other fields of research. An important AI problem, the "correspondence problem," is an artifact of the sampling process and is avoided completely by both generations of motion sensors. The state space energy function for the motion detector is developed and compared with that of Hopfield neural nets [6]. Extensions of the motion sensor architecture to more complex motions than just rigid translation are shown. These extensions mesh well with ideas from the computational vision field of Artificial Intelligence. The suitability of the motion detector as a model for biological vision modeling is discussed.

Chapter 2

A Correlating Motion Detector[1]

Here we describe an optical motion detector that uses integrated light sensors and analog and digital processing on the same chip. An image of an arbitrary scene or working surface is sensed by an array of photodiodes, stored, and correlated with the image taken on the next cycle. The position of maximum correlation indicates the relative motion of the image during the time between samples. This peak is detected using mutual inhibition and is converted to digital signals that go off chip to indicate motion. This single chip motion detector has applications in optical mouse systems and other optical tracking systems. It relaxes certain limitations of present devices which require a special operating surface. Other potential uses are in automated vision systems and robotics. This motion detector could be used, for example, to track parts moving down an assembly line. We have built a one-dimensional motion detector and demonstrated it in the laboratory.

2.1 Introduction to the Correlating Sensor

The *mouse* is a popular two-dimensional graphical input device for computers. Older mechanical mouse designs are being replaced by newer designs that use optics instead of moving parts to detect motion. These optical mice improve the reliability and decrease the intermittent action so common with mechanical mice. To date all the optical mouse designs must be moved over a special surface pattern in order to sense their motion properly. One commercially available optical mouse uses a metal plate with orthogonal grid lines. The lines in one direction reflect infrared and the ones in

[1]The bulk of this chapter has been previously published [22].

the other direction reflect visible light. Sensors within that mouse that are sensitive to only one color of light can detect motion in two directions independently. Lyon developed an innovative optical mouse design [10] that integrated sensors onto the same chip with the processing. His design requires a working surface consisting of a hexagonal grid of light dots on a dark background. Part of the motivation for our work is to relax the requirements on the working surface of the optical mouse. Our goal is to make an optical motion detector general enough to allow it to work on a wide variety of surfaces, like those commonly found on desk tops, thus eliminating the need for a special working surface.

2.2 How it Works

The optical motion detector consists of a single chip and a lens to project an image onto the chip. Figure 2.1 shows the functional block diagram of the chip. It consists of an array of photodiodes for detecting the light pattern, a storage array for the image, circuitry to compute the correlation between the stored image and the current one, decision circuitry to determine where the correlation is greatest, and a self-timed controller to sequence the entire system. Included is a test register that can electrically simulate optical images, allowing the chip to be tested for fabrication defects in a conventional non-optical setting.

2.3 Image Detection

The optical transducers are photodiodes patterned closely after those described by Lyon. In this nMOS process, a region of diffusion forms a diode with the grounded substrate. See Figure 2.2. Light striking the circuit side of the chip forms electron-hole pairs that create a leakage photocurrent through the reverse biased diode. In operation, the diodes are charged up by enhancement mode pull-ups which then shut off leaving them isolated. The diodes then discharge due to the photocurrent at a rate proportional to the intensity of the light striking the diffusion region. An array of these sensors starts out with all diodes charged. Each sensor discharges at a rate determined by the intensity of the image at that point. Eventually all sensors are discharged. The Lyon optical mouse uses mutual inhibition at this image level to detect a fixed set of "wired in" images. His set of images was carefully chosen by analyzing the possible positions of his hexagonal grid of dots relative to the orthogonal grid of sensors. Since this method is not capable of imaging a general pattern, we have pursued a more general approach.

Charge patterns that reflect interesting properties of the image occur somewhere

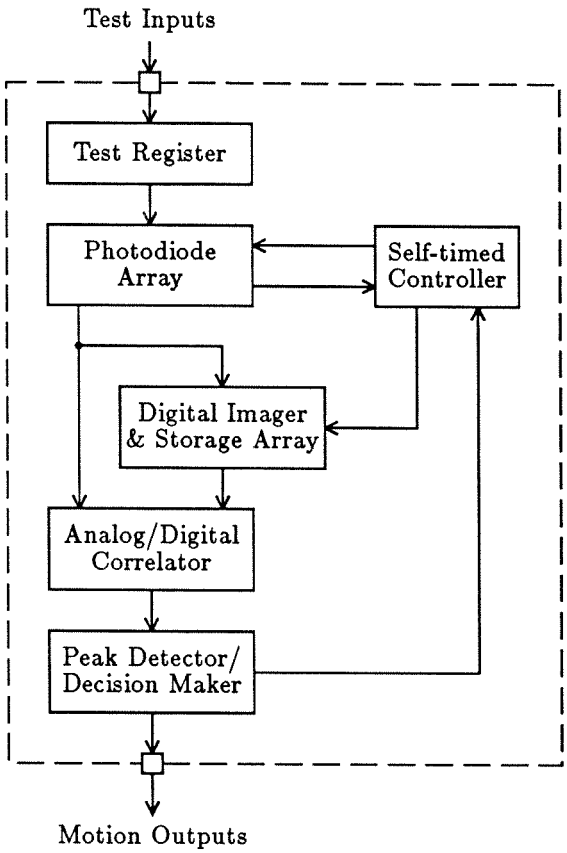


Figure 2.1: Block diagram of the optical motion detector chip.

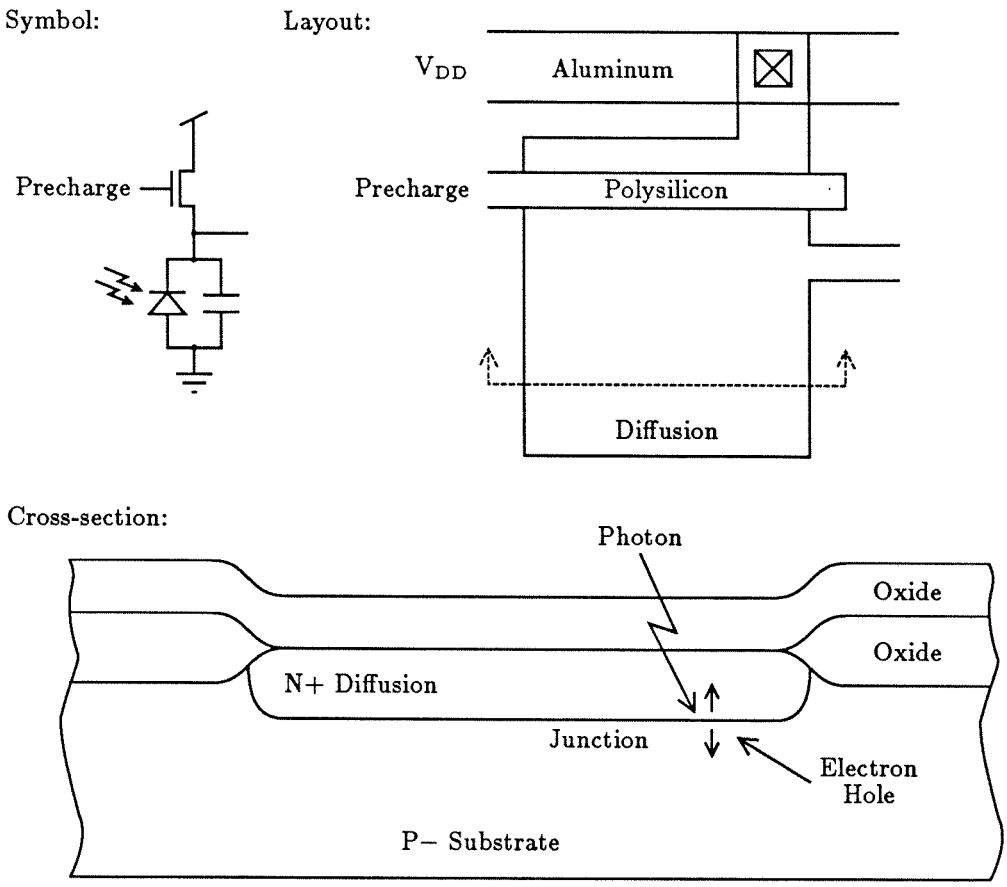


Figure 2.2: An nMOS photodiode used as a light sensor.

between the time when all the sensors are high and when all of them are low. We chose to sample and store all sensors at once, at a time when half of them are below a threshold and half of them are above. The circuitry to accomplish this operation is shown in Figure 2.3. There is a global Half-Down line with a single pull-up and a pull-down transistor for each photosensor. When the sensors are all high, the pull-down transistors are all on. The Half-Down line is low, with each of the sensors contributing to the pull-down current. As each photodiode discharges and passes the threshold of the transistor, the transistor turns off, subtracting its current from the total pull-down current. When the total current decreases far enough, the Half-Down line goes high. This threshold level is chosen by the width to length ratio of pull-up and pull-down devices so thresholding occurs when about half of the photodiodes are down. The rise of the Half-Down line triggers the latches to end their sample of the falling sensor value and via positive feedback turn it into a restored digital value. The digital image in these latches, one bit per sensor, is later compared with the next image. Section 2.12 discusses improvements to this circuit.

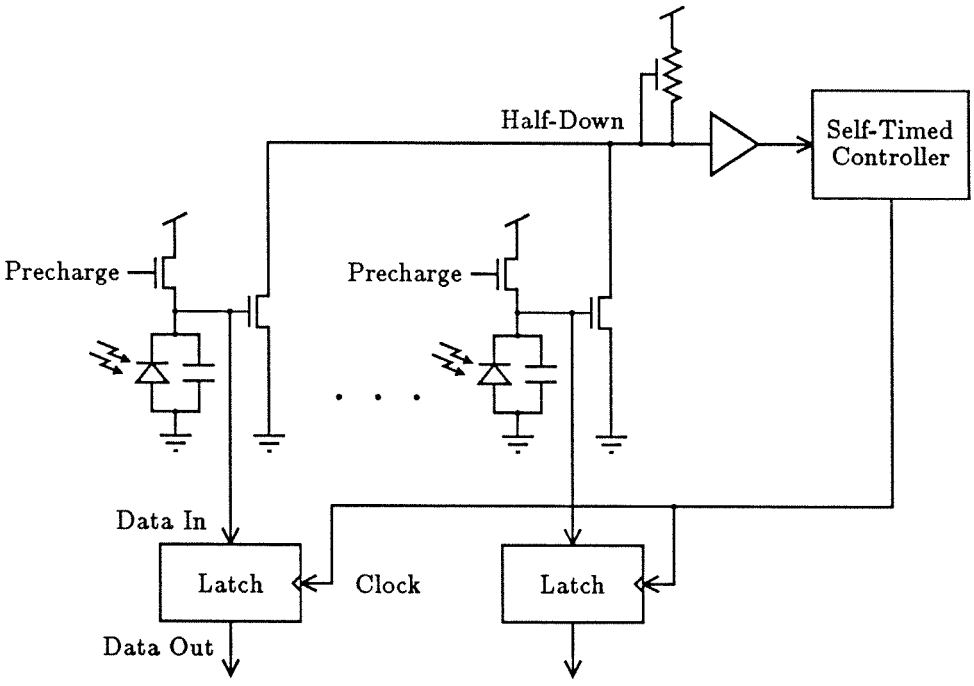


Figure 2.3: Digital imager and storage array.

2.4 Detecting Motion

Given two consecutive time samples of a one bit image, the task of motion detection becomes a comparison of the two images. Under the assumption that the object in view has changed relatively little, the images should be nearly the same except for a translation that corresponds to the motion. One method of detecting that motion is to shift one image past the other and at each position of the shift, count the number of bits that match. The position where this tally is the greatest will indicate how much the image has moved between samples. The comparisons could be implemented with an exclusive NOR gate and the addition with a current summing network.

This comparison and counting process corresponds to the one-dimensional correlation function. The continuous version is given by the formula:

$$C(s) = \int_{-\infty}^{\infty} I_0(x) I_1(x+s) dx.$$

I_0 is the image at time = 0 and I_1 is the next image sample taken at time = 1. The correlation of I_0 and I_1 is $C(x)$, where x is the amount one image is shifted relative to the other. $C(x)$ is maximum at the shift amount, x, that corresponds to the distance moved between samples I_0 and I_1.

The discrete approximation to the correlation function is given by:

$$C(s) = \sum_n I_0(n) I_1(n+s).$$

If the motion detector can operate fast enough to guarantee that the fastest motion never moves the image more than one sensor width (pixel) between two consecutive time samples, then this computation need only be performed within a one pixel neighborhood. Only three values need to be computed and compared for the one-dimensional case: $C(s)$ where $s = -1, 0$ and 1. These values correspond to the image having moved left by one pixel, right by one pixel or not having moved.

The logic diagram for the correlator is shown in Figure 2.4. Each of the three required correlation values are calculated by performing a multiplication of each of the old image values with the corresponding new image value and summing the results. The only difference between the three calculations is that the match between old image and new image values is shifted.

2.5 Analog versus Digital

When the analog voltage on the light sensor was quantized to one bit for storage, much of the light level information was lost. This information was sacrificed for the

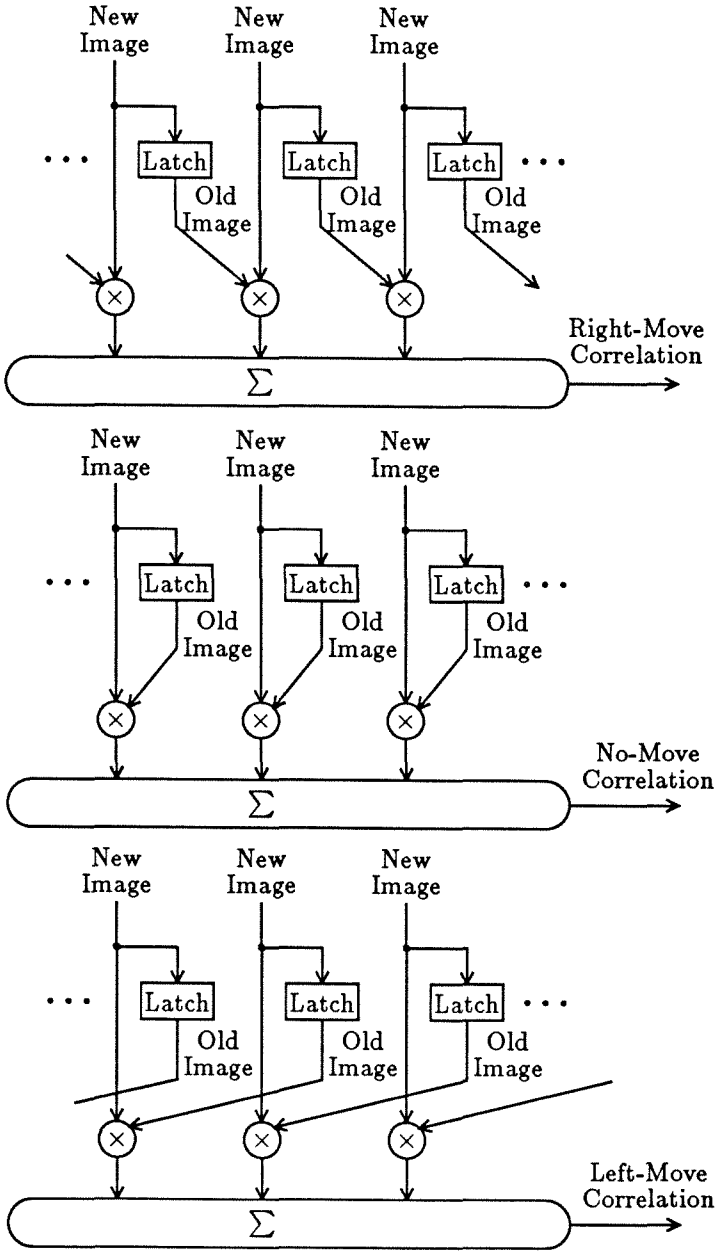


Figure 2.4: Logic diagram for computing three values of the one-dimensional correlation function.

ability to have long term storage of the image. In the correlation, we could have digitized two consecutive images and performed the correlation on them. Instead we chose to retain as much of the analog information as possible and use it in the correlation computation. The correlation is performed between the previous image, stored as 1-bit digital values, and the current image which is analog and develops in time from an all-high state toward an all-low state. The correlation values are represented as analog voltages that develop during the cycle as the photodiodes discharge.

2.6 Two Ways to do Multiplication

Exclusive NOR gates could be used to do the multiplications in the correlation computation. For this case, the individual correlation values within a cycle start out at an intermediate value when the photodiodes are all high, rise to a peak in the middle of the cycle and return to an intermediate value when all the diodes are discharged. To determine which correlation was the greatest would require first finding the peak voltages of each time-varying correlation voltage and then comparing them. Determining when all the peaks have passed so that the self-timed cycle can start over is a hard analog circuit problem, especially when the circuits must work with light hitting them and must work over several orders of magnitude of speed range.

AND gates can perform the multiplications *and* provide a monotonic time behavior. Correlation values using AND multiplication start at an intermediate value when the photodiodes are all charged and develop in time to a zero value when the sensors are all low. This monotonicity allows the comparison circuitry to be much simpler. Now comparing the correlation values is just seeing which of them goes to zero first. The end of the cycle is much easier to find also. The cycle ends when the first correlation value goes to zero. Circuits to detect this final condition are much simpler than circuits to detect the passage of a peak.

It is interesting to note that for motion detection either XNOR gates or AND gates can be used to achieve the same results. The XNOR function is equivalent to multiplication for digital levels assigned the values of 1 and -1, while AND gates perform multiplication for values of 0 and 1. The difference between these two ranges is the simple transformation:

$$I_{\mathrm{XNOR}} = 2\, I_{\mathrm{AND}} - 1.$$

Substituting this transformation into the correlation equation and simplifying yields the result:

$$C_{\mathrm{XNOR}}(x) = 4\, C_{\mathrm{AND}}(x) + \mathit{constant}.$$

This result shows that for the correlation computation, the difference between using XNOR and AND gates is a simple scaling and translation of the resulting values. Since we are interested only in the correlation values relative to each other, either type of multiplier will do just as well. It may seem that the XNOR case has a built-in gain factor of four over the AND case, but in either case our circuit implementation would scale the results into the same range; an analog voltage between the power and ground supply rails. For simplicity and the monotonicity reason given above, an AND structure implements the multiplications.

The circuitry to perform one of the three correlations is shown in Figure 2.5. The complement of the stored one-bit image is multiplied by the value of the new image by a pair of series pull-down transistors that perform the AND function. Instead of a correlation, the use of the inverted stored bit in the multiplication results in an anti-correlation—an output value that decreases with a better match instead of increasing. The series transistor pairs perform the required one-bit multiplication by sinking current when both the old stored image was low (inverted latch output is high) and the new image input is high. When the anti-correlation current is high, the correlation voltage is low. A global correlation line connecting the pull-downs performs the current summing. For the one-dimensional motion detector there are three correlation lines. On one of these lines, the current level indicates the strength of the correlation on the image moved left by one pixel. Another indicated the strength of "moved right" and the last gave "unmoved". The currents or voltage on these three lines must be compared to determine which of the three possibilities has occurred.

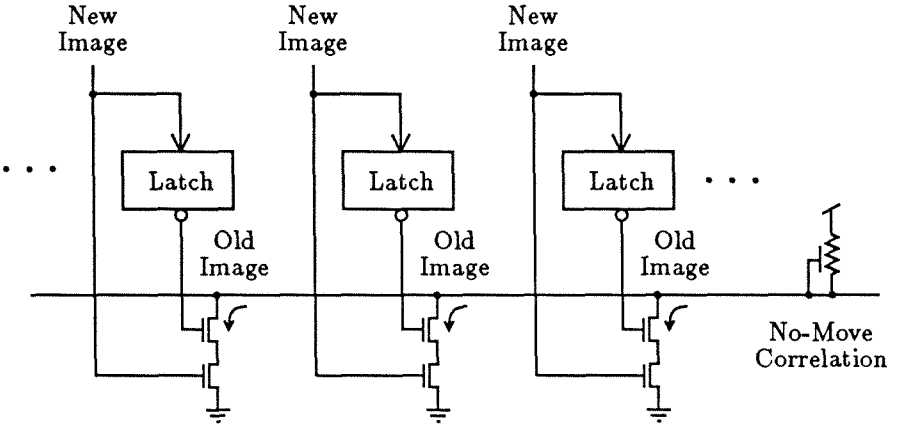


Figure 2.5: Circuitry to compute one of the three correlation values.

The Right-Move, Left-Move and No-Move correlation lines start at a high current, low voltage state and evolve toward a low current, high voltage state. The line with the greatest correlation will go high before the other two. We will make the comparison on this condition.

2.7 Making the Comparison

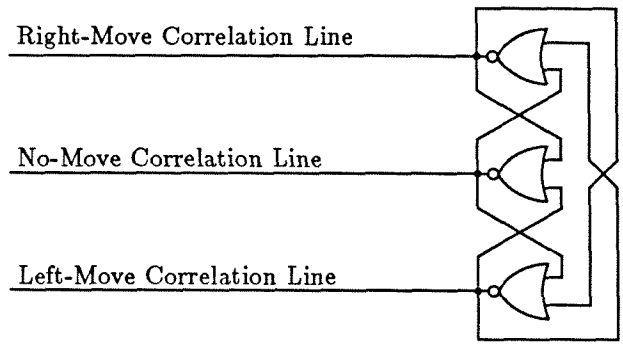
The comparison is done using mutual inhibition. The idea of mutual inhibition comes from neurobiology. Some sets of neurons are connected such that the firing of any one of them prevents or inhibits the others from firing. This connection arrangement allows at most one neuron to fire. If two or more neurons in such a set are stimulated at nearly the same time, there will be a race to determine which will fire first and inhibit the others.

In the integrated circuit, mutual inhibition occurs between the correlation lines. Each of the three correlation lines has a rising voltage that is in a race with the other two. The winner of the race is the one to go high first. As each line goes high it inhibits the rise of the other two, pulling them back down. In this way the final winner is never ambiguous because the only possible final states of the system are those with one line all the way high (the winner) and the other two low (being fully inhibited by the winner). If two or more lines are rising at nearly the same rates, the time required for a winner to be chosen is unbounded. The circuit is then in a metastable condition [11]. In that sense the mutual inhibition circuit may be viewed as a three-way arbiter [19].

The decision circuit is shown in Figure 2.6. Mutual inhibition is implemented by a three-way NOR flip-flop that starts out in the balanced or "illegal" state of all low. As the three lines are allowed to rise by the correlation circuitry, one of them will rise high enough to begin pulling down the other two. The final state of the peak detector will be two lines low and one line high. The high line indicates which direction the image has moved (or that it hasn't moved). When this circuit falls into one of these stable states, it has "decided" which of the three correlation values was greatest.

The gates driven by the three outputs of the decision circuit must have high enough thresholds so they are not falsely triggered by the lines rising to their metastable levels. The cross coupling of the NOR gates guarantees that at most one line will rise past the metastable voltage all the way up. Figure 2.7 shows two plots from a SPICE simulation of the decision tri-flop circuit. In the first, the correlation value for Right-Move is 1.0% higher than that of No-Move and Left-Move. Here the Right line has no trouble winning the race and going all the way high.

Logic Diagram:



Circuit:

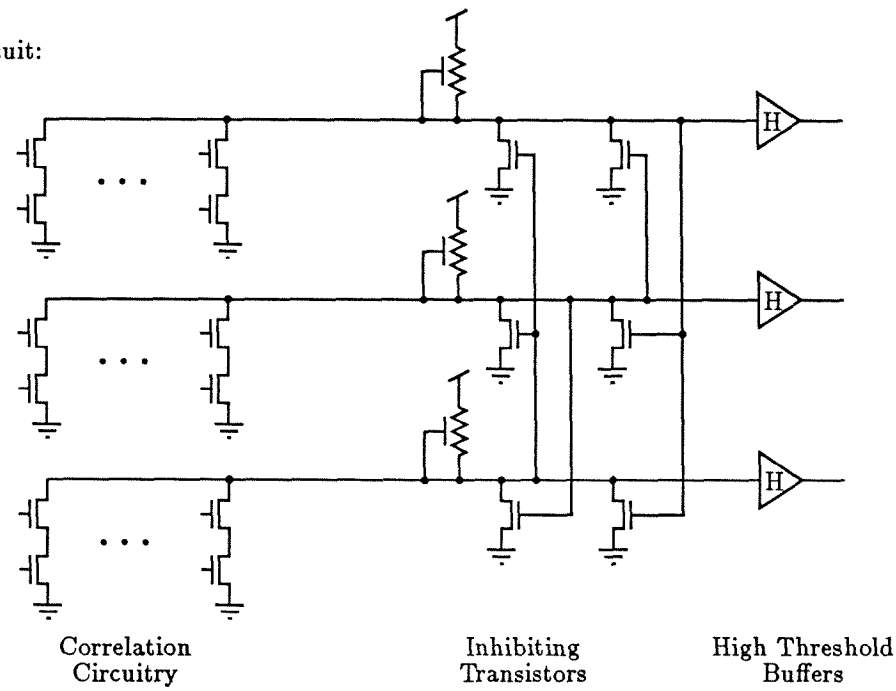


Figure 2.6: Logic diagram and circuit to implement mutual inhibition to make a decision.

In the second plot, the correlations are only 0.05% different. Here, both lines rise to the point that their mutual inhibition prevents them from rising further. The simulation shows that they hang for more than 50 μsec near the metastable point before finally one wins out and goes all the way high. These simulations do not take into account thermal noise or device parameter variations.

The three buffered outputs of the decision circuitry are outputs of the chip. They indicate detected movement by pulsing high from the time the decision is made until the next cycle is begun. These signals can be further encoded on chip for other motion encoding schemes.

2.8 The Minimum Velocity Problem

During a cycle there are two independent processes going on. The latched image from the last cycle is being used during the correlation-comparison process, and the current image is being latched for use next time. Since there is no guarantee that the computation-decision process will finish with the last image before the current image needs to be latched, there must be a two-level latch. The first level latch samples the light sensor and brings it to a digital level. The second level holds the previous sample for the correlation computation. The image is transferred from the first level to the second after a decision has been made by the tri-flop circuitry.

If the image moves by less than half a pixel between samples, the greatest correlation will always be on the unmoved line. If the second level latch always contains the previous sample, continuous motion at speeds less than one-half pixel/sample will never indicate a motion. For each cycle, the best image match will always be for the unmoved position. This occurrence is clearly a problem since the maximum speed of the image is only one and one-half pixels/sample (for a motion detector that only calculates correlation in a one pixel neighborhood). A good motion detector should have no minimum velocity, especially not one so close to its maximum velocity. We solved this problem by keeping the old image in the second-stage latch if an unmoved condition occurs. The only time the new image is moved into the register used for comparison with successive images is after a movement is detected (Figure 2.8). This technique reduces the minimum velocity of the motion detector to zero.

Conditionally loading the second-stage latch introduces a potential initialization problem. If the second-stage latches happen to power up with all values low or all high, the system may become stuck—either not able to load the second-stage latch from the image or not able to cycle at all. The present version of the chip has an external reset line that can force selected internal state to release the system from

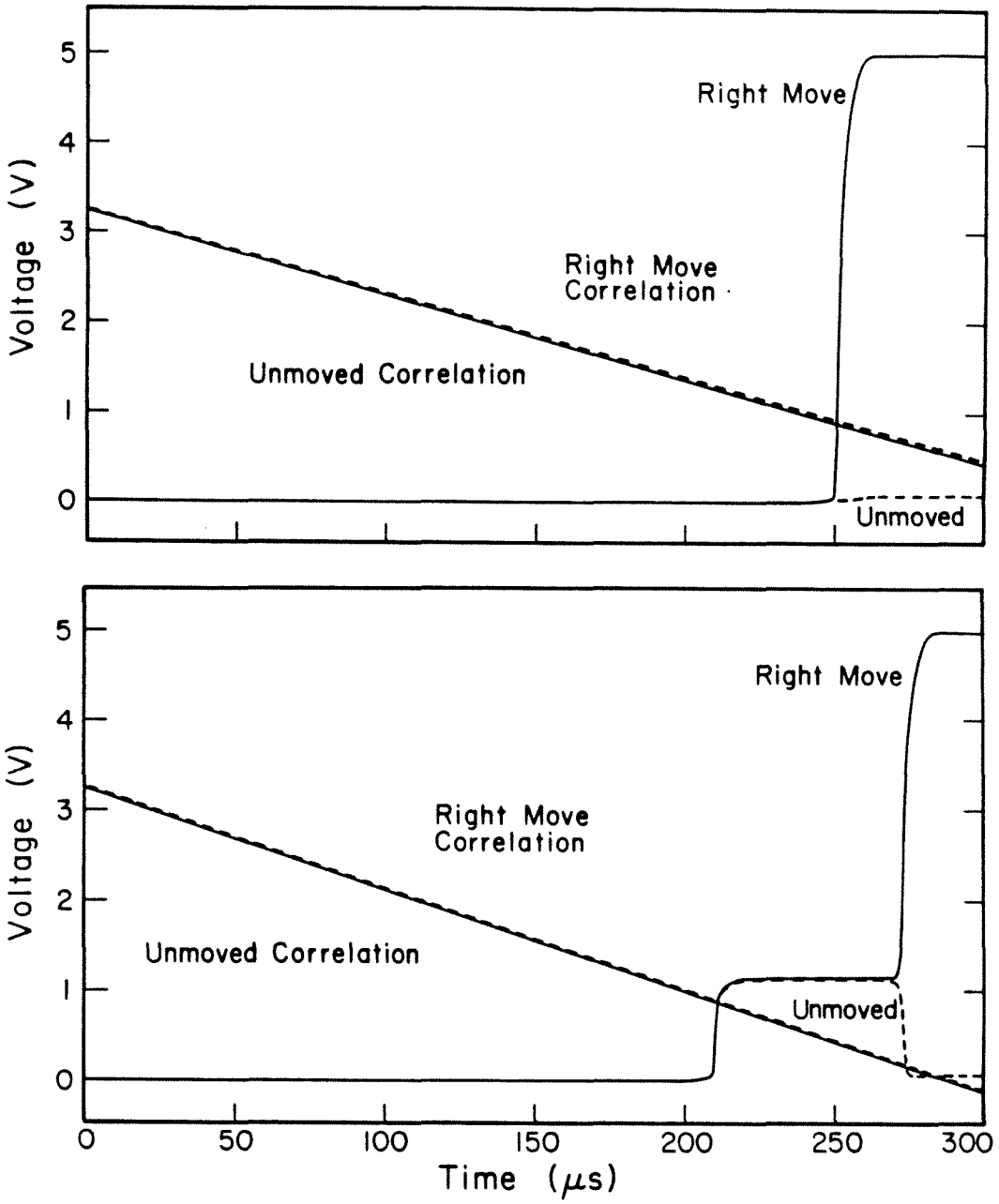


Figure 2.7: SPICE plots for the mutual inhibition decision circuit. Each of the two falling correlation lines in each plot represents the voltage on the gate of single transistor used to model the collection of correlation transistors. These voltages decrease steadily with time due to the discharge of the capacitance by the collective photocurrents associated with each of the correlations shown. In the top plot, the difference between the two lines is 1% and for the bottom 0.05%.

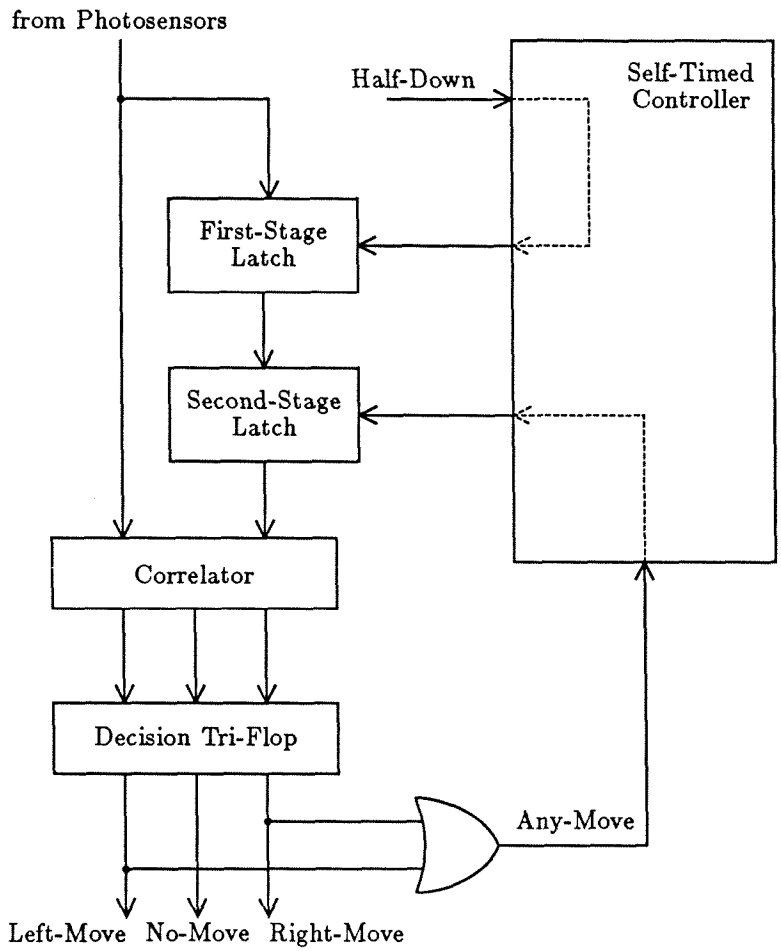


Figure 2.8: Conditionally loading the stored image into the second-stage latch array.

either the all-high or all-low conditions. During normal operation, the Half-Down circuit insures that every latched image contains some high and some low values. An alternative method of releasing a stuck system is to provide circuitry to internally detect the all-high or all-low condition and provide reset accordingly.

2.9 Cycling the Detector

A cycle in the detector consists of initializing the photodiodes, latching the new image into the first-stage latch, computing the correlation, making the decision, conditionally transferring the new image into the second-stage latch and then starting over again.

Since the cycle is self-timed, some circuitry is needed to detect when the photodiodes have all reached their high precharged level. Figure 2.9 shows how a distributed NOR gate and high threshold inverters are used to generate the required Ready signal. If any of the diodes are below the threshold, the Ready line will be low. When all photosensors are pulled high enough, the Ready line goes high.

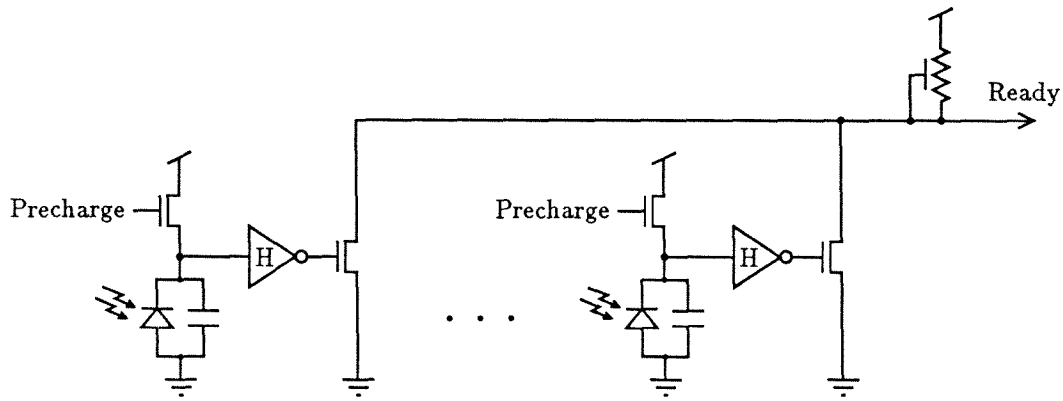


Figure 2.9: Generating the Ready signal that goes high when all the photodiodes are fully charged.

A Petri net of the flow of events in the motion detector is shown in Figure 2.10. Note that the Half-Down and Ready transitions are not truly independent. The Ready line always goes low before the Half-Down and goes high after the Half-

Down. There is an extra pathway, shown by dotted lines, that serves to synchronize the first-stage and second-stage latches. This link guarantees that the image data is transferred to the second-stage latch before the first-stage latch is cleared in preparation for the next cycle.

2.10 The First Prototype Chip

A one-dimensional motion detector chip was designed using the Caltech Stick design tools. It was submitted in March 1983 for fabrication to MOSIS, the ARPA community silicon foundry [2]. The fabrication process was a single polysilicon, single metal nMOS process with buried contacts and 4 μm minimum device size ($\lambda = 2\,\mu$m). The chip was 5711×1734 μm and contained sixteen photodiodes in a linear array. Each photosensor was a diffusion rectangle 200×400 μm. The chips came packaged in 40 pin packages with cavity covers that could easily be removed to project images onto the chip.

Figure 2.11 is a photograph of the chip. There are 16 identical sensor cells, the controller, pads and wiring. Each sensor cell consists of the large photodiode, a two-stage one-bit latch, one bit of the test register and part of the distributed circuitry to produce the Half-Down and Ready signals and perform the correlation. The controller cell on the right contains the decision tri-flop and the self-timed controller logic.

2.11 Maximum Image Speed

Figure 2.12 shows the operating frequency of the optical chip as a function of light level. The solid line in the graph represents a first order approximation assuming that the discharge rate of the photodiodes is the major delay in the cycle of operation and that the rate is proportional to the intensity of the incident light. The operating frequency, f, is given by the equation:

$$f = \frac{1}{t} = kL,$$

where t is the time for one cycle, L is the illuminance of the incident light, and k is a constant. Experimental results show that this linear approximation is reasonable over almost three orders of magnitude of light level variation. The theoretical proportionality constant for the frequency/light level relation, k, can be calculated from the unit area capacitance of the diffusion layer and from a conversion between

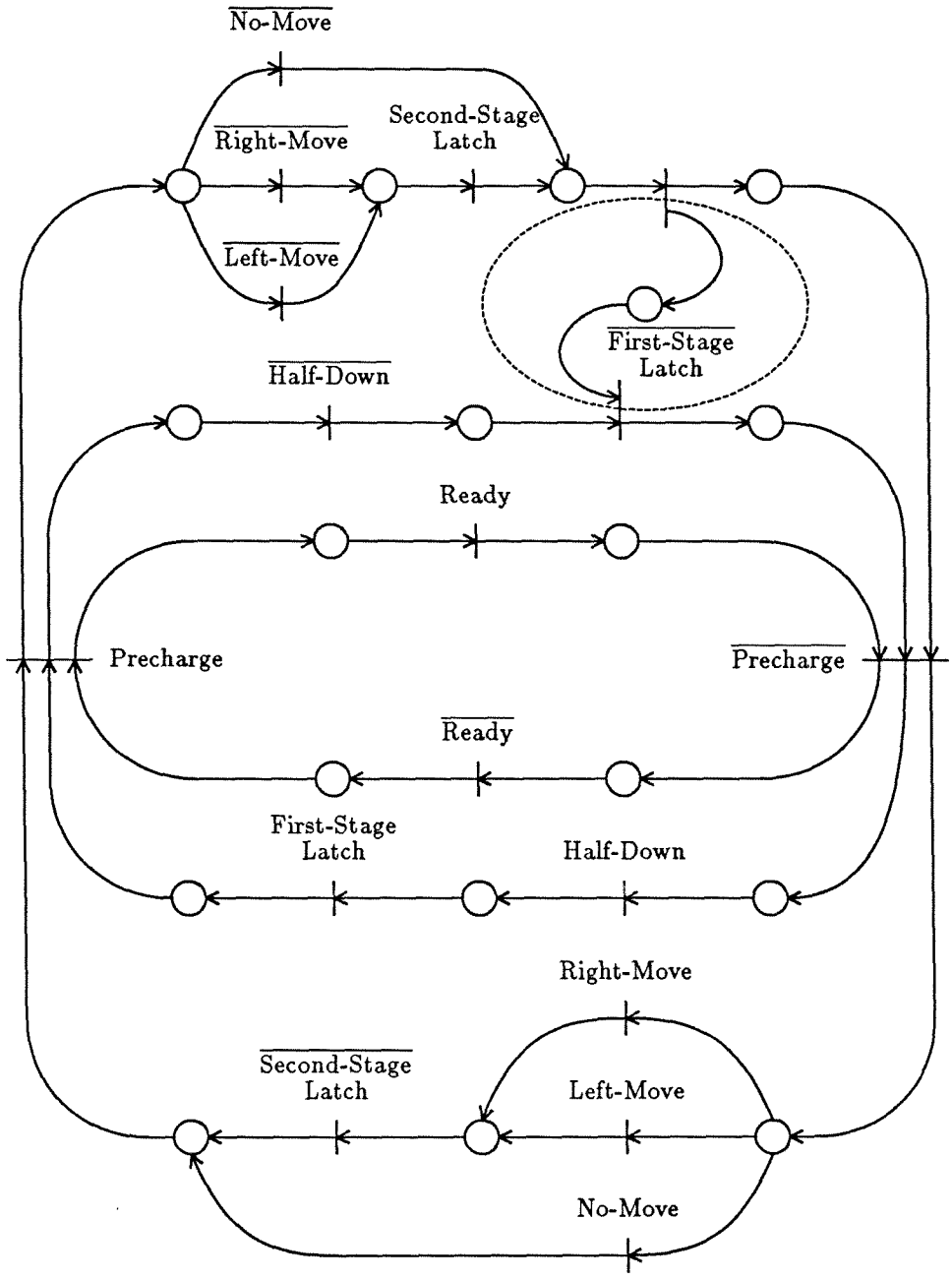


Figure 2.10: A Petri net representation of the motion detector cycle.

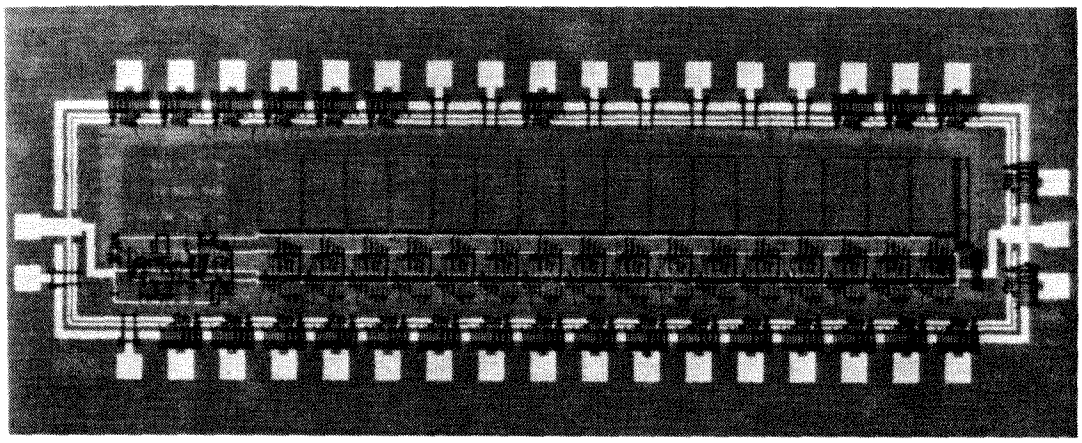


Figure 2.11: Photograph of the prototype optical motion detector chip.

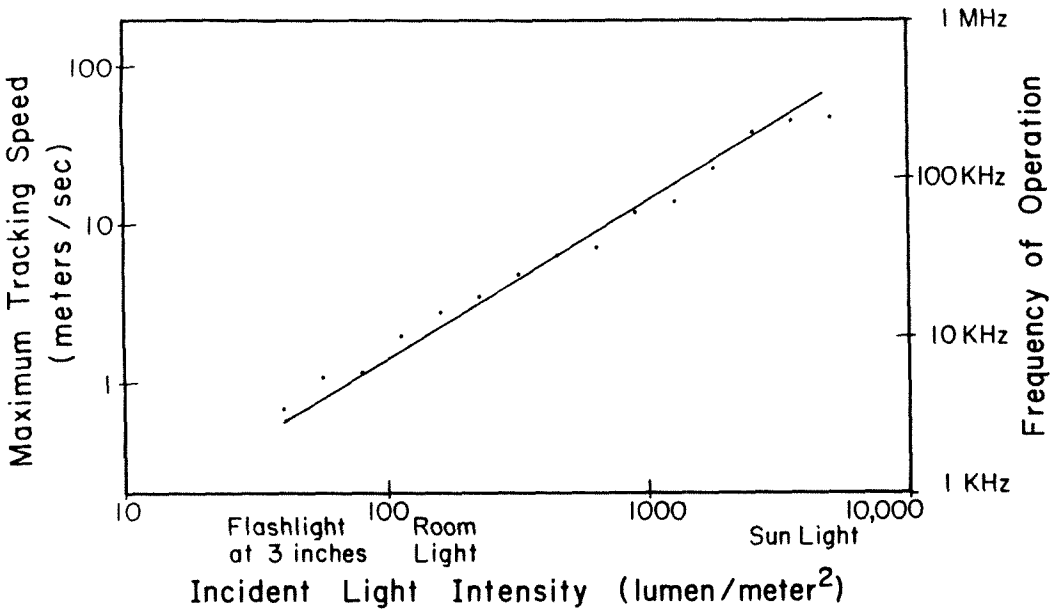


Figure 2.12: Plot of operating frequency of the optical motion detector as a function of incident light level (right vertical axis). The dots are from measurements of the working chip. The straight line is an ideal approximation. The maximum tracking speed (left vertical axis) is for the sensor spacing of 200 μm and assumes an optical magnification of 1x and a maximum tracking speed of one pixel/cycle (200 μm/cycle).

incident light and photocurrent. Depending on the exact assumptions made, k is easily within a factor of 2 of the experimental results.

If the image moves on the chip between image samples farther than the neighborhood of correlation calculation, the motion detector will not accurately report the motion. In our chip the correlation is calculated only to the nearest neighbor, about the width of one photosensor or about 200 μm. At room light levels, the free running cycle frequency of 10 KHz corresponds to an image velocity of about 2.0 meters/second. The magnification provided by the lens is the ratio of the size of the image on the chip surface to the real object. In a mouse application, a magnification of 1x gives a resolution of about 100 points/inch and a maximum mouse speed of 2.0 meters/second. The magnification can be changed by moving the positions of the lens and chip relative to the object. Adjusting the magnification effectively trades off resolution for maximum tracking speed (Figure 2.13). A greater magnification increases the resolution because one pixel distance on the chip now corresponds to a smaller distance on the object. For the same reason, the maximum speed of the object decreases for a constant image speed. If the same light level per unit area on the object is maintained, the maximum speed will decrease even further. This decrease is the result of a lower intensity image due to the magnification. To keep the speed-resolution product the same, the image intensity must remain constant. A greater magnification will therefore require the object to be illuminated with the same amount of light concentrated on a smaller area, an effect that can be obtained with a simple condensing lens on the light source.

2.12 Analog Control Inputs

Several circuits, such as the one that generates the Half-Down signal, depend critically on their switching threshold for correct operation. The threshold can be set by carefully choosing the relative device sizes of the enhancement pull-downs and depletion pull-ups. In this first prototype we wanted the flexibility of choosing the thresholds *after* the chips were fabricated. One reason was that we wanted to vary some of the thresholds to see how they affected performance. In other cases where we knew the desired thresholds, we were reluctant to risk the success of the chip on simulation results using device parameters from past MOSIS fabrication runs. Figure 2.14 shows a circuit that allows the threshold of a gate to be varied by using an off-chip potentiometer. The current mirror configuration limits the current flow in each pull-down to a value proportional to the control current set by the potentiometer. The proportionality is set by the relative device sizes of the two current mirror transistors. Decreasing the control current decreases the current in each pull-down

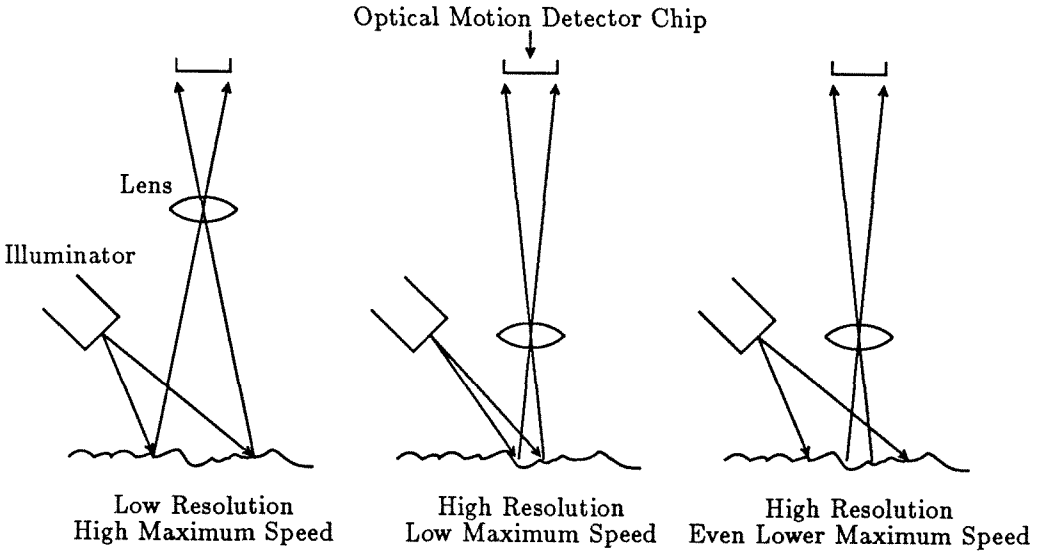


Figure 2.13: The effects of optical magnification on resolution and maximum speed.

so more of them need to be on to overcome the pull-up. With the Half-Down control input we can vary the threshold smoothly from a level where all zeros are latched to a level where all ones are latched. A similar analog control was built into the test register enable line so that test patterns could be made to discharge the photodiodes at slower rates.

Another approach to building the Half-Down circuit is to rely on the matching of like transistors across the chip instead of the relatively poor matching of the pull-up to pull-down transistor ratios between fabrication runs. The simple Half-Down circuit of Figure 2.3 will produce an output voltage when half of the inputs are high and half are low. This voltage represents the desired Half-Down threshold and will vary from run to run. We can build a Half-Down reference circuit on the same chip with identically sized pull-up and pull-down transistors. If we tie half of the inputs to the reference circuit permanently high and the other half low, the output of the circuit will be the Half-Down threshold. An on-chip differential amplifier, connected to the outputs of the Half-Down circuit and the reference, can compare the two values and determine when the Half-Down line has crossed the threshold. Although not yet implemented, methods such as this that use on-chip transistor matching instead of run-to-run parameter matching show promise for

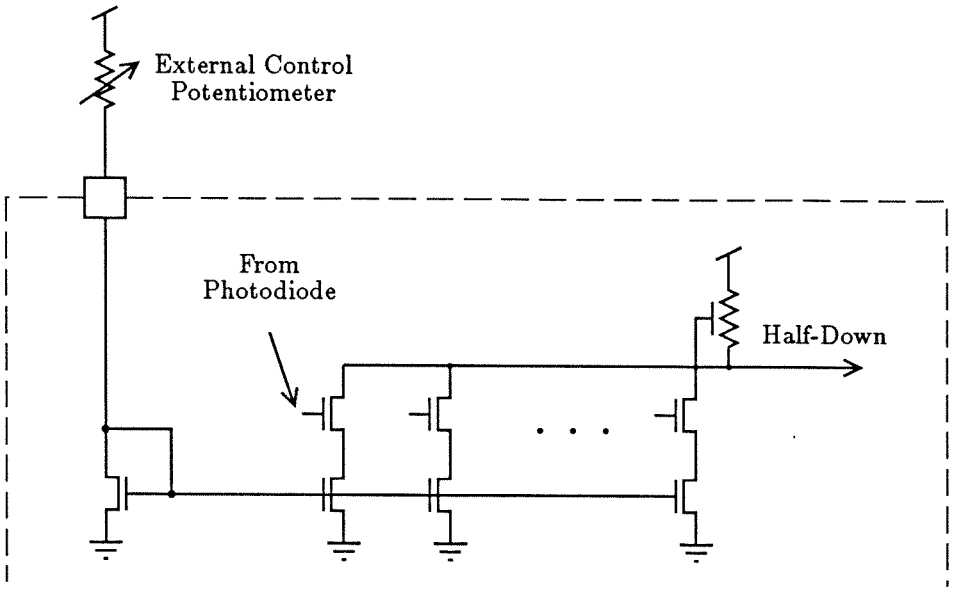


Figure 2.14: External analog control of the Half-Down threshold.

future versions of correlating motion sensor.

2.13 State of the Design

The one-dimensional version of the chip operated over a wide range of intensities as reported in Figure 2.12 above. The optimal setting of the Half-Down control line varied with light level. A single setting of the control would result in a good image over a small range of light level variation. As the light level moved from range, the image would become all 1's or all 0's. This problem would be quite serious if the system was required to operate over a wide range of light levels without human adjustment. If the problem was caused by light hitting part of the circuits other than the photosensors, than a layout change to cover the sensitive circuit may solve the problem. Otherwise, the chip could be made self regulating by feeding back the number of 1's (or 0's) in the latched image to increase or decrease the analog control current. Although I never implemented a solution to this problem, I feel it could be solved.

A more serious problem is the global nature of the thresholding of latched images. The resulting image reflects only the intensity variations that occur near the threshold intensity. Any edges in the image that occur in a relatively bright or dark

area will not be captured in the image. Global gradients in intensity such as those produced by a non-uniform light source, will make images that change from light to dark over the chip. The inability of the chip to make use of local information away from the global threshold is a serious limitation of this design. The motion detector described in the next chapter avoids this limitation.

2.14 Summary of Correlating Sensor

The integration of sensors and computing structures onto the same chip is a natural way to capitalize on the parallel nature of many problems by avoiding any sequential representation or communication of information until after it has been processed at the lowest level. Circuit tricks and ideas from biology such as mutual inhibition can be used profitably in the design of these sensing/computing chips.

Some goals are met:

- The chip operates over a wide range of light levels.
- A wide variety of patterns, regular and irregular can be used as an operating surface.
- Operating frequency is sufficient to allow reasonable velocities while meeting the requirement of motion less than one pixel per cycle.
- Local computations allow a compact efficient design. The correspondence problem is avoided.
- The design is extensible to any size array.

Some problems and shortcomings remain:

- The stored image is digitized to binary values and so information is lost.
- The global threshold results in a sensitivity to intensity gradients. Although the correlation computation is local, the imaging does not utilize local intensity variations.

Chapter 3

A One-Dimensional Analog Motion Detector

Analysis of a simple one-dimensional analog image shows some of the measurable quantities present and the computation necessary to extract velocity from the moving image. An architecture is presented for combining local velocity calculations into a reliable global result.

3.1 A New Analog Design

To overcome some of the limitations of the previous motion detectors [1,10,22] I have designed a new motion detector with the following properties:

- It still operates over many orders of magnitude of light intensity.
- It utilizes the analog values of intensity to compute velocity continuously.
- It utilizes local information extensively.
- It does not depend on a global clocking scheme.
- The longest communication wires required are those within a region over which we are assuming uniform velocity (at first the whole chip).

3.2 One-Dimensional Motion Detection

Figure 3.1 shows a plot of the intensity of some incident image as a function of distance along the one spatial dimension of interest for two moments in time. We wish to exploit purely local properties of the image to determine motion. An observer

fixed at x_0 and resting on the intensity terrain will be moved up and down as the intensity curve (the scene) moves by him. His rate of movement up or down ($\frac{\partial I}{\partial t}$) and the slope of the terrain ($\frac{\partial I}{\partial x}$) can be determined by local observations. From these two values, the observer can infer the velocity of the intensity curve past him.

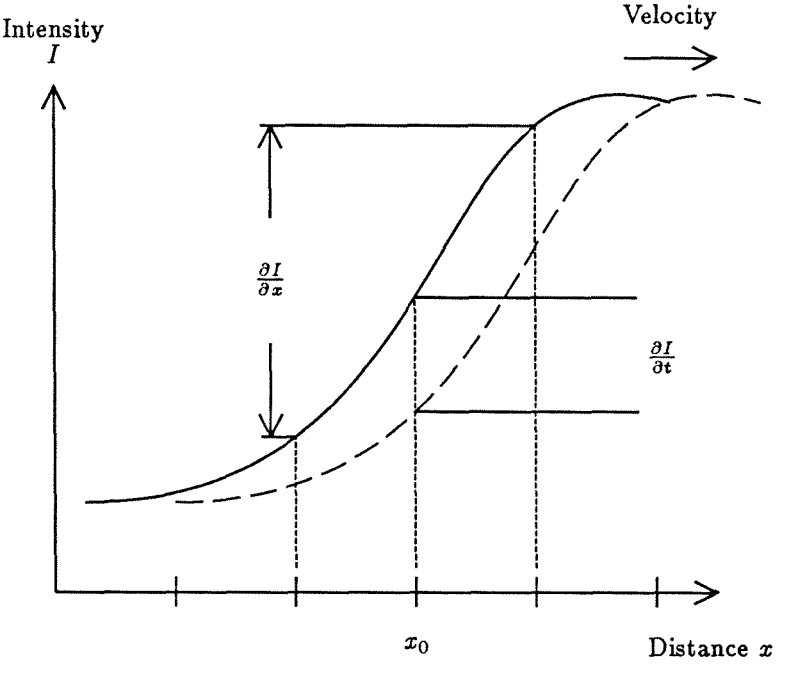


Figure 3.1: A one-dimensional image for motion detection.

The observers are implemented by an array of sensors described in detail in Section 5.2. Each sensor produces a voltage monotonically related to the light intensity incident upon it. At point x_0 the spatial derivative can be approximated locally by taking the difference between the intensities of the neighboring sensor on either side and dividing by the fixed spacing between them. This approximation is quite good if the sensors are spaced close enough together relative to the highest spatial frequency in the image.

A local circuit can also determine the time rate of change of the local intensity by taking the time derivative of the intensity signal. By knowing the local intensity gradient (slope) and knowing how fast the intensity is changing, the velocity of the intensity profile can be calculated. The equation for the tangent line to the intensity curve at x_0 is:

$$I(x,t) = m[(x - x_0) - vt] + I_0,$$

where v is the velocity of the image and m is the slope of the line.

The time and space derivatives of the intensity are:

$$\begin{aligned} \frac{\partial I}{\partial x} &= m, \\ \frac{\partial I}{\partial t} &= -mv = -\frac{\partial I}{\partial x} v. \end{aligned}$$

Therefore:

$$v = -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}}. \tag{3.1}$$

Intuitively, this makes sense because the definition of velocity is just $\frac{dx}{dt}$. The minus sign arises from the motion of the observers coordinate system with respect to the image which is opposite the motion of the image with respect to the observer. (The x in Equation 3.1 is not the same x as in $v = \frac{dx}{dt}$.)

The relationship of Equation 3.1 allows us to take two quantities that are dependent on the image, its two derivatives, and calculate the velocity which is independent of the image. A hypothetical velocity detector is shown in Figure 3.2.

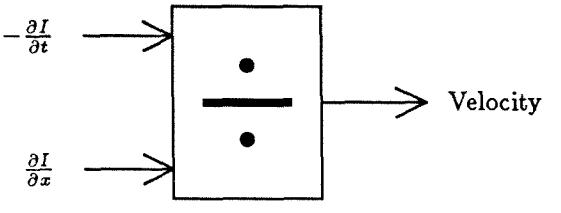


Figure 3.2: A hypothetical divider used to calculate velocity.

There are some problems with the above formulation that prevent us from implementing it directly with analog circuits.

- A four-quadrant analog divider is hard to build. Since both derivatives can be positive or negative, we need a circuit that operates in all four quadrants of the input space.
- We can't divide by zero. When the spatial derivative is zero (e.g. on the peak of a hill) we can't infer anything about velocity. Mathematically our equation is undefined.
- For small $\frac{\partial I}{\partial x}$, any errors (e.g. noise) in our circuit would produce a large error in the resulting velocity.

The effect of errors in some positions can be reduced by combining information from many positions. An array of local velocity sensors should be capable of combining velocity information in a way that increases reliability. Each locally computed velocity should contribute to the resulting velocity in the following ways:

- A greater number of elements should yield a more reliable result.
- A local result that has a lower confidence level should contribute to the more global result with a lower weighting. A local lack of information (when $\frac{\partial I}{\partial x}$ is zero) should not contribute at all.
- The global result should not be affected drastically by the error or failure of a small number of sensors relative to the total number of contributors.

3.3 A Simple Aggregation Scheme—Averaging

A simple method of aggregating many local signals is to average them. The equation for a uniform average is:

$$\bar{v} = \frac{1}{n}\sum_{j=1}^{n} v_j = \frac{1}{n}\sum_{j=1}^{n} -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}}.$$

If we had a divider, this computation could be implemented as shown in Figure 3.3 where the average circuit is a simple current summing wire and the scaling by the constant $\frac{1}{n}$ is ignored.

The equation for the average assumes an equal weighting of all local velocities and doesn't solve any of our problems. An average with general weights is given by:

$$\bar{v} = \frac{\sum_{j=1}^{n} v_j w_j}{\sum_{j=1}^{n} w_j} = \frac{\sum_{j=1}^{n} -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}} w_j}{\sum_{j=1}^{n} w_j} \tag{3.2}$$

and can be implemented with an additional multiplier per cell (Figure 3.3).

The individual local velocity measurements should be weighted according to the confidence of their contributions. The greater in magnitude that the spatial derivative is, the greater our confidence in it. Sharper edges are counted more heavily. We therefore consider two possible weightings and consider their effect on the implementation. First we will try:

$$w_j = |\frac{\partial I}{\partial x}|,$$

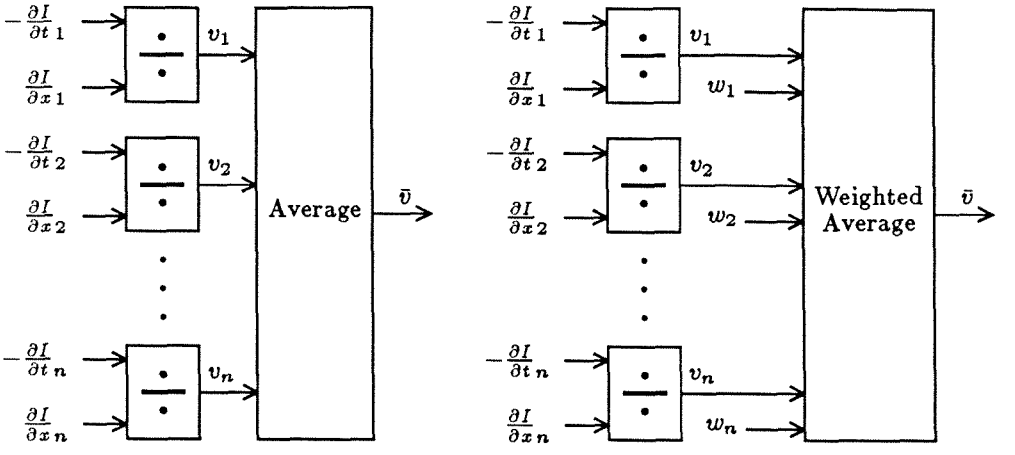


Figure 3.3: Aggregating local velocity information into a global quantity using simple averaging and weighted averaging.

which results in:

$$\bar{v} = \frac{\sum_{j=1}^{n} v_j |\frac{\partial I}{\partial x}|}{\sum_{j=1}^{n} |\frac{\partial I}{\partial x}|} = \frac{\sum_{j=1}^{n} -\frac{\partial I}{\partial t}\,\mathrm{sign}(\frac{\partial I}{\partial x})}{\sum_{j=1}^{n} \frac{\partial I}{\partial x}\,\mathrm{sign}(\frac{\partial I}{\partial x})}.$$

The second choice to consider for the weight is:

$$w_j = (\frac{\partial I}{\partial x})^2,$$

where confidence goes up with the square of the spatial derivative. For this choice, sharp edges are weighted even more heavily than by the first choice of the weighting function. The global average velocity becomes:

$$\bar{v} = \frac{\sum_{j=1}^{n} v_j (\frac{\partial I}{\partial x})^2}{\sum_{j=1}^{n} (\frac{\partial I}{\partial x})^2} = \frac{\sum_{j=1}^{n} -\frac{\partial I}{\partial t}\frac{\partial I}{\partial x}}{\sum_{j=1}^{n} \frac{\partial I}{\partial x}\frac{\partial I}{\partial x}}.$$

Figure 3.4 shows an implementation of these equations.

This formulation has several effects on the implementation.

- The local division is not present anymore.

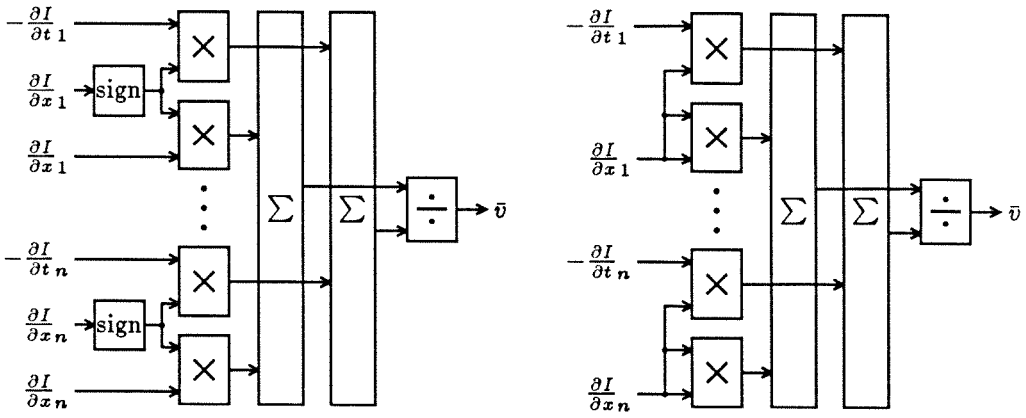


Figure 3.4: Block level implementation of velocity calculation for confidence weightings of $|\frac{\partial I}{\partial x}|$ and $(\frac{\partial I}{\partial x})^2$

- We need to accumulate two quantities globally, the numerator and the denominator, instead of the single quantity required by the previous formulation.
- We need to do a global division. Note that now the only time the denominator is zero is when all local $\frac{\partial I}{\partial x}$s are zero. This condition occurs only when there is no information in the entire image, a much less frequent and therefore less troublesome condition than individual $\frac{\partial I}{\partial x}$s equal to zero.
- The division that we do need is only a two-quadrant division because the denominator is always positive. This two-quadrant divider is much easier to build than a four-quadrant one.
- We need circuits to take the sign of an analog quantity and to implement analog four-quadrant multipliers.

A CMOS amplifier, described in Section 5.3, can be used to implement either the sign function or the analog multiplication. The amplifier behaves like a sign circuit in part of its operating range and like a multiplier in another part of the range.

The transfer curve of a typical amplifier is shown in Figure 3.5. The output voltage, V_{OUT}, is a smooth function of the input voltage, V_{IN}. For V_{IN} near zero, the transfer curve is nearly linear and can be approximated by:

$$V_{\text{OUT}} = AV_{\text{IN}},$$

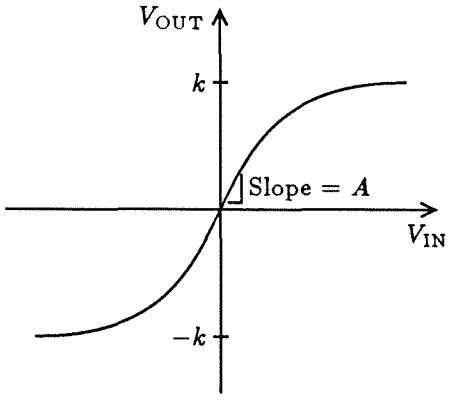


Figure 3.5: Transfer curve of a typical CMOS amplifier.

where A is the gain of the amplifier, the slope of the transfer curve near zero. For larger $|V_{IN}|$, the amplifier output limits and becomes relatively independent of V_{IN}. Here the transfer function is a good approximation to the sign function:

$$V_{OUT} = k \text{ sign}(V_{IN}),$$

where k is the limit of V_{OUT}.

Figure 3.6 shows the two ideal weighting functions and the real weighting function of the amplifier of Figure 3.5. The amplifier behaves like each of the ideal cases in different ranges of its operation.

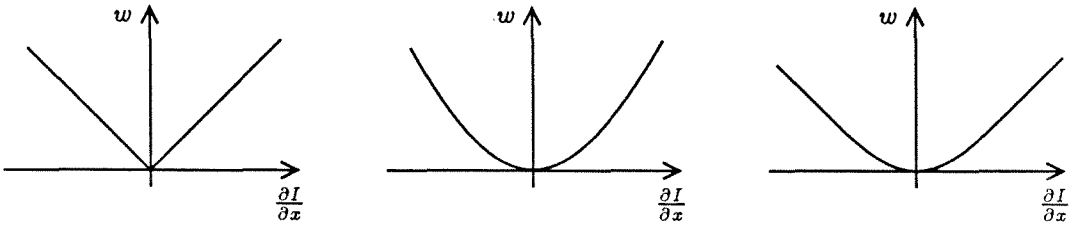


Figure 3.6: Weighting curves for $|\frac{\partial I}{\partial x}|$, $(\frac{\partial I}{\partial x})^2$ and the limiting amplifier.

The two choices for weights provide flexibility in the implementation. Circuits have different behaviors in different ranges of operation. We use circuits that at

times implement one choice of weights and at other times implement the other choice. At all times, they calculate a weighted average of velocity.

Both implementations are robust against out-of-range signals. If the sign function of Figure 3.4 is implemented as a high gain amplifier but has input signals so small that the output is not limited, the system will still work. The weighting of that input will change but the output will still be a weighted average velocity. Alternatively, if the implementation of the squared weighting is chosen and an input signal becomes so large that a multiplier limits, the weighting of that one input will be less than the squared weighting but the result is still a weighted average. If a multiplier limits, its saturation level must be independent of the other input so that the two multipliers in each cell saturate at the same time. The details of the multiplier circuit are given in Section 5.6. It turns out that this saturation property does hold.

The choice of weighting criterion, absolute value or square law, can vary from cell to cell in the same system as well as varying from system to system. An implementation of a system provided with a wide range of input signal amplitudes will have circuits operating in different ranges and therefore using different weights. The output is always a weighted average velocity.

After considering the implementation, the weighted average scheme has another important benefit:

- The system is robust against out-of-range values of $\frac{\partial I}{\partial x}$. Small values of $\frac{\partial I}{\partial x}$ and large ones that limit the circuits alter the weightings but the result is still average velocity.

Some shortcomings remain:

- A global circuit is required to produce the desired velocity quantity.
- Two global wires are needed to communicate the numerator and denominator of the global velocity.

The above method of finding the weighted average accumulates the numerator and denominator separately and then divides. I call this organization the numerator-denominator method (or formulation) to distinguish it from methods that follow.

3.4 A Two-Quadrant Divider

A local divider in each cell could reduce the number of global wires and eliminate the need for a global circuit. This section examines the difficulty of implementing a four-quadrant divider.

Building a multiplier feedback circuit yields a divider with correct operation for two quadrants of input. Often a forward transfer function can be implemented by using its inverse in the negative feedback path of a high gain amplifier. The four-quadrant multiplier is well behaved and can be implemented as described in Chapter 5. The multiplier can be used in the feedback path as shown in Figure 3.7. The output of the summation represents the error of the output, ϵ:

$$\epsilon = \frac{\partial I}{\partial t} + v\frac{\partial I}{\partial x}.$$

If the error is zero then:

$$0 = \frac{\partial I}{\partial t} + v\frac{\partial I}{\partial x}.$$

This equation is just a rearrangement of the velocity equation (Equation 3.1). For zero error, the output of the circuit is velocity. For non-zero error, the idea is to greatly amplify the error and supply this signal to the output in a direction that will reduce the error. We can see that for positive values of $\frac{\partial I}{\partial x}$ the feedback is negative and moves the output to a point of near-zero error. In this case, the output is velocity and the circuit really does implement a two-quadrant divider. As soon as $\frac{\partial I}{\partial x}$ becomes negative, the feedback around the loop becomes positive. Any error produces an output that increases the error instead of reducing it and the circuit will race off to its limits. In these two quadrants of input space, the chip behaves more like a latch than a divider.

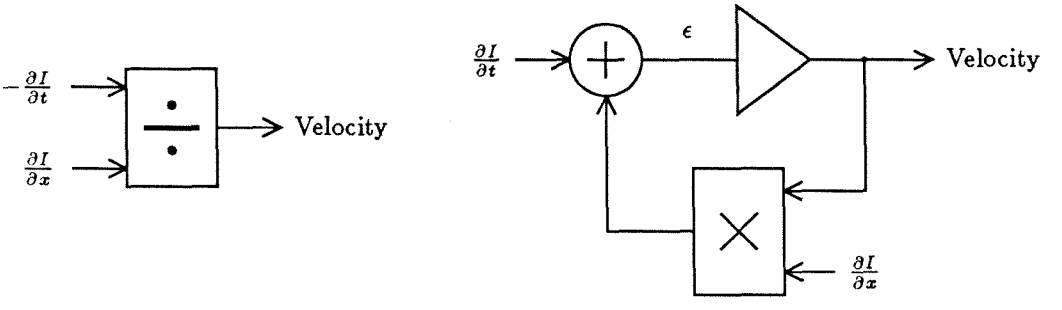


Figure 3.7: Hypothetical four-quadrant divider and an attempt to build it that operates successfully in two quadrants.

In Section 3.6 I introduce an extension of this design that performs a weighted division but does so for all four quadrants, for both positive and negative values of $\frac{\partial I}{\partial t}$ and $\frac{\partial I}{\partial x}$.

3.5 A Resistor Network for Weighted Averaging

A simple resistor network can be used to perform weighted averaging. Our previous implementation found the weighted average by accumulating the numerator and denominator of the average and dividing externally. The resistor network uses the much simpler structure of Figure 3.8 to achieve the same result.

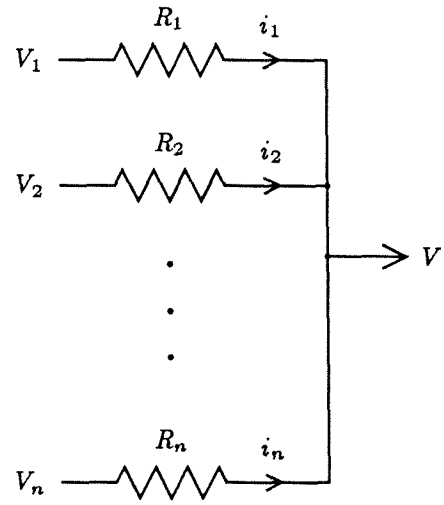


Figure 3.8: A resistor network that computes the weighted average of several input voltages. Each weight is the conductance g where $g = \frac{1}{R}$.

Intuitively, each of the resistors provides a connection that pulls the average voltage toward its corresponding input voltage. A smaller resistance means a stronger connection and a greater influence on the average. The currents through the resistors provide the means for changing the output voltage until it reaches equilibrium at the weighted average. At equilibrium, the total current contribution from all resistors is zero. Total current, i_{TOT}, is the sum of the currents through each of the resistors due to a voltage drop across them. Here we use i for current and V for voltage. These quantities must not be confused with light intensity, I, and velocity, v, used throughout:

$$0 = i_{\text{TOT}} = \sum_{j=1}^{n} i_j = \sum_{j=1}^{n} \frac{V_j - V}{R_j}.$$

Replacing the reciprocal of resistance, $\frac{1}{R_j}$, with conductance, g_j, and solving for V,

we get:

$$V = \frac{\sum_{j=1}^{n} V_j g_j}{\sum_{j=1}^{n} g_j}.$$

Comparing this equation to Equation 3.2 we see that the output voltage, V, is just the weighted average of the input voltages, V_j, where the weights are the conductances, g_j.

The average computation performed by a resistor network is an example of a very simple collective computation.

- The computation is disperse. There is no one central circuit performing the crucial computation.
- The computation is done in parallel. Every resistor current changes continuously without waiting for any other resistor.
- The computation is local. Each resistor responds only to the two voltages on its own nodes.
- The network scales well. Adding or removing resistors still results in a valid global average calculation. There are no scaling constants due to the number of inputs.

We wish to use the collective nature of a resistive network to compute the weighted average of the local velocities. If we had a local four-quadrant divider, we could use the architecture of Figure 3.9. Analog voltage represents velocity and $\frac{\partial I}{\partial x_j}$ controls the value of the resistances R_j such that $(\frac{\partial I}{\partial x_j})^2 = \frac{1}{R_j}$.

Although we cannot implement the four-quadrant divider, we can build a circuit that behaves like the combination of divider and weighting resistor surrounded by the dashed box in Figure 3.9. The next section describes that implementation.

3.6 A Four-Quadrant Weighted Divider

In this section we present a block level diagram for a four-quadrant weighted divider with inherent averaging capabilities. The new design addresses the shortcomings of the numerator-denominator method of weighted averaging and has these properties:

- One circuit, by itself, is a four-quadrant divider.
- The divide-by-zero problem is avoided by an output drive with variable strength.

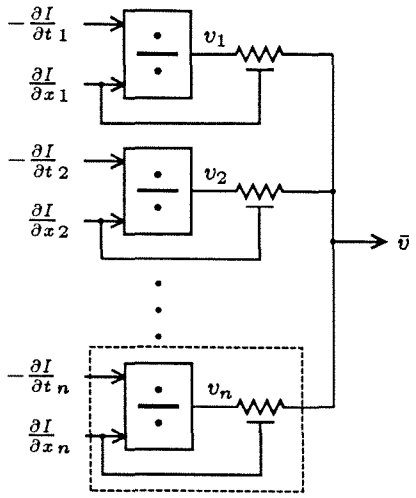


Figure 3.9: Using a resistor network and hypothetical local dividers to compute average velocity.

- A group of these circuits generates a weighted average of their dividends on their common output.
- No global circuits are needed.
- One global wire is needed.

To derive the circuit, we view the resistor as an element that computes an output current that it injects onto a global line. This current is a function of the value of the resistor, R_j, and of the voltage difference across it, $V_j - V$:

$$i_j = \frac{V_j - V}{R_j}.$$

For the velocity calculation, voltage represents velocity, $V = \bar{v}$ and $V_j = v_j$, and the conductance weighting goes as the square of $\frac{\partial I}{\partial x}$, so $\frac{1}{R_j} = (\frac{\partial I}{\partial x})^2$. Substituting, we have:

$$i_j = (v_j - \bar{v})(\frac{\partial I}{\partial x})^2.$$

The velocity, v_j, is in turn a function of the two derivatives, $\frac{\partial I}{\partial t}$ and $\frac{\partial I}{\partial x}$. Again substituting we have:

$$i_j = -\left(\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}} + \bar{v}\right)(\frac{\partial I}{\partial x})^2.$$

Multiplying through we get:

$$i_j = -\left(\frac{\partial I}{\partial t} + \bar{v}\frac{\partial I}{\partial x}\right)\frac{\partial I}{\partial x}. \tag{3.3}$$

This equation is important:

- It contains no division.
- No internal representation of infinity is necessary.
- The calculated current, i, is a function of the two measured inputs $\frac{\partial I}{\partial t}$ and $\frac{\partial I}{\partial x}$ that define local velocity and is a function of the weighted average, $\bar{v}$.

To find out what happened to infinity, we examine Equation 3.3 for the previously troublesome case of small values of $\frac{\partial I}{\partial x}$. As $\frac{\partial I}{\partial x}$ approaches zero, the output voltage of the hypothetical divider increases rapidly, but due to a resistance that increases at a faster rate, the current decreases. Since the weighted divider computes the current directly, and the choice of weighting function for the resistance eliminated the divide-by-zero in this computation, there is no need for any internal representation of infinity.

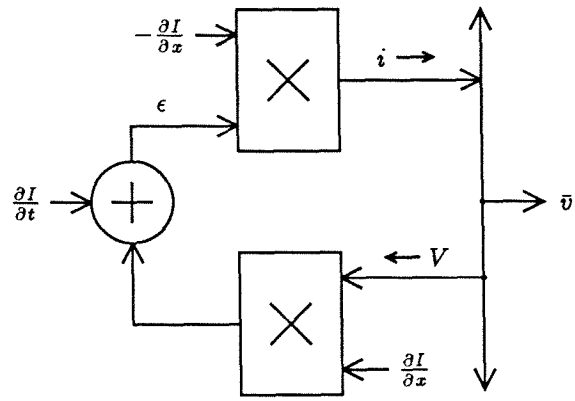


Figure 3.10: Circuit to implement the four-quadrant weighted divider.

The block diagram for a circuit that performs weighted averaging is shown in Figure 3.10. It directly implements Equation 3.3 It requires two analog four-quadrant multipliers just as the numerator-denominator method of Section 3.3 but requires only a single global line and no external divider.

	Numerator-Denominator Formulation	Resistor Network Formulation
Number of Multipliers per Cell	2	2
Sets of Global Wires	2	1
Requires External Divide Circuit	Yes	No

The weighted four-quadrant divider of Figure 3.10 is an extension of the two-quadrant feedback divider of Figure 3.7. Study of these two diagrams shows that they both have a four-quadrant multiplier in the feedback path and a summing node that computes the same error, ϵ:

$$\epsilon = \frac{\partial I}{\partial t} + v\frac{\partial I}{\partial x}.$$

For zero error, the output is the velocity as was the case for the two-quadrant divider. Any non-zero error is multiplied by $\frac{\partial I}{\partial x}$ according to Equation 3.3:

$$i = \left(\frac{\partial I}{\partial t} - v\frac{\partial I}{\partial x}\right)\frac{\partial I}{\partial x},$$

to generate a correction current that will move the output closer to the correct velocity. The additional multiplier in the four-quadrant circuit serves to scale the correction but more importantly flips the sign of the error term so that the correction is always in the right direction—the feedback is always positive. The high gain amplifier corresponds to the high voltage gain of the current from a high impedance current source, integrated by the capacitance of the global line to form a voltage.

Using a simple physical analogy, the correction generated by the circuit can be thought of as a force that pulls the global average velocity toward the locally derived velocity. To illustrate the simplicity of the necessary computation, the force, F, can be written in the form of Equation 3.3:

$$F = \left(\frac{\partial I}{\partial t} + \bar{v}\frac{\partial I}{\partial x}\right)\frac{\partial I}{\partial x}.$$

Rearranging to get the force as a function of velocity, we have:

$$F = \left(\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}} + \bar{v}\right)\left(\frac{\partial I}{\partial x}\right)^2 = (\bar{v} - v)C.$$

The force is proportional to the difference between the local velocity, v, and the global average $\bar{v}$ which gives rise to the global averaging property of the collection

of circuits working together. The velocity average moves until the net sum of forces on it becomes zero. The force is also proportional to $(\frac{\partial I}{\partial x})^2$ which weights the contributions of each cell according to the confidence in its local information, C.

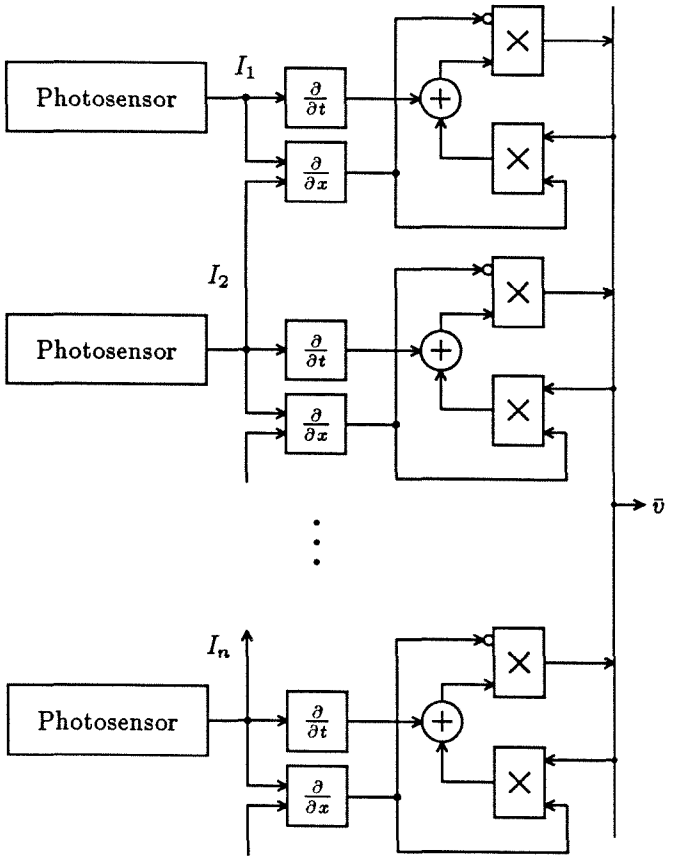


Figure 3.11: An array of photosensors and weighted dividers that collectively computes average velocity.

An array of cells, each with its photosensor and divider circuitry, form a simple collective system (Figure 3.11). Among the benefits offered by such a system are:

- Local information is used by each cell.
- Aggregation of local information is done in a way that increases the accuracy of the result.
- Cells that have no information ($\frac{\partial I}{\partial x} = 0$) behave nicely and don't contribute to the aggregated value.

- Analog intensity information can be used. This eliminates the information lost due to quantization.
- Operation is completely parallel. Each cell in the array continuously computes.
- There are no global circuits needed to post-process the data or to cycle the system.

3.7 Summary of One-Dimensional Motion Detection

Starting from the moving intensity curve of a one-dimensional image, we derived the local calculation to extract velocity of the image from locally measured derivatives of the image. An architecture modeled after a resistor net requires only simple computational elements locally yet performs the equivalent of calculating all the local velocities and finding their weighted average. A single global wire serves two functions (Figure 3.12). First, the wire transmits the result of the weighted average computation to each location where it is used in the local calculations. Second, the wire takes part in the global calculation by accumulating all the local corrections to the global weighted velocity.

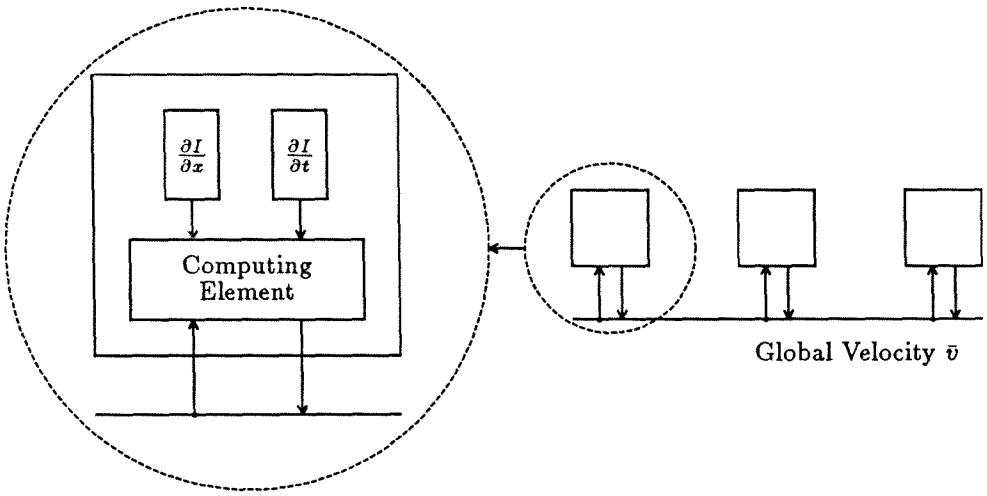


Figure 3.12: Architecture for the one-dimensional motion detector.

In the next chapter, this collective scheme will be extended to handle motion in two dimensions.

Chapter 4

A Two-Dimensional Analog Motion Detector

Generalizing the motion detection algorithm for velocity detection in two dimensions is not exactly straightforward. A problem arises from an inherent ambiguity between motions along the two axes. This ambiguity stems from a limited field of view such as the view through an aperture. The *aperture problem* is well known for binary-valued images.

Figure 4.1 shows the view through a rectangular aperture. A black-and-white image containing a single straight edge is moving with some velocity so that the position of the edge at a later time is shown by the dashed line. The velocity cannot be uniquely determined from these two snapshots. There is an infinite family of possible velocities as illustrated by the arrows. We can view the image velocity components v_x and v_y as the x and y coordinates in a plane we will call the *velocity plane*. In this plane, the actual velocity of the image defines a point. The family of possible image velocities define a line in velocity space. This line, as plotted in Figure 4.1, has the same orientation in velocity space as the edge does in physical space. To be consistent with the visual information from the local aperture the actual velocity point is constrained to lie on the line in velocity space. This line is known as a *constraint line.*

Please Note: Figure 4.1 illustrates the ambiguity problem but does *not* depict the operation of the system described below. The velocity detectors emerging from the analysis in this chapter:

- Represent intensity values continuously. The images we consider are *not* just black and white.
- Represent time continuously. There is *no* notion of snapshots of the image or

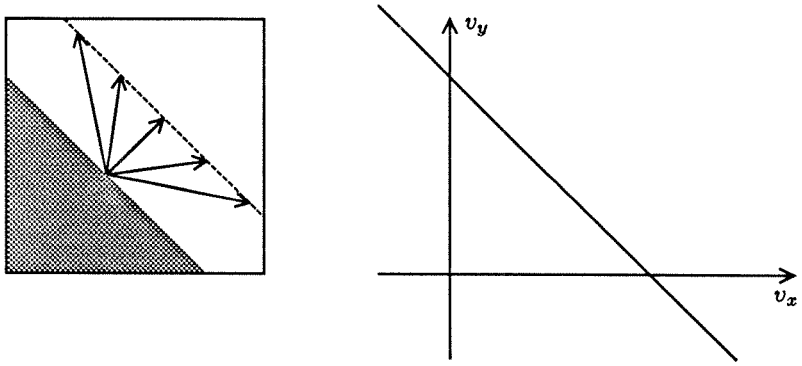


Figure 4.1: The aperture problem: Local information is not sufficient to uniquely determine two-dimensional velocity.

of clocking in these systems.

Using analog values for intensities and gradients doesn't eliminate the ambiguity problem. Figure 4.2 shows the intensity plot of an image that contains gray-scale information and varies smoothly in intensity throughout. A local observer on the intensity terrain cannot tell if his upward or downward movement, $\frac{\partial I}{\partial t}$, is due just to motion along the x-axis, just to motion along the y-axis, or to a combination of motions. Two of these possibilities are shown as arrows in Figure 4.2. The inherent ambiguity cannot be resolved by strictly local information.

Here we derive an expression that relates the intensity derivatives to the velocity. Following a route of analysis similar to the one-dimensional case, what was an intensity curve in one spatial variable is now a surface function of two variables.

The equation for the tangent line:

$$I(x,t) = \frac{\partial I}{\partial x}\Big((x - x_0) - v \cdot t\Big) + I_0$$

becomes a vector equation for a tangent plane:

$$I(\mathbf{x},t) = \nabla\mathbf{I} \cdot \Big((\mathbf{x} - \mathbf{x}_0) - \mathbf{v}t\Big) + I_0,$$

where $\mathbf{x}$ is the position vector $\langle x, y\rangle$, $\nabla\mathbf{I}$ is the two-dimensional gradient at the position $\mathbf{x}_0$, and the velocity becomes a two-dimensional quantity as well where $\mathbf{v} = \langle v_x, v_y\rangle$. The expanded form of this equation is:

$$I(x,y,t) = A\Big((x - x_0) - v_x t\Big) + B\Big((y - y_0) - v_y t\Big) + I_0,$$

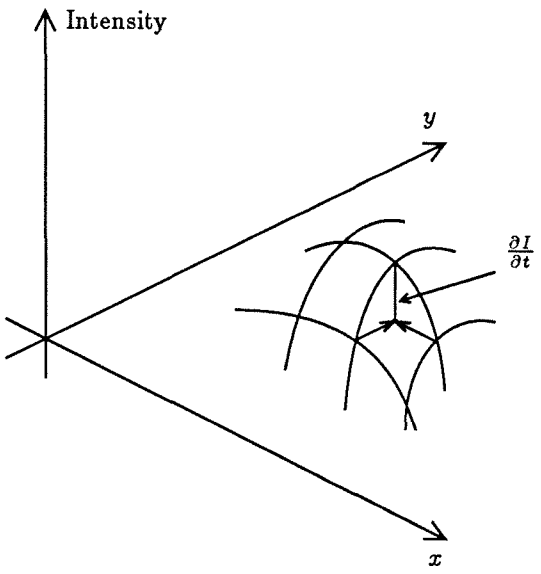


Figure 4.2: The intensity surface of a two-dimensional image.

where A and B are the spatial derivatives $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$, respectively, and $\nabla \mathbf{I} = \langle A, B \rangle = \langle \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \rangle$.

The equation that relates the three partial derivatives is:

$$\begin{aligned} \frac{\partial I}{\partial t} &= -\nabla \mathbf{I} \cdot \mathbf{v} \\ &= -\frac{\partial I}{\partial x} v_x - \frac{\partial I}{\partial y} v_y. \end{aligned} \tag{4.1}$$

We can see from this equation that knowing the three local derivatives of the intensity does not allow us to uniquely determine the velocity. There is an inherent ambiguity.

The local intensity derivatives do provide some useful information—they constrain the possible values of the x and y components of velocity just as we found for the aperture problem and black-and-white images.

Writing Equation 4.1 in the form of the line equation $Ax + By + C = 0$, we get:

$$\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} = 0.$$

Each local set of three derivatives defines a line in the velocity plane along which the actual velocity must lie. The slope of this constraint line is $-\frac{\partial I}{\partial x} / \frac{\partial I}{\partial y}$. If we view the gray-scale image as having a fuzzy "edge" with orientation perpendicular to the

intensity gradient, the constraint line has the same orientation in velocity space as the "edge" has in physical space. This is the same orientation as the constraint line of the black-and-white image.

The aperture problem for binary-valued images is just a special case of the general two-dimensional velocity ambiguity. Local images, gray-scale or black-and-white, can only provide a family of possible velocities. This set of velocities can be represented by the coefficients of the equation for the constraint line.

It is much easier to determine the constraint line if the analog information is retained. For gray scale images, the coefficients of the constraint line equation are just the three partial derivatives, $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$, and $\frac{\partial I}{\partial t}$, that can be locally measured. An image with continuous intensity values can be made into a black-and-white image by thresholding. To determine the orientation of the edge of the binary-valued image (and so its constraint line) is a more global problem of determining the boundary between black and white regions and fitting a line to the boundary. To determine the velocity constraint line, it is much easier to locally measure the coefficients than to throw away the information and then try to reconstruct it with a global process.

The ambiguity of a single local set of measurements can be resolved by using another set of local values from a nearby location. These values define another line in the velocity plane. The intersection of these two lines uniquely determines the actual velocity. This intersection of constraint lines is illustrated in Figure 4.3.

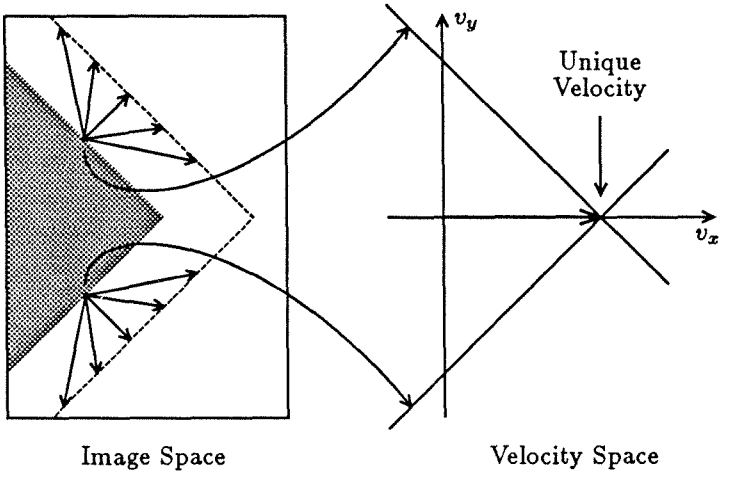


Figure 4.3: Uniquely determining velocity by the intersection of constraint lines.

4.1 Solving Simultaneous Constraints

In practice, to find the actual velocity, we will use constraint contributions from each site on the sensor array. Using a small number of sites, close together relative to an object size, results in a few constraining lines in the velocity plane that are nearly parallel. A small error in any of the derivatives or in the constraint solver can then result in a large error in the computed velocity. Errors will be kept to a minimum when two lines in the velocity plane cross at right angles. This intersection occurs when there are contributions from two sites on "edges" that are perpendicular. An edge in this case is used loosely to mean a perpendicular to the direction of greatest intensity change. Contributions from a large number of sites will then assure us of having pairs of orthogonal constraints for any reasonable image.

The barber pole illusion is a well known example where the orthogonality of constraint lines cannot be assured. In this illusion, the rotating cylinder produces a purely horizontal velocity. Our vision system erroneously reports "seeing" a vertical velocity. Images such as gratings and stripe patterns with intensity variations along only one axis cause this problem. All constraint lines are coincident so their intersection is not unique. It is not possible for man, beast, computer, or chip to disambiguate the motion of such a pattern.

In practice, there is no such thing as a perfect stripe pattern. The question then is a matter of degree. Our chip should reliably report the actual velocity unless the signals resulting from intensity variation along one axis lie below the noise level.

4.2 Constraint Solving Circuits

Our constraint solving circuit contains a set of global wires that distribute a best guess of velocity to all the individual constraint generating sites (Figure 4.4). Each locale performs some computation to check if the global velocity satisfies its constraint. If there is an error, circuitry within the local site then supplies a "force" that tends to move the global velocity in a direction to more closely satisfy the local constraint. The global velocity components are represented as analog voltages on the set of global wires. The correcting forces are currents that charge or discharge the global wires.

Finding the intersection of many lines is an over-constrained problem. Any errors will result in a region of intersection in which the real desired point most likely lies. To compute a most probable intersection point (velocity) requires us to know what types of errors to expect, to define "most probable" and select on the basis of that definition a forcing function that varies with detected error. In the absence of rigor, we can make some reasonable guesses for the forcing function. It

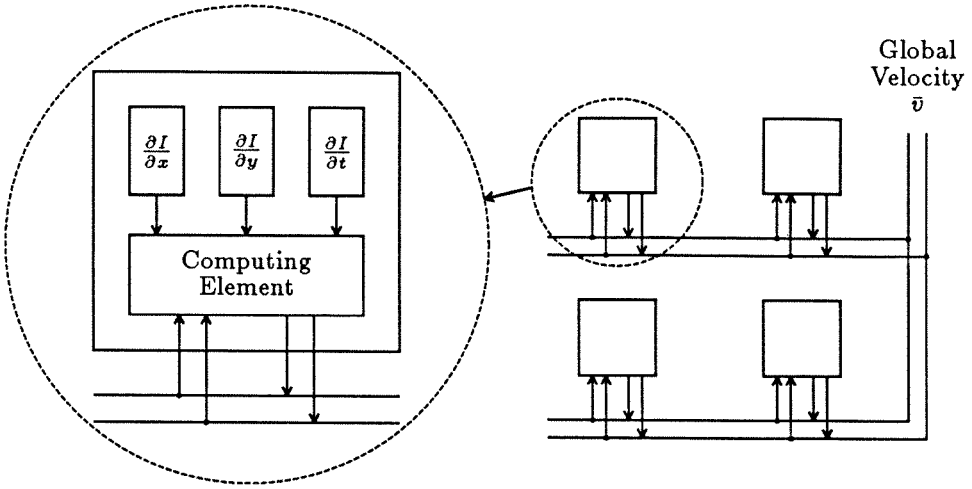


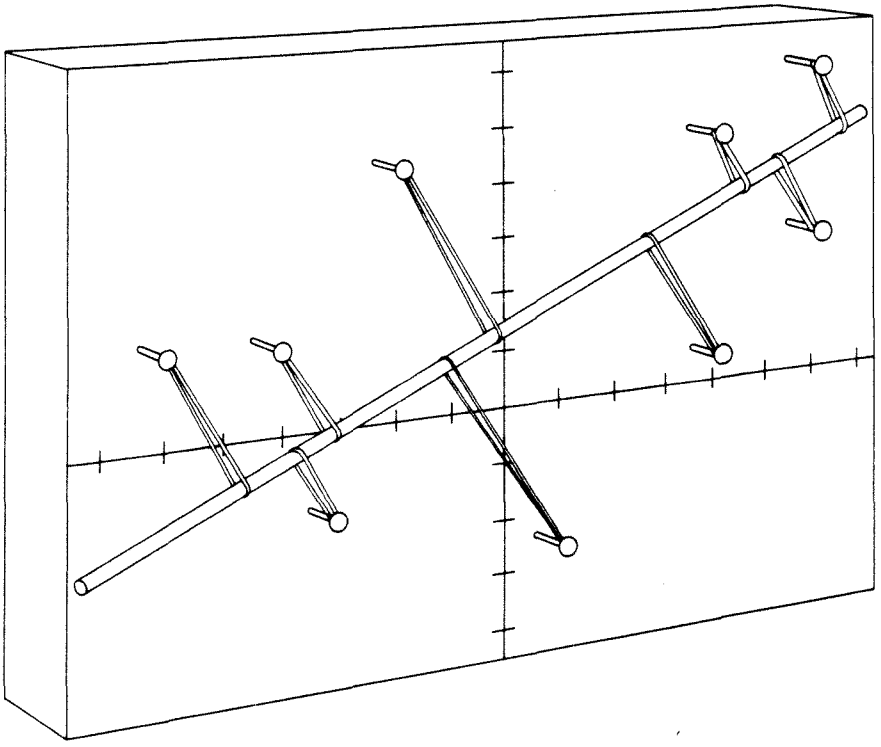
Figure 4.4: Block diagram of the constraint solver cell and array.

should be monotonic—the greater the error, the harder we should try to move it in the right direction. Since a forcing function linear in error distance is easiest to implement, we have selected it. A linear force gives rise to a quadratic energy function. (More on energy in Chapter 7.) In the energy context, the constraint solver is minimizing the error energy by finding the least-squares fit of the velocity point to all the constraint lines.

The similar problem of finding the best fit of a line to many points can be done by the mechanical analog device drawn in Scientific American [3] (Figure 4.5). Rubber bands connect fixed known points with a rigid rod that is free to move. The rubber bands generate forces on the rod that move it until it comes to rest at the best-fit position. Our converse problem of finding the best fit of a point to many lines can be similarly diagrammed (Figure 4.6) as a movable point attached by rubber bands to a number of fixed rods. The rods in this device represent the constraint lines and at any time are fixed in velocity space by the three line parameters derived from the image. The movable ring represents the global velocity point that comes to rest at the best fit to the intersection of all the constraint lines (fixed rods).

4.3 A Preliminary Formulation

We first extend the theory to incorporate two-dimension images and velocities by modifying the one-dimensional numerator-denominator formulation. We will see later that this preliminary formulation has some of the same shortcomings as the



A gadget for finding the line that best fits a series of data points

Figure 4.5: An analog mechanical device for finding a best-fit line. The rigid rod, connected by rubber bands to a set of fixed points, comes to rest in a position of "best-fit" to the points. From June 1985 Computer Recreations by A.K. Dewdney. Copyright 1985, Scientific American. Used by permission.

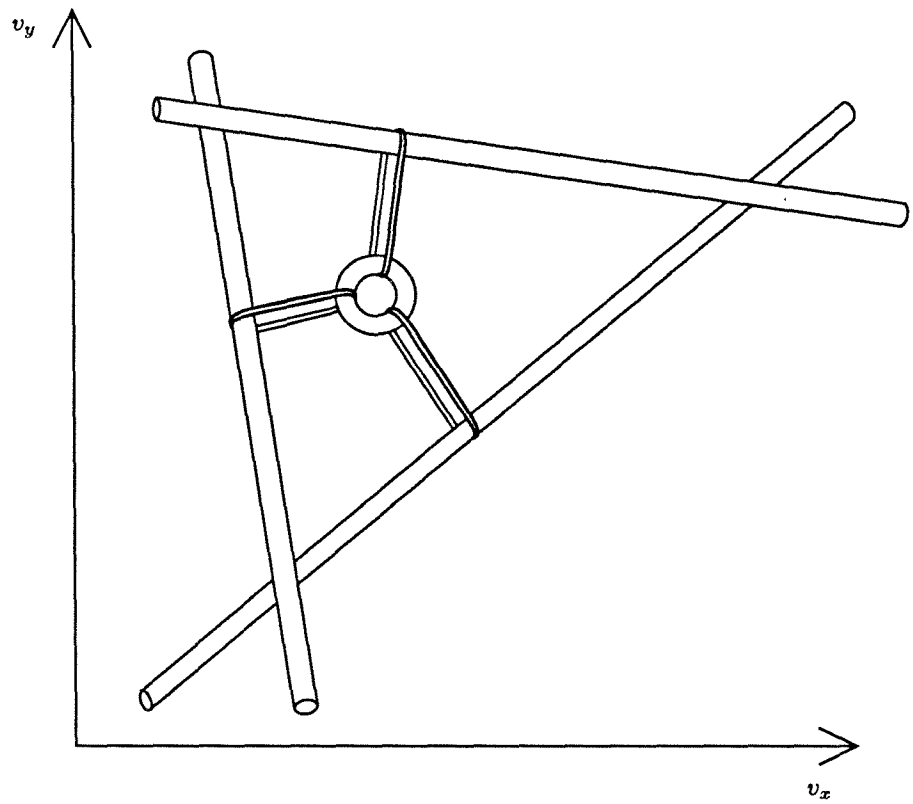


Figure 4.6: An analog model for velocity space constraint line intersection solver. Each rod represents a constraint line, fixed in velocity space by the image parameters $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$. The movable ring represents the global velocity and comes to rest at a point in velocity space that is a best fit to the intersection of the constraint lines.

one-dimensional formulation from which it arises. The value of this analysis is that the recognition of the major problem leads directly to the much better solution.

The analysis will proceed as it did for the one-dimensional case as follows:

- Find the equation for the relationship between velocity and the intensity derivatives.
- Solve this equation for velocity.
- Write the expression for the weighted average of velocities.
- Choose an appropriate weighting function.
- Substitute the expression for velocity into the weighted average equation.

The left-hand column of equations represents these steps already applied to the one-dimensional case in Chapter 3. The right-hand column contains the corresponding equations for the two-dimensional case.

One-Dimensional		Two-Dimensional		
$\frac{\partial I}{\partial t} = -\frac{\partial I}{\partial x} v$		$\frac{\partial I}{\partial t} = -\frac{\partial I}{\partial x} v_x - \frac{\partial I}{\partial y} v_y$		
$v = -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}}$		$v_x = -\frac{\frac{\partial I}{\partial t} + \frac{\partial I}{\partial y} v_y}{\frac{\partial I}{\partial x}}$		(4.2)
$\bar{v} = \frac{\sum_{j=1}^{n} -\frac{\partial I}{\partial t}\frac{\partial I}{\partial x}}{\sum_{j=1}^{n} \frac{\partial I}{\partial x}\frac{\partial I}{\partial x}}$		$\bar{v}_x = \frac{\sum_{j=1}^{n} -(\frac{\partial I}{\partial t} + \frac{\partial I}{\partial y} v_y)\frac{\partial I}{\partial x}}{\sum_{j=1}^{n} \frac{\partial I}{\partial x}\frac{\partial I}{\partial x}}$		(4.3)

What we have done is solved the local constraint equation for v_x in terms of v_y. Although with strictly local information we cannot determine v_x and v_y, given an initial guess at v_y, we can calculate v_x at each location, average the results and use the result to be the new v_x. Similarly, we can simultaneously use the current guess for v_x to compute a new value for v_y. Each cell then is using its knowledge of its own constraint between v_x and v_y to move the global values, initially just guesses, into closer agreement with its own constraint. Graphically this operation amounts to moving the present point in velocity space, represented by the global values $\langle v_x, v_y \rangle$, closer to the constraint line determined by $\frac{\partial I}{\partial t}$, $\frac{\partial I}{\partial x}$, and $\frac{\partial I}{\partial y}$ (Figure 4.7).

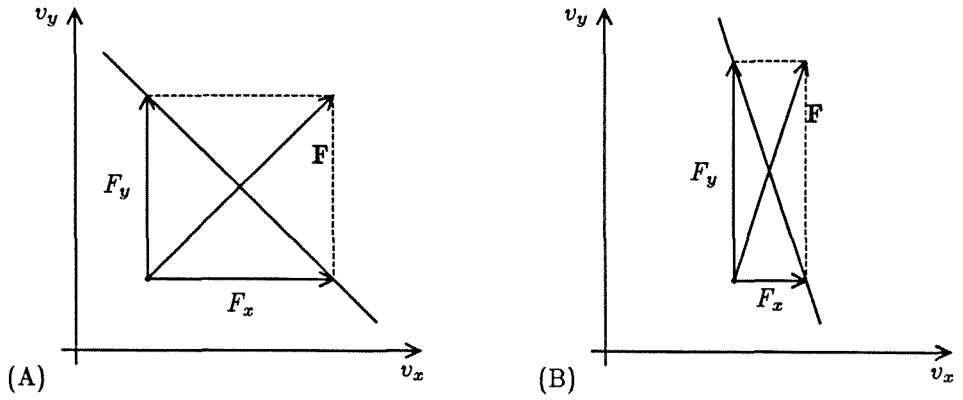


Figure 4.7: The naive way to try to satisfy your line constraint. (A) Sometimes it moves perpendicular to the line. (B) Sometimes it doesn't.

There are two problems with this approach. First, since the global values, v_x and v_y are used in the local computations, they must be normalized. This normalization is done by dividing by the sum in the denominator of Equation 4.3. In the one-dimensional case we could transfer the problem into the next level of aggregation, perhaps avoiding the division altogether, depending on the form of the velocity representation needed by this higher level. With the two-dimensional formulation, we must perform the division, with all its inherent problems, and supply this normalized value back to all the cells.

A more serious problem turns up when we examine a constraint line that is more nearly parallel with one of the axes (Figure 4.7(B)). We see that our method of determining the local contribution to the new global velocity according to Equation 4.2 has the following graphical interpretation: We get the new y-component of the correction force by finding the y-coordinate of the point on the constraint line directly above (same x-coordinate as) the present velocity point. Similarly, the x-component is determined by the point on the line across from the present velocity point. The resulting correction vector is in roughly the direction toward the constraint line *but* it is not necessarily orthogonal to it. This problem gets worse as the constraint line becomes parallel to one of the axes. Because the constraint line is the only information that exists locally, a local cell should try to move the global velocity point onto its constraint line. The cell should express no opinion as to where along its line the global velocity point should be. Any non-orthogonal component to the correction force amounts to an expression of bias that does not come from any information in the image.

If we were to implement this version of the algorithm, it would require three

multipliers for each axis. The term $\frac{\partial I}{\partial x} \cdot \frac{\partial I}{\partial y}$ can be shared between the two axes so a total of five multipliers per cell would be needed. We would need three sets of global wires per axis: one to carry the normalized velocity component, one for the normalizing factor, and one for the unnormalized velocity component. In addition, two global divider circuits would be needed.

4.4 A Better Formulation

In this section we address the orthogonality problem and as a result arrive at a new solution with the properties:

- The correction is always perpendicular to the constraint line.
- No division is required.

The new formulation directly constructs a correction force that is perpendicular to the constraint line as shown in Figure 4.8. The problem with the first attempt at a two-dimensional velocity tracking algorithm was that the correction force could have a component that was not orthogonal to the constraint line used to generate it. A direct orthogonal construction eliminates this problem.

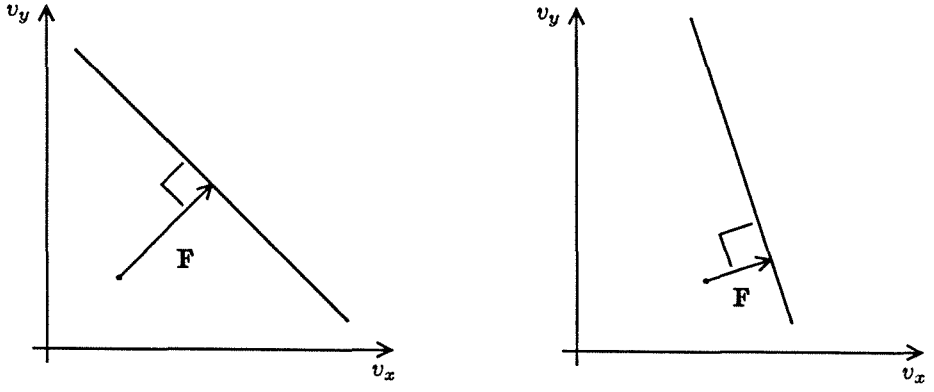


Figure 4.8: The correction force should be perpendicular to the constraint line.

If we rearrange the constraint line equation $\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0$ from the implicit form $Ax + By + C = 0$, to the slope-intercept form $y = \mathrm{m}x + \mathrm{b}$ we get:

$$v_y = -\frac{\frac{\partial I}{\partial x}}{\frac{\partial I}{\partial y}} v_x - \frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial y}}.$$

The slope of the constraint line, where defined, is $\mathrm{m} = -\frac{\partial I}{\partial x}/\frac{\partial I}{\partial y}$, so the slope of the desired perpendicular correction force is then the negative reciprocal, $-\frac{1}{\mathrm{m}} = \frac{\partial I}{\partial y}/\frac{\partial I}{\partial x}$. A unit vector in this direction would be:

$$\widehat{\Delta \mathbf{v}} = \frac{\nabla \mathbf{I}}{|\nabla \mathbf{I}|} = \left\langle \frac{\frac{\partial I}{\partial x}}{\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}}, \frac{\frac{\partial I}{\partial y}}{\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}} \right\rangle.$$

The magnitude of the correction force should be greater if the present point in velocity space is farther away from the constraint line. The force should go to zero as the point comes closer to lying on the constraint line. The direction of the force should always be perpendicular to the constraint line and with a sign such that the global velocity point will move toward the constraint line. A forcing function that is linear with error distance fulfills all these requirements and can be easily computed as follows:

$$D = \frac{\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}}{\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}}.$$

If we just plug the present values for the velocity components into the line equation and normalize by the quantity under the radical, we get D, a signed distance. The magnitude of D is the distance of the present velocity point $\langle v_x, v_y \rangle$ to the constraint line. The sign of D indicates which side of the line the point is on and therefore the direction of the correcting force. The vector, $\Delta \mathbf{v}$, from the current velocity to the point on the constraint line is then:

$$\begin{aligned} \Delta \mathbf{v} &= D \cdot \widehat{\Delta \mathbf{v}} = \left\langle D \frac{\frac{\partial I}{\partial x}}{\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}}, D \frac{\frac{\partial I}{\partial y}}{\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}} \right\rangle \\ &= \left\langle \frac{(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial x}}{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}, \frac{(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial y}}{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2} \right\rangle. \end{aligned}$$

Each cell should produce a force (electrical current), $\mathbf{F}$, that will tend to move the global velocity proportional to the detected error, $\Delta \mathbf{v}$. We would also like to scale this correcting force according to our confidence in the local data, C.

$$\mathbf{F} = C \cdot \Delta \mathbf{v}$$

There is more information in a higher contrast edge, or at least there is a higher signal to noise ratio. We should afford a greater weight to the correcting forces in

those higher contrast areas. Our measure of contrast in the image is the intensity gradient, a vector quantity $\nabla \mathbf{I} = \langle \frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \rangle$. Confidence is related to the magnitude of the gradient, $|\nabla \mathbf{I}| = \text{sqrt}(\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2)$. If we choose our confidence, C, to be the square of the magnitude of the intensity gradient we have:

$$C = |\nabla \mathbf{I}|^2 = \frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2.$$

This choice greatly simplifies the correcting force calculation by canceling out the denominator. Our force equation becomes:

$$\mathbf{F} = \left\langle (\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial x}, (\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial y} \right\rangle.$$

Writing the two components of this vector equation separately we have:

$$\boxed{\begin{aligned} F_x &= (\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial x} \\ F_y &= (\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}) \frac{\partial I}{\partial y}. \end{aligned}} \tag{4.4}$$

These are the equations implemented by the circuits of Chapter 5.

4.5 Comparing the Two Formulations

Notice first that the better formulation requires *no* division. This feature is very important because of the difficulty with implementing a divider circuit and the divide-by-zero problem. In the first formulation, we tried to get around this problem by pushing it to the next higher level. This "hack" required two sets of global wires out of each cell to carry the numerator and the denominator, and one set to carry the result of the division back to each cell. The first formulation required five or six multipliers. In the better formulation, the common term, $(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t})$ in the equations for the two components can be calculated once and used for both further computations. Each cell then requires only four multipliers. In addition, the second formulation avoids the data dependency of the first one by moving the global velocity perpendicular to its constraint regardless of the position of the constraint relative to the axes of the sensing array.

	Numerator-Denominator Formulation	Orthogonal Formulation
Number of Multipliers per Cell	5	4
Sets of Global Wires	3	1
Requires External Division Circuit	Yes	No
Always Moves Perpendicularly	No	Yes

4.6 Design of the Constraint Solver Cell

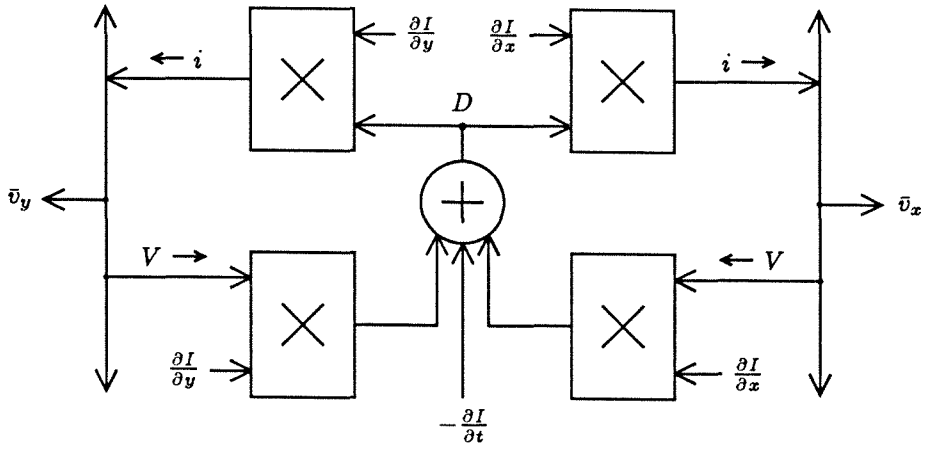


Figure 4.9: Block diagram for each cell's motion detection circuitry.

The block diagram of an implementation of the orthogonal two-dimensional formulation is shown in Figure 4.9. There are several similarities to the weighted divider implementation of the resistor network formulation for one dimension as shown in Figure 3.10. If v_y in the two-dimensional case is set to zero, the effect of the two multipliers on the right is eliminated and the system reduces to the one-dimensional case. One view of the one-dimensional system was that it computed an error and used this as feedback to correct the global average velocity. This is consistent with the two-dimensional view where the error term is the signed scalar quantity D, the distance in velocity space of the global average velocity to the locally known constraint line. This distance error is also used as feedback to correct the system. For two dimensions, this error is multiplied by the appropriate vector perpendicular to the constraint line, to generate a correction force in the same direction to correct the global velocity vector.

	One Dimension	Two Dimensions
Velocity Computed	v	$\mathbf{v}$
Locally Measured Quantities	$\frac{\partial I}{\partial x}, \frac{\partial I}{\partial t}$	$\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial I}{\partial t}$
Error Term	Difference $(v_j - \bar{v})$	Distance D
Correction Force	$F = (\frac{\partial I}{\partial t} + \bar{v}\frac{\partial I}{\partial x})\frac{\partial I}{\partial x}$	$F_x = (\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial x}$ $F_y = (\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial y}$
Multipliers / Cell	2	4

4.7 Summary of One-Dimensional Motion Detection

The extension of the velocity sensor to two dimensions has the same collective nature as the one-dimensional case. Local information, weighted by confidence, is aggregated to compute a global result. Each cell performs a simple calculation based on moving the global velocity state into closer agreement with its locally measured information. The collective behavior that emerges is the tracking of the intersection of constraint lines to solve the two-dimensional ambiguity, when possible, and accurately report the two-dimensional analog velocity of the image.

Chapter 5

Implementation: Circuit Details

Our motion detection chip consists of a two-dimensional array of cells. Each cell contains a photosensor and some other circuitry. The outputs of the photosensors are routed to adjacent cells so that the computational elements of each cell can monitor the light intensity at its nearest neighbor in each dimension as well as its own intensity. The x and y components of velocity are distributed on wires globally to each cell. These values represent the present best guess for the global velocity. The voltages on these velocity wires are inputs to each cell. From the global velocity inputs and the local light intensity inputs each cell calculates a correction to the global velocity and expresses this correction in terms of currents that it applies to the same global velocity wires with an appropriate magnitude and sign. The global velocity wires perform a current sum of the correction contributions from all cells. The velocity voltages change according to the net correction from all the cells.

The inputs to the chip consist of the optical image focused on the die (whose motion the chip measures) and several analog wires to control the gains of each component of the cells' circuitry. The outputs of the chip are the analog voltages on the global velocity wires.

Each cell contains four analog multipliers, a differential amplifier and some current summing nodes to implement the correction computation of Equation 4.4. Figure 4.9 shows the block diagram for each cell. We present details of the individual circuits used to implement the velocity detector.

5.1 Choice of Technology and Representation

This new class of motion sensors is implemented using CMOS technology. CMOS was chosen because in this technology, high voltage gain differential amplifiers can be easily built using both n-type and p-type transistors. In addition, the inherent bipolar transistors in a CMOS process can be used to generate photocurrent and to perform low-noise current amplification [12]. Finally, because CMOS is perceived to be the dominant digital technology for some time to come, we will continue to have access to better and better CMOS fabrication processes as time passes.

Analog values are represented throughout the chip as the difference between the analog voltages on two wires. This dual-rail scheme, although requiring more wires to represent a value, has several advantages. There is no need of an absolute voltage reference and the operating range restrictions that this would entail. Many of the circuits describe below are based on the differential pair. This circuit, in its simplest form, has an input that is the difference in voltage between two nodes and has an output that is the difference in the current drawn from two nodes. The differential pair is used throughout the motion sensor design and so the dual-rail representation is a natural choice.

5.2 A Logarithmic Photo Detector

In Section 3.2 we derived the computation necessary to extract the velocity of a intensity terrain from local measurements. This method works for the motion of any terrain. In particular, a compression of the intensity scale such as a logarithmic one, produces a different terrain curve. As long as the same compression is done at each position, the analysis remains the same. The velocity of the $\log(I)$ terrain curve is the same as the velocity of the image.

The reasons for using a logarithmic sensor are two-fold:

- The compression of large amplitude signals results in a wide dynamic range.
- Logarithms weight contrast ratios in the image the same for all levels of scene illumination (more detail below).

The intensity, I_j, incident on a photosensor is proportional to the reflectivity, R_j, of the corresponding point on the object multiplied by the illumination of the scene, L so:

$$I_j = L_0 R_j.$$

If the illumination changes with time or varies slowly across the image, the ratio of nearby intensities, the local contrast ratio, C, remains the same:

$$C_{1:2} = \frac{I_1}{I_2} = \frac{L_0 R_1}{L_0 R_2} = \frac{L_1 R_1}{L_1 R_2}.$$

With sensors taking the logarithm of all intensities, differences between sensor values are independent of scene illumination levels. The difference quantity is the logarithm of the contrast ratio:

$$\log I_1 - \log I_2 = \log \frac{I_1}{I_2} = \log C_{1:2}.$$

The logarithmic photosensors have been fabricated, tested and reported by Mead [12].

To build a logarithmic photosensor, the inverse function, an exponential is used in the feedback path of a high gain amplifier (Figure 5.1). Any error between the input intensity and the exponential of the output is amplified and affects the output such that the error is reduced.

The schematic for the photosensor is shown in Figure 5.1. The Light striking the base-emitter junction of a phototransistor creates photocurrent that is amplified by the transistor. The parasitic bipolar transistor in a CMOS process typically has a current gain, β, of 300 to 500. The cross-sectional view shows the vertical PNPphototransistor. The well forms the base and the substrate forms the collector. Since the substrate is tied to ground (or some other constant voltage) to insure uniform behavior of the MOS devices, the bipolar can only be used with its collector grounded. A Darlington configuration of the four bipolar transistors provides the high amplification in the circuit. A diode-connected MOS p-type transistor, M1, provides a pull-up and yields an output voltage as a function of the amplified current. The output voltage is fed back to the gate of another p-transistor, M2, operating in its subthreshold region (where $V_{GS} < V_{TH}$). In this region the source-drain current, i_{DS}, is proportional to the exponential of the gate-source voltage, V_{DS}, so $V_{DS} \propto e^{i_{DS}}$. This exponential behavior in the feedback path provides the logarithmic forward response of the circuit. A third p-type device, M3, provides a voltage offset from V_{DD} to keep the second transistor in subthreshold. Finally, the drain-source current from the subthreshold transistor is summed with the photocurrent, amplified by the phototransistor, to make the error current.

The feedback connection is to the emitter of the phototransistor instead of the base to improve response time of the sensor. The capacitance of the MOS transistor source is charged and discharged by a current 300 times greater than the photocurrent. This connection choice yields a lower range of operating intensities than does

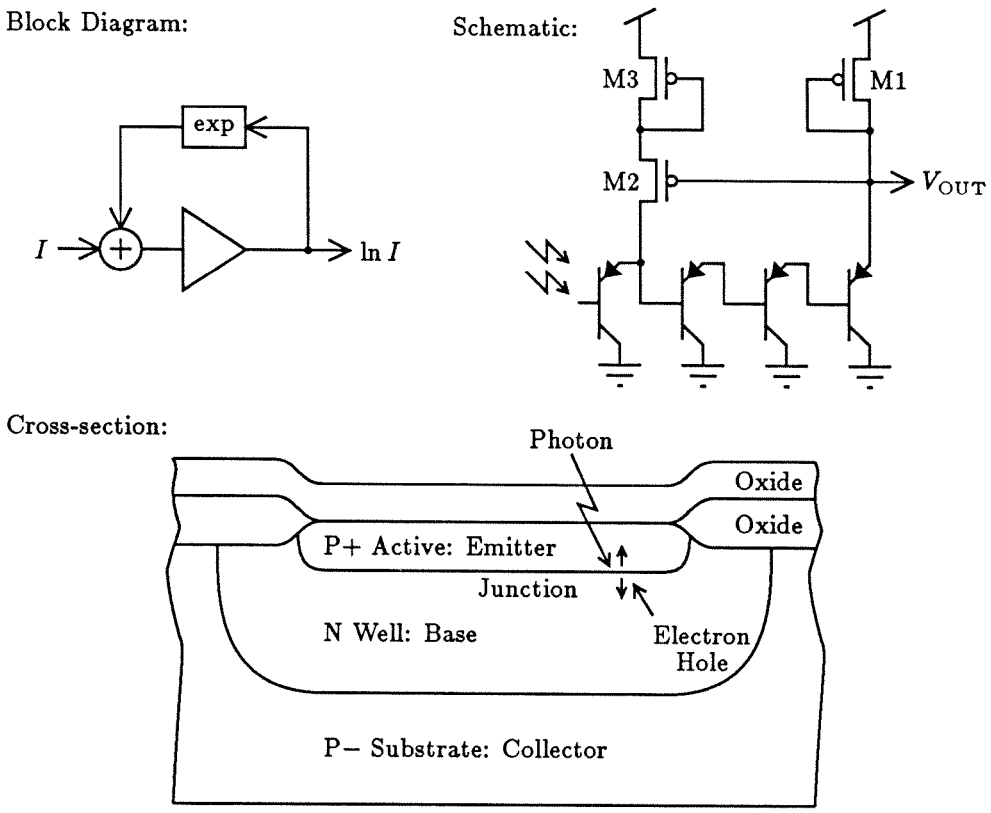


Figure 5.1: Block diagram and circuit schematic for the logarithmic photosensor and cross-section of the vertical PNP phototransistor.

a base connection. The range of currents over which the circuit's transfer curve will closely approximate a logarithm is determined by the fabrication parameters of the transistors. The pre-amplification of the photocurrent by the phototransistor reduces the maximum intensity but extends the operating range for low light levels.

The offset voltage provided by transistor M3 is not constant but follows the same exponential voltage-current relationship as does M2. For M3, the current increases with gate-source voltage at about 115 mV/decade of current, over more than five decades of current. Transistor M2 has a similar constant that is increased by the back gate effect so that the combination of M2 and M3 produce a voltage of about 325 mV/decade of current. Room light level produces a voltage of about 1.5 Volts down from V_{DD} or 3.5 Volts. A factor of 30 variation in light intensity either way will produce an output voltage within the range of 3 to 4 Volts.

5.3 The Differential Pair and the V_{MIN} Problem

A differential pair of MOS transistors (Figure 5.2) working in subthreshold with current source i_0 have a differential output current proportional to the product of i_0 and the differential input voltage. This result can be shown by noting that in saturation the drain current, i_{DS}, is independent of V_{DS} and in subthreshold saturation i_{DS} is proportional to $e^{V_{GS}}$. Therefore:

$$\begin{aligned} \Delta i &= i_2 - i_1 \\ &\propto e^{V_{GS2}} - e^{V_{GS1}}. \end{aligned}$$

Using a linear approximation for the exponential we get:

$$\begin{aligned} \Delta i &\approx (V_{GS2} - V_{GS1}) e^{V_{AVG}} \\ &= \Delta V \, i_{AVG} \\ &\propto \Delta V \, i_0. \end{aligned}$$

This differential pair circuit, with its current source, takes a differential voltage input and produces a signed differential current. This differential output current is scaled by the current from the current source.

Although the differential pair performs well it does have an important restriction in its operating range. The drain-source voltage, V_{DS}, of the two transistors of a differential pair must be greater than about 100 mV for them to be in saturation and therefore act as current sources independent of drain-source voltage. The restriction on the output voltage range is determined by the voltage on the common source node, V_0. The transistor with the higher gate voltage will carry most of the current,

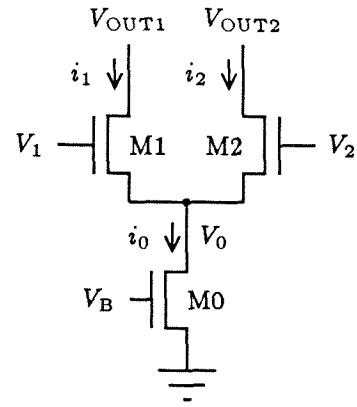


Figure 5.2: An important analog building block: the differential pair.

i_0, so the gate-source voltage of that transistor will be approximately the same as the gate-source voltage of the biasing transistor, V_B. The common source node will track the higher of the two gate voltages so:

$$V_0 \approx \max(V_1, V_2) - V_B.$$

For proper operation, the output voltages, V_{OUT1} and V_{OUT2}, must both be at least 100 mV greater than the common source node voltage, V_0. The minimum output voltage, V_{MIN}, is then:

$$V_{MIN} = \max(V_1, V_2) - V_B + 100\,\text{mV}$$

and for proper operation:

$$V_{OUT1} \geq V_{MIN} \quad \text{and} \quad V_{OUT2} \geq V_{MIN}.$$

If the voltage on one of the output drain nodes drops below V_{MIN}, the drain-source voltage for that transistor will be less than 100 mV. The output current will reduce to zero and can become negative if the output voltage drops even further. We call this the V_{MIN} problem. Clearly under these circumstances the assumption that the transistors of the differential pair act as current sources is no longer valid. The circuit no longer produces a differential output current proportional to the differential input voltage. The direction of the current can even change, drawing current from the other output node.

5.4 Current sources and mirrors

A MOS transistor operating in the saturated region makes a reasonable current source. As V_{DS} changes, i_{DS} remains relatively constant. The current can be set by the gate voltage. A convenient way to set this gate voltage is with a current-mirror transistor and a controlling current as shown in Figure 5.3. Above threshold, the condition for saturation is $V_{DS} > V_{GS} - V_{TH}$. The control transistor is guaranteed to be saturated because $V_{GS} = V_{DS}$. The control transistor biases its gate voltage so that in steady state it sinks all the control current. This same gate voltage is provided to the controlled transistor. Provided the drain voltage is high enough to keep the controlled transistor saturated, the drain-source current i_{DS} will be essentially the same for both transistors. The circuit will maintain a fixed current equal to the input control current over quite a wide range of output voltages.

In the subthreshold region (where $V_{GS} < V_{TH}$), MOS transistors behave much like bipolar transistors so the condition for saturation is $V_{DS} > \sim 100\,\text{mV}$ [13]. Saturation is not a function of V_{GS} so within subthreshold the output of the current mirror will remain constant down to within 100 mV of ground independent of the current level.

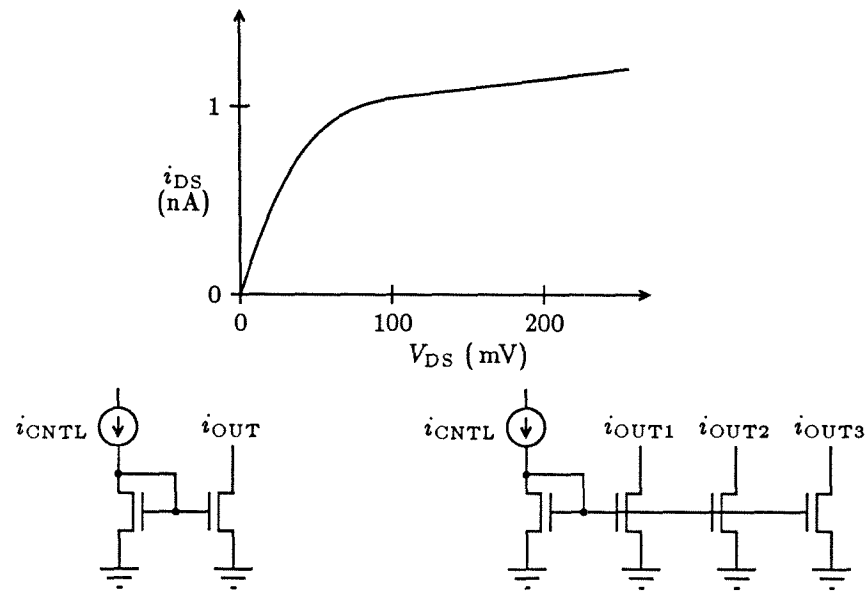


Figure 5.3: (A) A characteristic transistor curve. (B) A current mirror. (C) A current mirror with multiple outputs.

A current mirror with differently sized controlling and controlled transistors will produce an output current scaled by the geometric ratio.

One common use for the current mirror is to set many currents to the same value. For example, all the current sources for the multipliers in our chip are set with a single controlling transistor. The gate voltage is then distributed throughout the chip to many identical controlled transistors—one for each multiplier. The current sources throughout the chip will produce the same current to the extent that the transistors are fabricated identically. Threshold variations and geometric variations will contribute to variations in the resulting currents. Also, the distributed line is susceptible to noise. In the subthreshold region especially, because i_{DS} goes as $e^{V_{GS}}$ a variation in the gate voltage due to noise of 0.1V can produce a current variation of about a factor of ten.

Current mirror are often used to combine the two differential current outputs of the differential pair into a single bidirectional current output. The differential pair, its current source, and a pair of transistors to mirror the differential pair currents, form a transconductance amplifier. Such an amplifier, with its output connected to its input, is used in the differentiator (Figure 5.4(D)).

5.5 Integrators and Differentiators

Part of the velocity correction calculation of each cell requires the time derivative of the light intensity signal. A conventional resistor-capacitor (RC) differentiator is shown in Figure 5.4. The time constant, τ, of the circuit is proportional to the product of the resistance, R, and the capacitance, C so $\tau = R \cdot C$. Since we will want to control this time constant after fabrication, we will make the resistor a variable one. It is difficult in our CMOS fabrication technology to make a linear variable resistor with one end tied to ground. It is also difficult to make a linear, low leakage capacitor with both nodes floating (unless you have a special capacitor layer in your CMOS technology). We can make somewhat better resistors and capacitors if we switch their order in the circuit. A capacitance to ground can be formed by the gates of two transistors. The high leakage sides are tied to the power supply rails, V_{DD} and Gnd, leaving the varying node in low leakage poly. A resistor has the property that the current through it is proportional to the difference in voltage across its two ends. A transconductance amplifier in its linear region, connected as a voltage follower, has the same property (Figure 5.4(D)). Furthermore, the scaling of this current and thus the value of the resistance, is set by the current source at the bottom of the amplifier. This source can easily be controlled from off chip after fabrication.

Since we swapped the order of the resistor and capacitor, we built an RC integrator instead of a differentiator. The current through the series resistor and capacitor

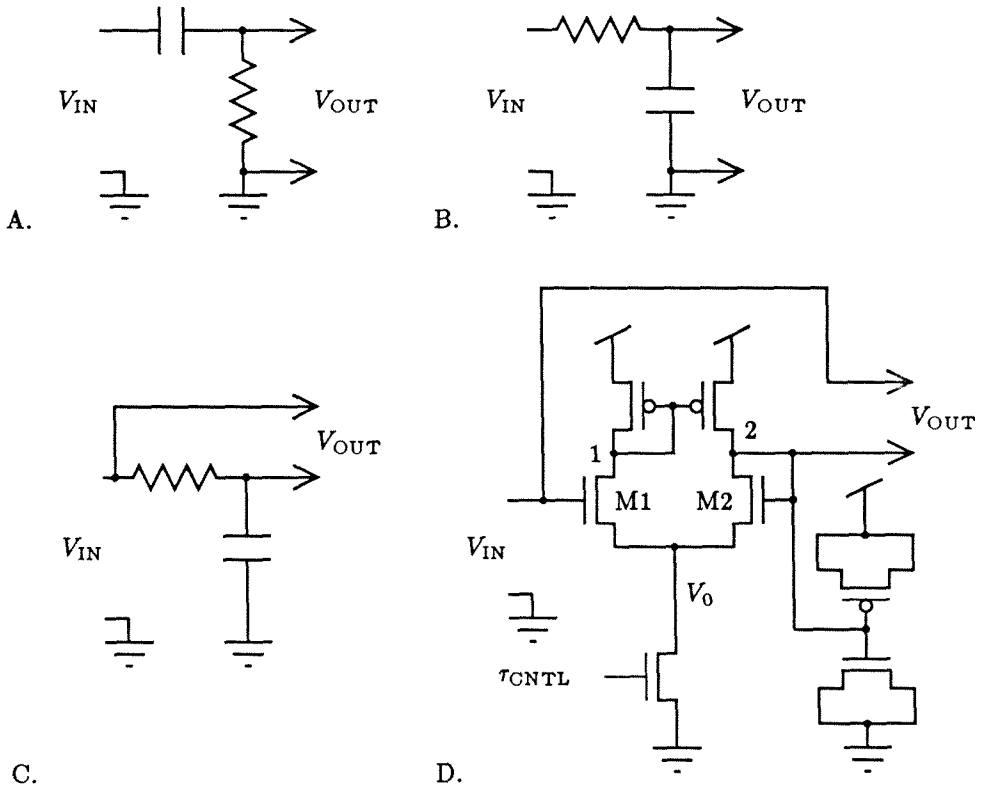


Figure 5.4: (A) The RC differentiator. (B) The RC integrator. (C) An RC integrator used as a differentiator. (D) A CMOS integrator used for differentiation.

did not change due to the swap so the time derivative signal of interest is still produced but is the voltage across the resistor, not the capacitor. Alternatively, we can say the derivative is the difference between a signal and its integral. (This is precise only if we are talking about the "leaky" RC type of integrators and differentiators.) In Laplace transforms, the output voltage of the RC differentiator taken across the resistor is:

$$V_R = V_{IN}\frac{s}{1/\tau + s},$$

where τ is the time constant $R \cdot C$. The output voltage of the RC integrator taken across the capacitor is:

$$V_C = V_{IN}\frac{1/\tau}{1/\tau + s}.$$

We can see that the difference between the signal, V_{IN}, and the output of the integrator, V_C, is:

$$\begin{aligned} V_{IN} - V_C &= V_{IN} - V_{IN}\frac{1/\tau}{1/\tau + s} \\ &= V_{IN}\frac{s}{1/\tau + s} \\ &= V_R. \end{aligned}$$

The output of our differentiator is then the difference between the voltages on two wires. This construction fits in well with our dual-rail scheme and the differential amplifier that follows this stage.

The resistor-equivalent circuit in the integrator avoids the V_{MIN} problem because of the way it is connected. The voltage on node 1, on the left, is the gate bias voltage for the current mirror made of p-type transistors. For subthreshold currents, this voltage will be within a threshold of V_{DD} or greater than about 4.3 Volts. The output voltage on node 2 can go higher than 4.3 Volts without exceeding V_{MIN}. For voltages below 4.3 Volts, M1 stays saturated as it must for normal operation. The drain of transistor M2, node 2, is an output and is connected to an input, the gate of the same transistor. This gate-drain connection assures that transistor M2 is saturated whenever it is conducting. When V_{IN} gets larger than the node 2 voltage, then the common node voltage, V_0, can become larger than the output node voltage. For the general differential pair this could cause reverse currents through transistor M2. For the integrator's resistor circuit, the gate-drain connection makes transistor M2 act like a diode. When V_{DS} becomes negative, the gate-drain connection becomes a gate-source connection. With $V_{GS} = 0$, the transistor will conduct no reverse current. The resistor circuit avoids the effects of the V_{MIN} problem by virtue of its input-output connection.

5.6 A Four-Quadrant Analog Multiplier

The differential pair circuit takes a signed differential voltage, multiplies by a unidirectional current and produces a signed differential current. We need a multiplier that can take as inputs two signed numbers (a four-quadrant multiplier) instead of one signed and one positive (two-quadrant). Another differential pair with inputs reversed can provide the other two quadrants of operation, provided that we arrange the two unidirectional current sources correctly. The difference between the two current sources must be proportional to the second (signed) input to the multiplier. If our second input is a dual-rail pair of currents, we can use this multiplier as is, or through a pair of current mirrors. We call this a Vi-i multiplier because it requires one voltage input (V), one current input (i), and produces a current output (i), all dual-rail (Figure 5.5(B)).

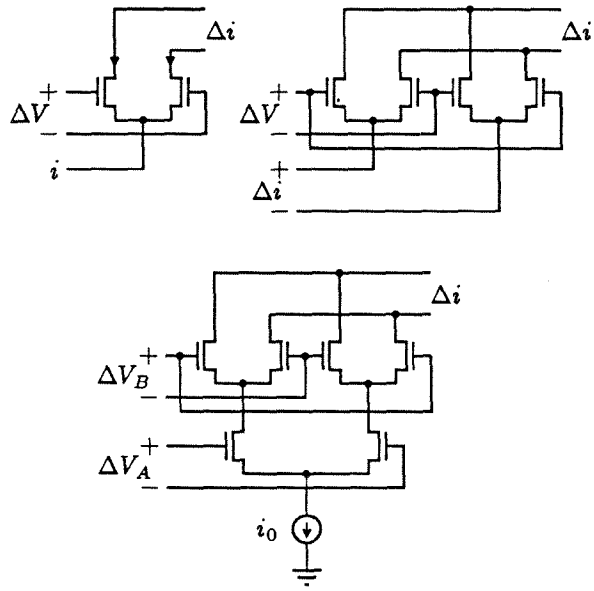


Figure 5.5: (A) A two-quadrant multiplier. (B) A four-quadrant Vi-i multiplier. (C) A four-quadrant VV-i multiplier.

We can make a multiplier with two pairs of differential voltage inputs and differential current outputs. This type is a VV-i multiplier. A third differential pair connected to the bottom of the Vi-i multiplier turns the second voltage input pair into the required proportional differential currents (Figure 5.5(C)). This configuration is structurally identical to the Gilbert multiplier well known in the bipolar

world [4].

Our first implementation of the four-quadrant multiplier is susceptible to the V_{MIN} problem. Each multiplier contains three differential pairs. Each of the output nodes of the bottom differential pair is the common node of the differential pair above. Applying the V_{MIN} restriction throughout yields the relationship between the two sets of input voltages and the output voltages. For correct operation:

$$\begin{aligned} V_B+ &\geq \max(V_A+, V_A-) + 100\,\text{mV} \\ V_B- &\geq \max(V_A+, V_A-) + 100\,\text{mV} \\ V_{\text{OUT}}+ &\geq \max(V_B+, V_B-) - V_{\text{B}} + 100\,\text{mV} \\ V_{\text{OUT}}- &\geq \max(V_B+, V_B-) - V_{\text{B}} + 100\,\text{mV}. \end{aligned}$$

In general, the V_B input voltages must be greater than the V_A input voltages and the output voltages must be higher yet.

The V_{MIN} restriction on the multiplier determines the choice of operating point throughout the present motion detector design. A modification of the multiplier circuit could reduce some of the restrictions. The bottom half of the multiplier can be decoupled from the top half by inserting a pair of current mirrors between the two halves. This eliminates the restriction between the two pairs of inputs. Adding an additional pair of current mirrors on the outputs, decouples the upper input voltages from the output voltages. Although requiring more chip area, this multiplier can be used over a wider range of input operating conditions.

5.7 Putting the circuits together

Figure 5.6 shows a detailed schematic for the circuitry in one cell. The output of the logarithmic photosensor goes to an integrator with a time constant set with the external analog current control labeled τ. A choice in the range of 0.3 to 0.4 Volts for the τ control knob allows operation for velocities typically encountered by a mouse pointing device. The difference between the intensity signal and its integrated value is amplified by a transconductance amplifier and summed onto the differential pair of nodes labeled D (for Distance in velocity space of the current global velocity point from this cells constraint line). The amplifier gain is set externally by the TS analog control. TS stands for Time Scaling.

The spatial derivatives are estimated by the difference in the intensity values of adjacent sensors. The two spatial derivatives are multiplied by the corresponding velocity components and the two results are summed onto the D nodes. The gain of the multipliers is controlled by the external analog control labeled GS for Gradient

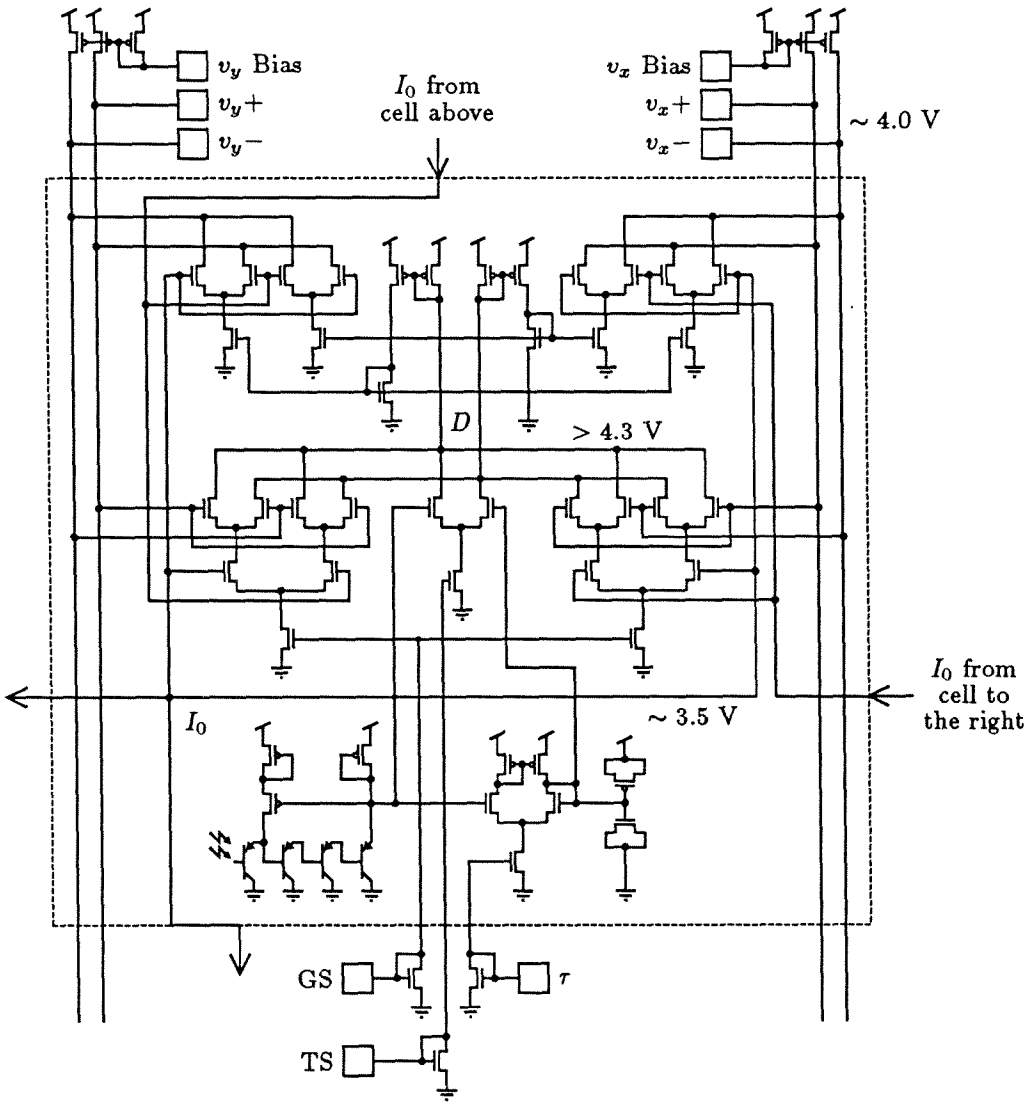


Figure 5.6: Schematic for the circuitry within each cell.

Scaling. The relative scaling of the gradient and the time derivative by the corresponding analog controls, determine the scaling of the global velocity output. In practice, the GS and TS controls are usually the same and are operated at a voltage between 0.5 Volts and 0.7 Volts.

The differential current summed onto the D nodes represents the distance term which is the common term in the two parts of Equation 4.4. This D current is passed through a double current mirror. The mirrors allow the currents to be duplicated and used in two places in the next stage of computation. The D currents are used as inputs to another pair of Vi-i multipliers that multiply them by the appropriate spatial derivative and sum the resulting current back onto the global velocity wires.

The global velocity wires are presently brought out to pads so that the pull-up resistors can be varied to set the operating point and so that enough capacitance can be added to assure one dominant time constant and thus prevent oscillation. In practice, the stray capacitance is sufficient to prevent oscillation. Appropriate values for external pull-up resistors are in the range of 100 to 1000 MΩ due to the small subthreshold currents. These unusually large resistors can be obtained and used as loads. A better way to provide pull-ups is to use the two pairs of on-chip p-transistors as high resistance current sources. The current through these transistors is set by the analog v Bias controls and is chosen to keep the common mode voltage of the $v+$ and $v-$ lines at an appropriate operating point.

The V_{MIN} operating range restrictions of the multipliers determine the necessary operating points throughout the circuit. The common mode voltages at selected points in the circuit are shown in Figure 5.6. These voltages satisfy the constraint between the two sets of inputs and the outputs of each of the multipliers. Since the outputs of multipliers must feed the inputs of other multipliers, circuitry is needed to shift the level of the signals to a lower common mode voltage. At the D nodes, a pair of double current mirrors provides this function. The diode-connected control transistors maintain the output nodes above 4.3 Volts over a large range of subthreshold currents. The controlled currents of these current mirrors can operate over a range of voltages down to within ~100 mV of ground. These currents feed into Vi-i type four-quadrant multipliers.

The output currents of the top two multipliers tend to discharge the global velocity lines. Current is supplied to the velocity lines by matched pairs of p-type pull-up transistors. The gate voltage is the same for both transistors in the pair and is chosen such that the common mode voltage of the pair of velocity lines is within operating range—about 4.0 Volts. The non-ideality of the current sources connected to the velocity lines, both the pull-ups and the multipliers, tend to keep the velocity voltages near the same value. Although not a strong effect, this tendency becomes important for low contrast images. Experimental data and discussion of this effect

are included in Chapter 6. Transistors in their saturation region provide a much greater dynamic resistance than would external resistors.

The high resistance of the pull-up transistors reduces their non-ideality as current sources but makes the common-mode voltage highly dependent on the controlling gate voltages, v_x Bias and v_y Bias. For any given image there is a setting for the bias voltage that will keep the common mode velocity voltage near 4.0 Volts. As the image intensity varies, the optimum setting of the bias changes as well. To make full use of the wide dynamic range of the photosensors, a mechanism for automatically adjusting the bias voltages is needed. Our present prototype chip is augmented with an external operational amplifier to provide this automatic bias. The common mode voltage is compared to a fixed voltage of 4.0 Volts and the error fed back to the gate of the p-transistor pull-up. Future versions of the motion detector chip will include a similar circuit on board.

5.8 Motion Detector Circuit Test Results

Test results for the CMOS differentiator of Section 5.5 are shown in Figure 5.7. Constant amplitude sine waves were applied to the input of the differentiator. The output amplitude is shown as a function of frequency on a log-log plot for various settings of the τ control. Each curve approximates closely the one-pole roll-off expected from an RC differentiator. The knees of the curves step regularly with τ control voltage. This behavior is expected because in the subthreshold region the resulting current varies exponentially with gate voltage.

Figure 5.8 shows a typical multiplier characteristic test result. The output differential current is plotted as a function of one of the inputs. The other input is used as a parameter for the family of curves. The linear region seems to be about 500 mV with smooth limiting behavior outside this region.

5.9 Circuit Summary

An 8×8 array of motion detection cells was fabricated on a standard MOSIS CMOS-Bulk fabrication run. Figure 5.9 is a photograph of the chip. A lambda of 1.5 μm yielded a die size of about 4500×3500 μm. Individual test results for the photosensor, differentiator, and multiplier are reported above. Test results and characterization of these circuits working in concert as an integrated motion detector are reported in the next chapter.

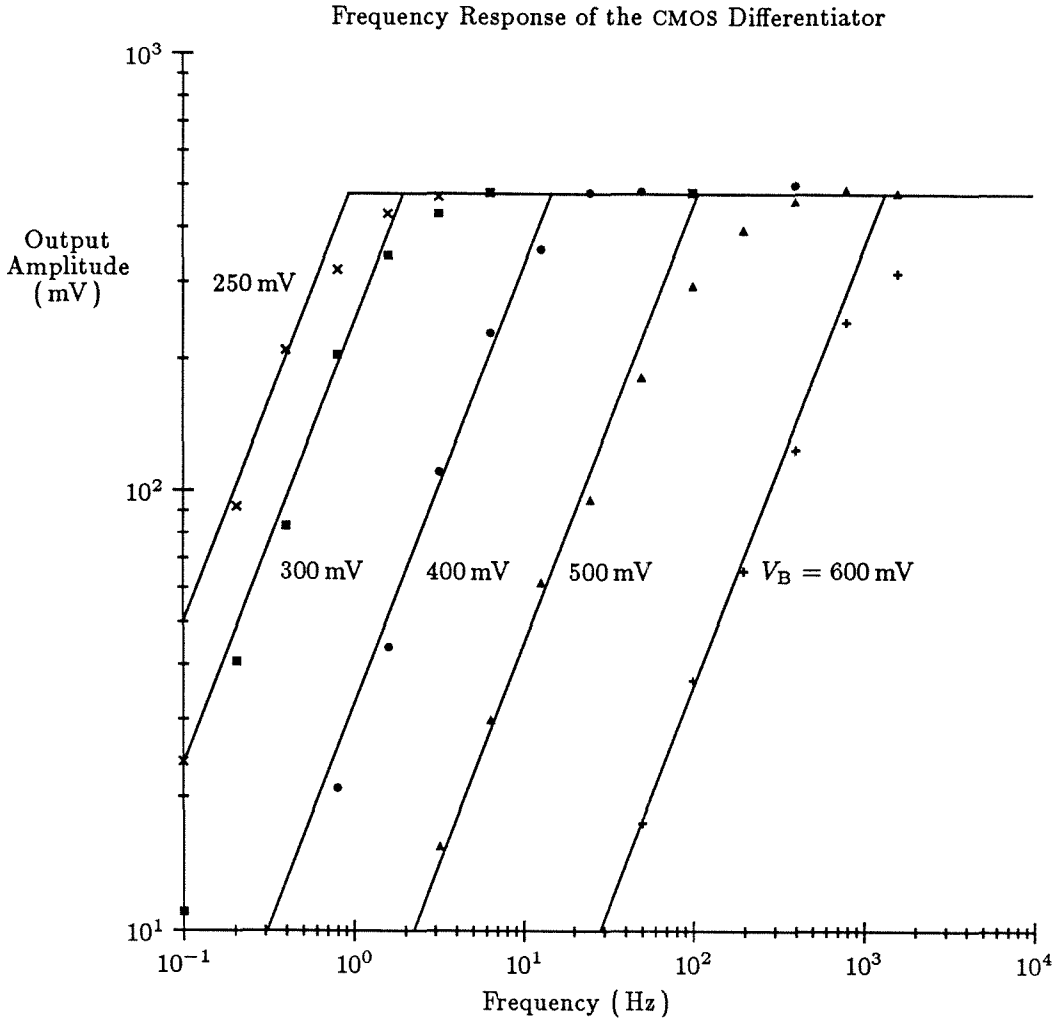


Figure 5.7: Differentiator output as a function of frequency and control voltage.

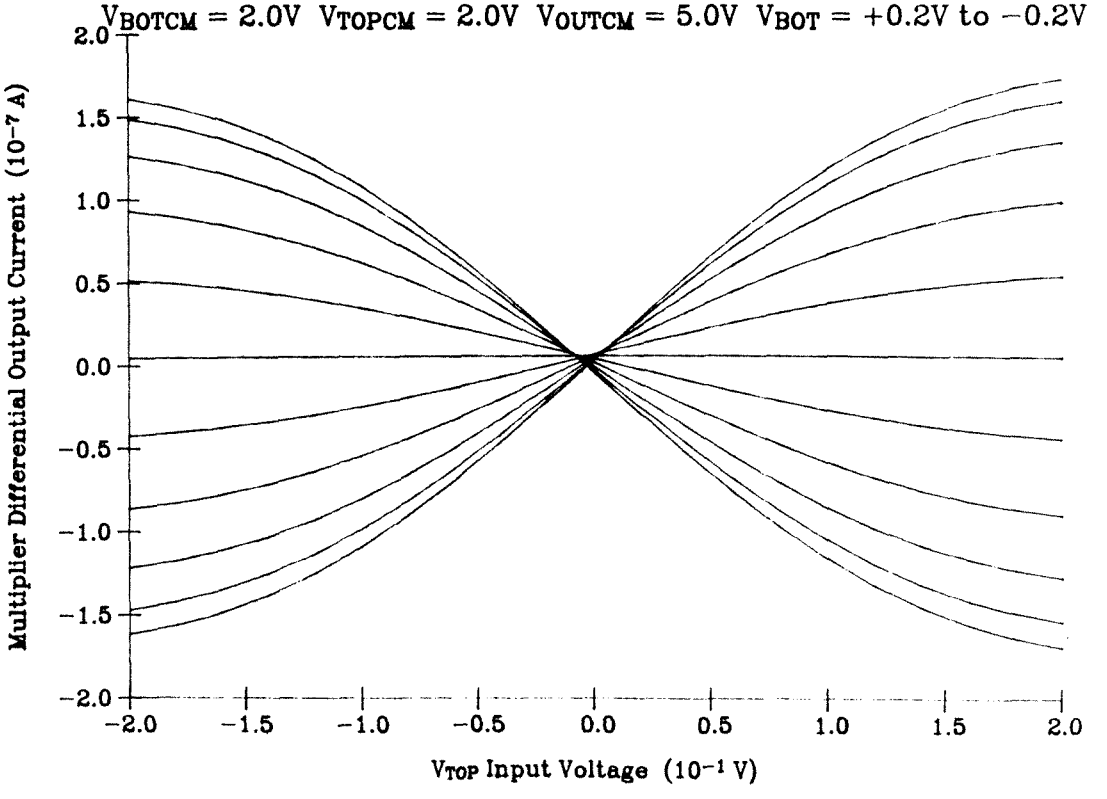


Figure 5.8: Multiplier test results.

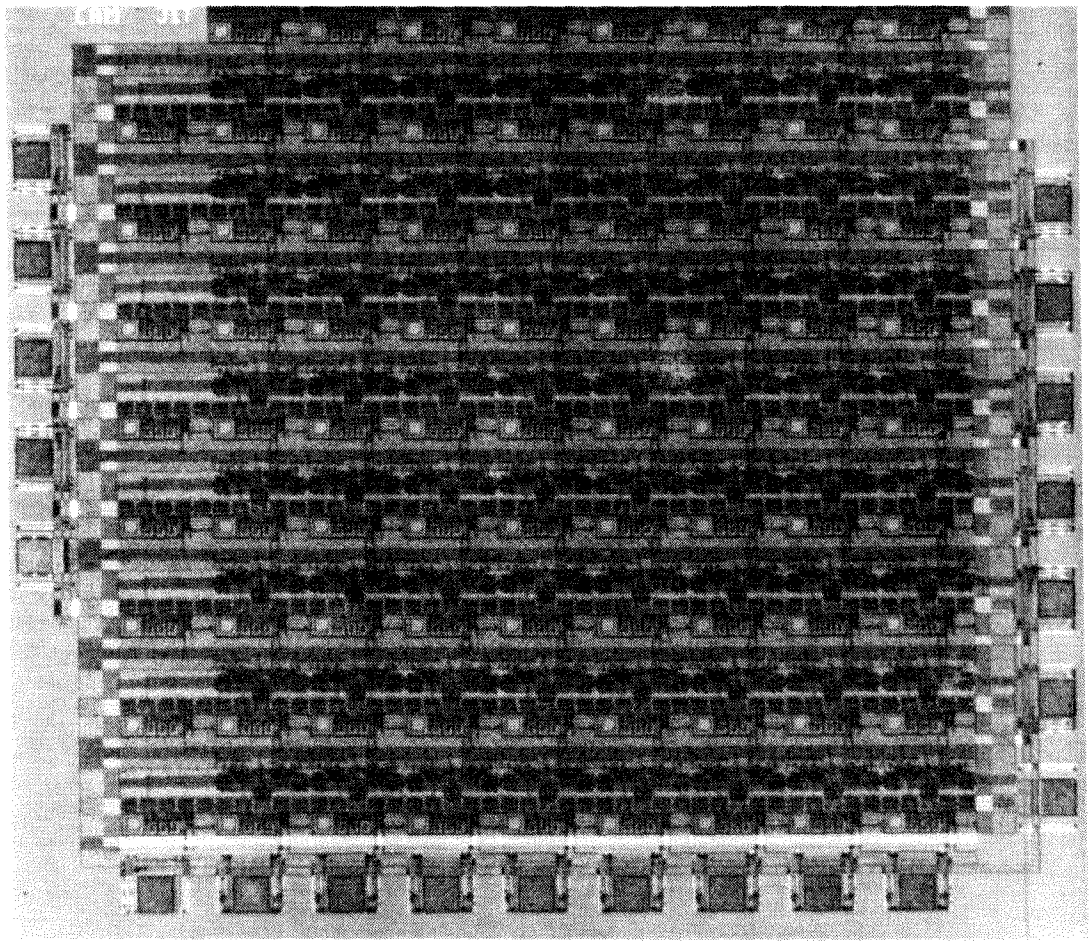


Figure 5.9: Photograph of the 8×8 array motion detection chip.

Chapter 6

Results

The 8×8 array chip has been extensively tested with actual moving images and by electronically simulating motion. Quantitative results of testing are reported below. The first set of experiments use an electronically controlled light source to apply an intensity field onto the chip that varies spatially across the chip and varies with time. The space and time derivatives of intensity are controlled by the experimental apparatus to simulate a moving intensity pattern while the velocity outputs from the chip are monitored. A second set of experiments focuses actual images onto the chips and measures the chip's response. The constraint line behavior is verified and the correction forces are mapped for different images. Finally, an interactive test set-up is described.

6.1 Characterizing the Motion Output

In the analysis for the motion detector, any changes in the light intensity were assumed to be due only to motion of the image, not to changes in the illumination level. By rapidly changing the illumination level under experimental control, the motion of a spatial intensity gradient can be simulated. The motion simulation test set-up is shown in Figure 6.1.

A time derivative is generated by changing the current through the LED light source. A triangle wave intensity, used in these experiments, makes a $\frac{\partial I}{\partial t}$ that is a square wave. The magnitude of the $\frac{\partial I}{\partial t}$ square wave is the slope of the triangle wave which is dependent on the amplitude and frequency of the triangle wave. Frequency is used to vary $\frac{\partial I}{\partial t}$.

An opaque screen between the LED and the chip that partially occludes the light causes a spatial derivative of intensity (edge) to fall on the chip. Moving the

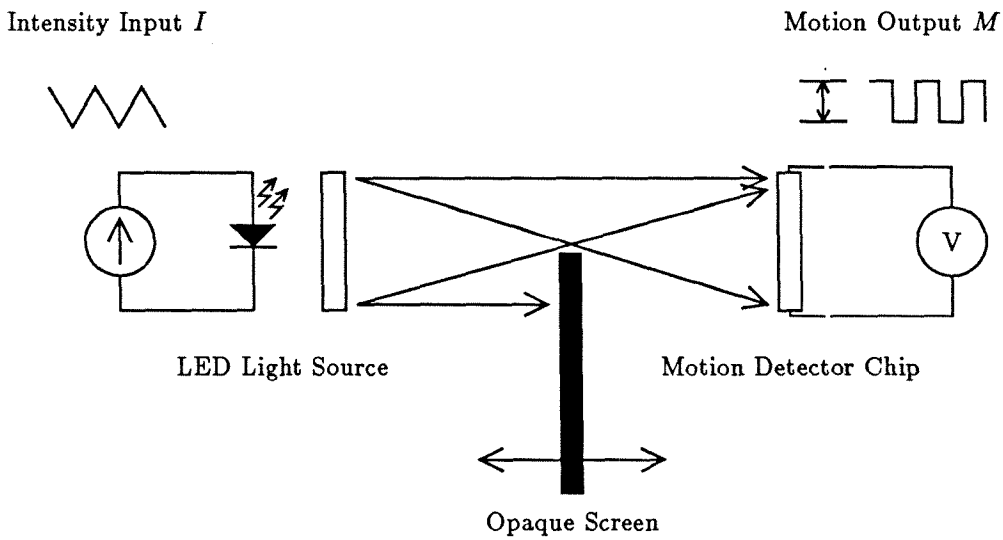


Figure 6.1: The test setup to electronically simulate motion. An LED casts light directly on the motion detector chip. Varying the LED current produces a controlled $\frac{\partial I}{\partial t}$. An opaque screen makes a shadow edge on the chip. The distance from the chip to the screen controls the sharpness of the edge, $\frac{\partial I}{\partial x}$.

screen closer to the chip makes a greater $\frac{\partial I}{\partial x}$, that is, a sharper edge. The position of the screen is adjusted until the measured spatial gradient is the desired value.

When a spatial intensity gradient that varies in time as a triangle wave is applied to the motion detector chip, the differential voltage on the chip's velocity outputs is a square wave. Figure 6.2 shows an oscilloscope trace of the LED input current and the velocity output of the chip. For these experiments, the screen producing the spatial gradient was aligned with the y-axis of the chip. As the intensities varied with time, the y-component of velocity reported by the chip was very nearly zero. The x-component of velocity is reported in the quantitative results.

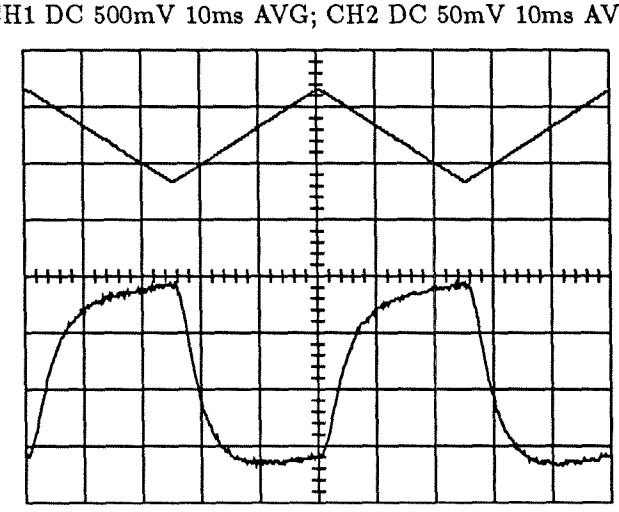


Figure 6.2: Oscilloscope traces of LED current (top) and reported velocity from the chip (bottom) for frequency of 20Hz.

With experimental control of $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial t}$, the velocity output of the chip can be tested to verify that the reciprocal relationship for velocity, $v = -\frac{\partial I}{\partial t}/\frac{\partial I}{\partial x}$, holds. Two sets of measurements are given. First, reported velocity as a function of $\frac{\partial I}{\partial t}$ is plotted for fixed values of $\frac{\partial I}{\partial x}$. A straight line graph is expected. Second, the triangle wave frequency generating $\frac{\partial I}{\partial t}$ is held constant and the spatial gradient is varied. The expected curve is a hyperbola. For all plots, the reported velocity is the amplitude of the square wave of the x-component output from the chip.

Figure 6.3 plots reported velocity versus $\frac{\partial I}{\partial t}$ (frequency) for three fixed values of $\frac{\partial I}{\partial x}$. The straight lines represent the theoretical proportional behavior as shown on the log-log plot. The experiment matches theory in the range from 1Hz to 40Hz.

Beyond that frequency, the amplitude of the motion rolls off.

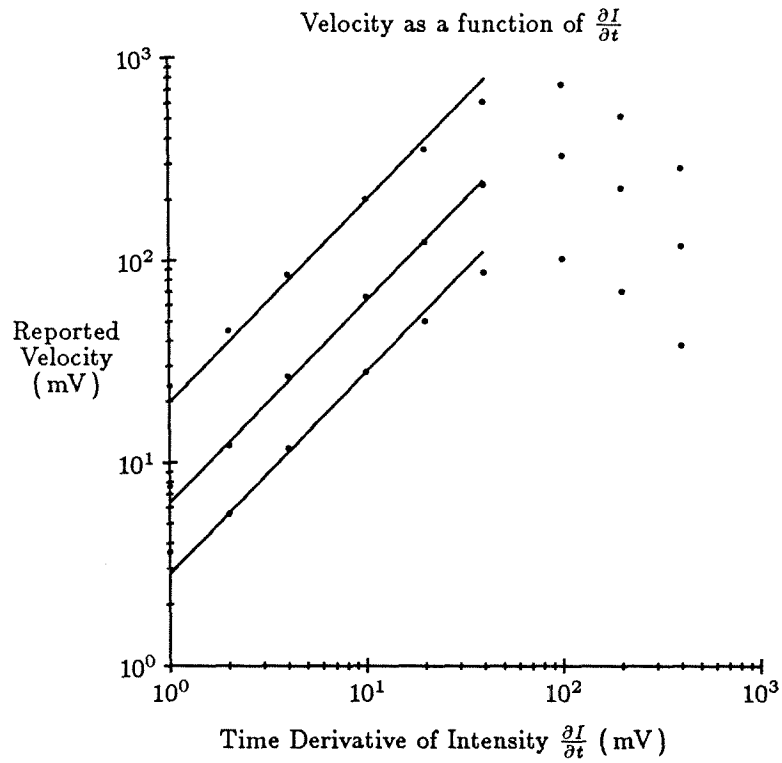


Figure 6.3: Measured motion response of the chip as a function of $\frac{\partial I}{\partial t}$ for fixed values of $\frac{\partial I}{\partial x}$. The straight lines represent an ideal linear response.

There are three circuit properties that can be responsible for the roll-off. First, the photosensor has a frequency limitation as reported by Mead [12]. This limitation is due to the charging of the capacitance of the first stage of the current amplifier by the photocurrent. This speed can be increased by operating at higher light levels.

Second, the frequency response of the differentiator has a roll-off frequency that is set by the τ knob as reported in Section 5.5. By increasing the control current into the integrator, the cutoff frequency can be made very large. The entire frequency response curve shifts, reducing greatly the signal levels produced by the differentiator for low frequencies. The operating frequency range of the differentiator is limited on the low end by noise levels and on the top end by either the unity gain of the differentiator at high frequencies and therefore the magnitude of the intensity change or the linear response of the differential pair circuits, whichever comes first. The

τ control current should be chosen so that the operating frequency range of the differentiator encompasses as much of the expected frequency range as possible.

Third, the charging of the capacitance on the global motion lines by the i_{OUT} current sets a limit on how fast a change in motion can be reported. This limit is the RC time delay of multipliers collectively charging the capacitance of the global motion lines. Note that this limitation is different from the first two because it is not a limit on how great a velocity can be reported but on how quickly a *change* in velocity can be reported. For these experiments, the applied intensity was driven by a triangle waveform generator. The reported velocity, as shown in Figure 6.2, approximates for suitably low frequencies the square wave that the theory predicts. For higher frequencies, the global velocity lines do not have enough time to respond to the apparent velocity change and in this slew-rate limited mode, become distorted from the desired square waves. The example of Figure 6.2 shows that even at 20Hz the chip starts to exhibit this departure from the ideal square wave. This limitation on the acceleration of the applied image can be affected by the current control on the multipliers that supply current to the global lines. Increasing the current provides a quicker reported velocity response at the expense of greater power consumption. Performance can be increased in this way until the operation of the multiplier exceeds subthreshold and becomes more non-linear. Figure 6.4 is a plot of reported velocity on the vertical axis versus applied $\frac{\partial I}{\partial x}$ on the horizontal axis for three fixed values of $\frac{\partial I}{\partial t}$. The curves approximate a hyperbola over most of their range. The behavior to the left of the peaks deviates significantly from a hyperbola and is investigated further below.

Recall that as $\frac{\partial I}{\partial x}$ decreases, the reported velocity should increase but that the resistance of the circuitry will increase, causing it to have less effect on the reported velocity. When $\frac{\partial I}{\partial x}$ becomes zero, the reported velocity could take on any value because it is not affected at all by the local cells. The current source loads in our implementation have large but finite impedance so that in the absence of any information from the visual field, the reported velocity will tend to zero. The schematic representing this effect is shown in Figure 6.5. For large $\frac{\partial I}{\partial x}$ (contrast ratios) the impedance of the local circuits is much smaller than that of the loads so their effect on reported velocity is negligible. As the contrast ratio is reduced, the load impedance must be taken into account.

So far we have referred to the output of the chip as the velocity v. Now, confronted with evidence that under some circumstances, the values on these global output lines may not be velocity, we will distinguish the chip's output by calling it "reported motion," M.

Since $M = \int i_{\text{TOT}} dt$, the condition for the system to be in steady state is

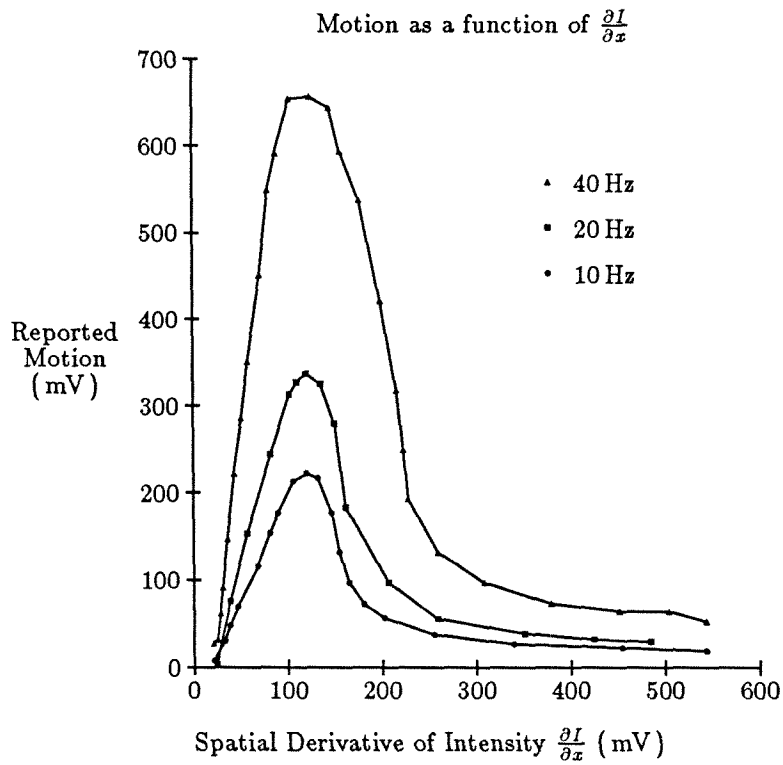


Figure 6.4: Measured motion response of the chip as a function of $\frac{\partial I}{\partial x}$ for fixed values of $\frac{\partial I}{\partial t}$. The response approximates a hyperbola over most of its range as expected for velocity. Near zero, the response is linear with $\frac{\partial I}{\partial x}$.

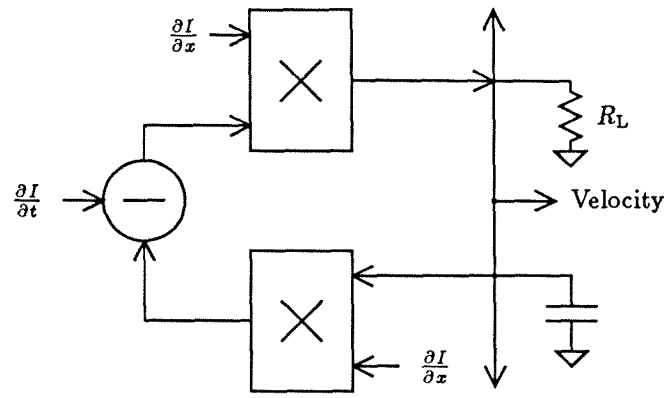


Figure 6.5: Schematic of the motion cell circuitry taking into effect the finite load impedance, R_L.

$i_{TOT} = 0$. Before considering the load impedance, $i_{TOT} = i_{OUT}$. Including the load impedance, $i_{TOT} = i_{OUT} - i_L$ so for steady state $i_{OUT} = i_L = M\frac{1}{R_L}$. The output current, i_{OUT}, produced by the divider circuitry is:

$$i_{OUT} = (M\frac{\partial I}{\partial x} - \frac{\partial I}{\partial t})\frac{\partial I}{\partial x}.$$

Steady state becomes:

$$M\frac{1}{R_L} = (M\frac{\partial I}{\partial x} - \frac{\partial I}{\partial t})\frac{\partial I}{\partial x}.$$

Solving for M we get:

$$M = \frac{\partial I}{\partial t}\frac{\frac{\partial I}{\partial x}}{(\frac{\partial I}{\partial x})^2 + \frac{1}{R_L}}. \tag{6.1}$$

A plot of this mathematical function is shown in Figure 6.6. For sufficiently large $\frac{\partial I}{\partial x}$, $(\frac{\partial I}{\partial x})^2 \gg \frac{1}{R_L}$ so the above equation reduces to:

$$M = -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}} = v \tag{6.2}$$

as we had before. As $\frac{\partial I}{\partial x}$ approaches zero so that $(\frac{\partial I}{\partial x})^2 \ll \frac{1}{R_L}$, the equation becomes:

$$M = \frac{\partial I}{\partial t}\frac{\partial I}{\partial x}R_L. \tag{6.3}$$

This analysis allows us to determine the chip's behavior for images with different contrast ratios. For sufficient difference between light and dark areas the motion

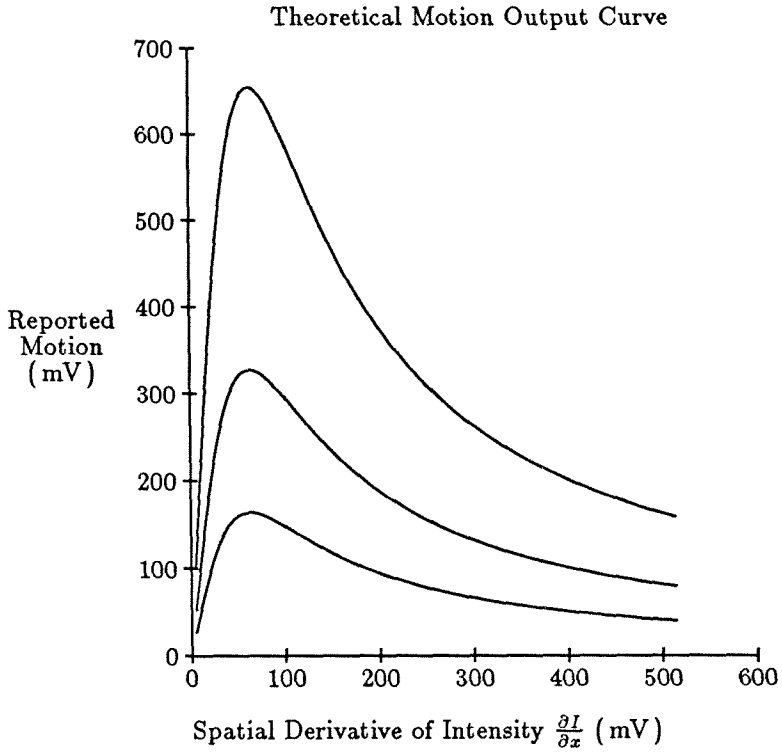


Figure 6.6: Plot of the theoretical motion response curve as a function of $\frac{\partial I}{\partial x}$ according to Equation 6.1. This curve approximates a hyperbolic response to the right and a linear response near zero.

detector chip will accurately report velocity. As contrast in the image is reduced, the motion output M will smoothly change to become a function proportional to both $\frac{\partial I}{\partial t}$ and $\frac{\partial I}{\partial x}$. There is evidence that some biological vision systems such as the fly's eye, utilize this motion M [18]. It seems a particularly graceful way for a system to fail as the contrast ratio in its field of view is reduced to the point where velocity can no longer be extracted.

Note that if the multiplicative definition of motion is desired over the entire operating range, the existing chip can be easily made to do this calculation. Setting the control current to zero on the feedback multiplier makes $i_{\mathrm{OUT}} = \frac{\partial I}{\partial t}\frac{\partial I}{\partial x}$. This current will be turned into a voltage by the load resistor or a higher performance current sensing arrangement could be built off chip.

In the absence of any information, at zero contrast, the motion detector chip will report zero motion. For a strict velocity detector, the zero contrast case is undefined so a device that reported "true velocity" could take on any value in the absence of information. Our chip behaves much better. It seems to be particularly useful that our motion sensor reports zero motion when it can detect no spatial intensity variation.

The stick and rubber band model needs a slight revision to include the effect of R_{L} on the circuit. Figure 6.7 includes an additional rubber band from the global velocity point to the origin. For images of sufficient contrast, the new rubber band is enough weaker than the others to be disregarded. As the image contrast decreases the new rubber band has more and more effect. When there is no information in the image, the additional rubber band pulls the global velocity to the origin. The effect the additional force has on ambiguous one-dimensional images is discussed in Chapter 7.

Over the complete range of $\frac{\partial I}{\partial x}$'s, and in particular in both the hyperbolic and linear regimes of motion, Equations 6.2 and 6.3 show that the magnitude of the motion response should be proportional to $\frac{\partial I}{\partial t}$. The three curves of Figure 6.4, above, taken at frequencies of 10Hz, 20Hz, and 40Hz seem to be scaled versions of the same curve and so bear out this proportionality over the range of $\frac{\partial I}{\partial x}$. Figure 6.3 shows a plot of reported motion as a function of $\frac{\partial I}{\partial t}$. The three curves are for fixed $\frac{\partial I}{\partial x}$'s, one chosen for each of the regimes of operation and one for midway in the transition region between them.

6.2 Verifying Constraint Line Behavior

The circuitry in each cell of the motion detector array was developed in Section 4.6 (Figure 4.9). The collection of circuits in each cell, working in concert, tries to

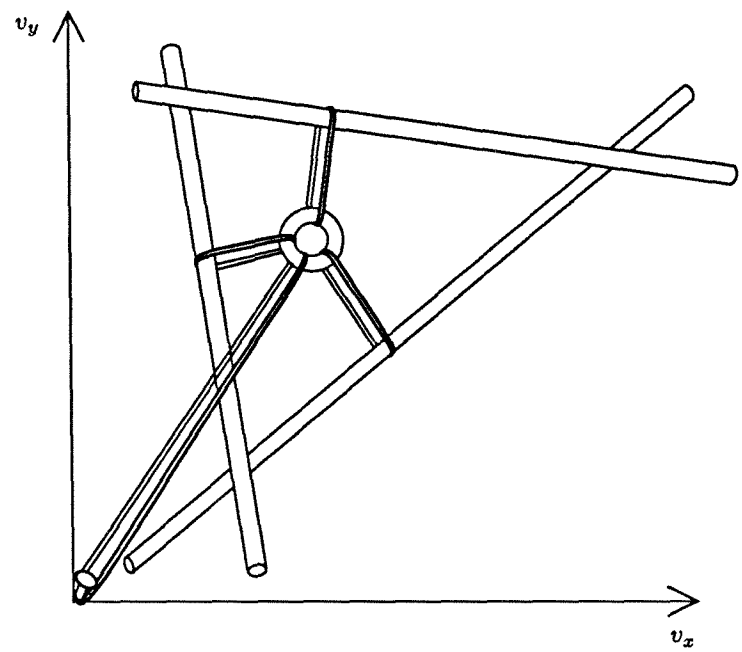


Figure 6.7: Physical model for the constraint solver with the addition of the tendency-to-zero effect.

satisfy the constraint between the x and y-components of velocity according to the line equation:

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0.$$

This constraint is defined by the inputs, the locally measured intensity derivatives, $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$, and $\frac{\partial I}{\partial t}$. If we force the value of one of the components of velocity, the circuit will drive the other component of velocity until its value satisfies the constraint. For a given image input the entire constraint line can be determined by sweeping the forced velocity value. Figure 6.8 plots three constraint lines from the measured response of the motion chip. A single edge was projected onto the chip so that the constraint lines of each cell in the array would all coincide. The x-component of velocity was driven to a sequence of values. For each value, the chip determined the y-component and the resulting point in velocity space was plotted. The image was not moving relative to the chip, so the constraint line should pass through the origin. The constraint line was plotted for three different orientations of the edge. To insure the relative angles of the three orientations, a single triangle was used as the image for each trial. Between trials, the part of the image falling on the chip was adjusted by translations only. Although the data deviates from the ideal slightly, this experiment clearly demonstrates the constraint line behavior of the motion detector chip.

6.3 Velocity Space Maps

To demonstrate the two-dimensional collective operation of the motion detector chip, we applied an image of a single high contrast edge to the chip at rest (zero velocity). The chip should report zero motion. The global output lines were driven externally to take on a particular sequence of values. The values were chosen to scan the velocity space in a regular grid. For each x-y pair of voltages driven onto the chip, the chip responded with a current intended to move the global point in velocity space into agreement with the velocity of its image input, namely zero velocity. These resulting x-y pairs of currents were measured for each point and displayed as a small vector originating at the forced point in velocity space. The resulting map of these vectors, Figures 6.9–6.11, shows in which way and by how much the chip is trying to pull the global velocity lines. The point of stability, the attractor point, is near zero as it should be. The amount by which the chip pulls as the global line gets further from the attractor depends on the structure of the applied image. A one-dimensional image such as the single edge used in this experiment provides information only about the velocity perpendicular to the edge. Thus the chip should pull harder when the velocity lines are forced away from the

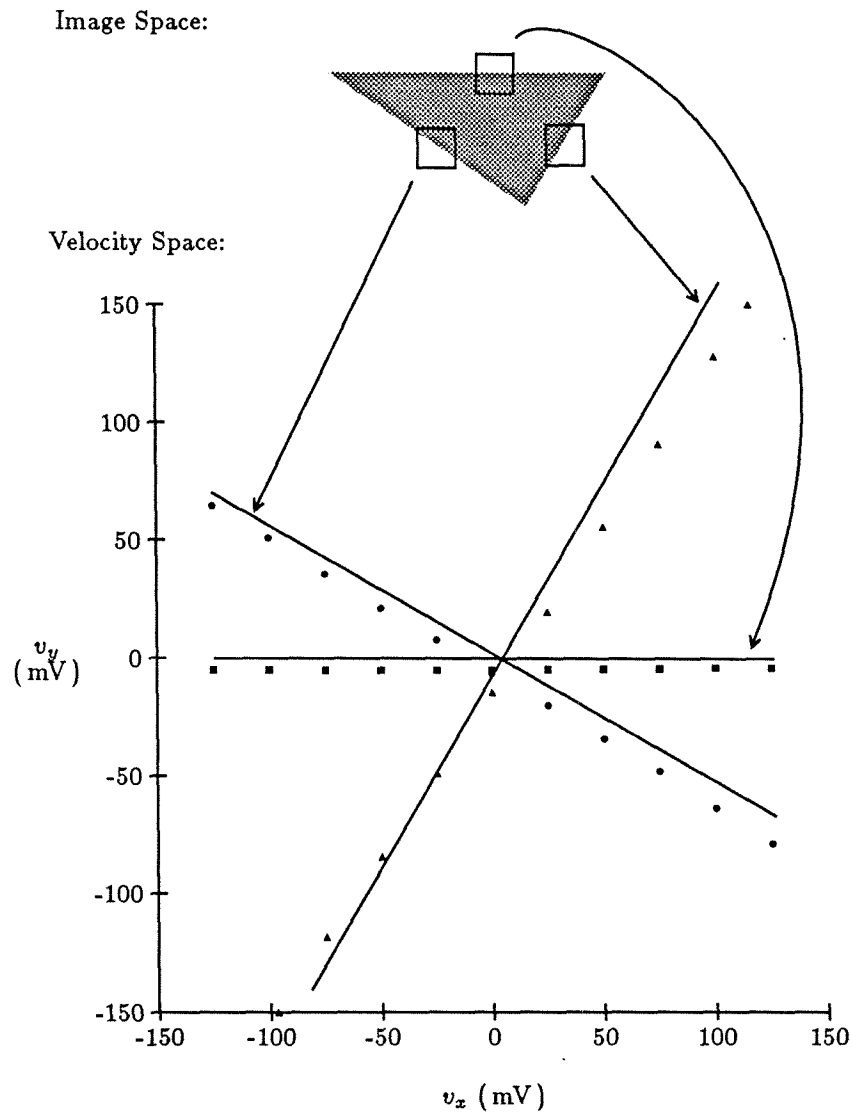


Figure 6.8: Demonstration of the constraint line behavior. The x-component of velocity was swept while plotting v_x versus v_y. The three trials were for the edge in the image oriented at 0°, 60°, and −30°. The lines are the ideal constraint lines.

real velocity in a direction perpendicular to the one-dimensional image stimulus. The image contains less information about the velocity parallel to the applied edge, so forced displacements of velocity away from zero in that direction result in much smaller restoring forces.

The motion detector has been extensively tested with the interactive arrangement shown in Figure 6.12. The chip is mounted on an x-y motion table. Motors translate the table in the x and y directions according to the position of a joystick control. The joystick is a velocity control—the velocity of the table increases with the distance the control is displaced from its center rest position. A photographic enlarger projects and focuses an image onto the chip. The analog velocity outputs from the chip are connected to the x and y channels of an oscilloscope that is set to display in x-y mode. The oscilloscope display represents velocity space and the dot on the display generated by the two analog inputs represents the reported velocity of the chip. As the joystick moves, the dot on the screen tracks the velocity of the chip and therefore tracks the position of the joystick. The speed range over which the chip can operate is more than adequate for use as a mouse pointing device.

6.4 Testing for Threshold Variations

Most of the circuits described depend on having matched transistors. The two most common requirements are that the two transistors of a differential pair have the same parameters and that identical circuits in adjacent computing cells behave the same so that the difference between their outputs is meaningful. As an example, the four-quadrant multiplier of Figure 6.13 has three differential pairs of transistors that must be matched to operate accurately. In addition, the top two pairs of the multiplier must be matched. For the output current of the multiplier to be scaled the same as for all multipliers, the current source transistor must match the current source transistor for all other multipliers.

The questions are:

- Is there a significant variation in transistors fabricated?
- If so, what is the effect of these variations on our circuit?

Data showing that transistors do have variations—some regular and some random—is followed by an analysis of what effect these variations will have on our circuit.

For test purposes, we fabricated an array of multipliers that could be individually tested. First, all the current source transistors were tested to see how well they matched throughout the array. Second, a simple test was performed for how

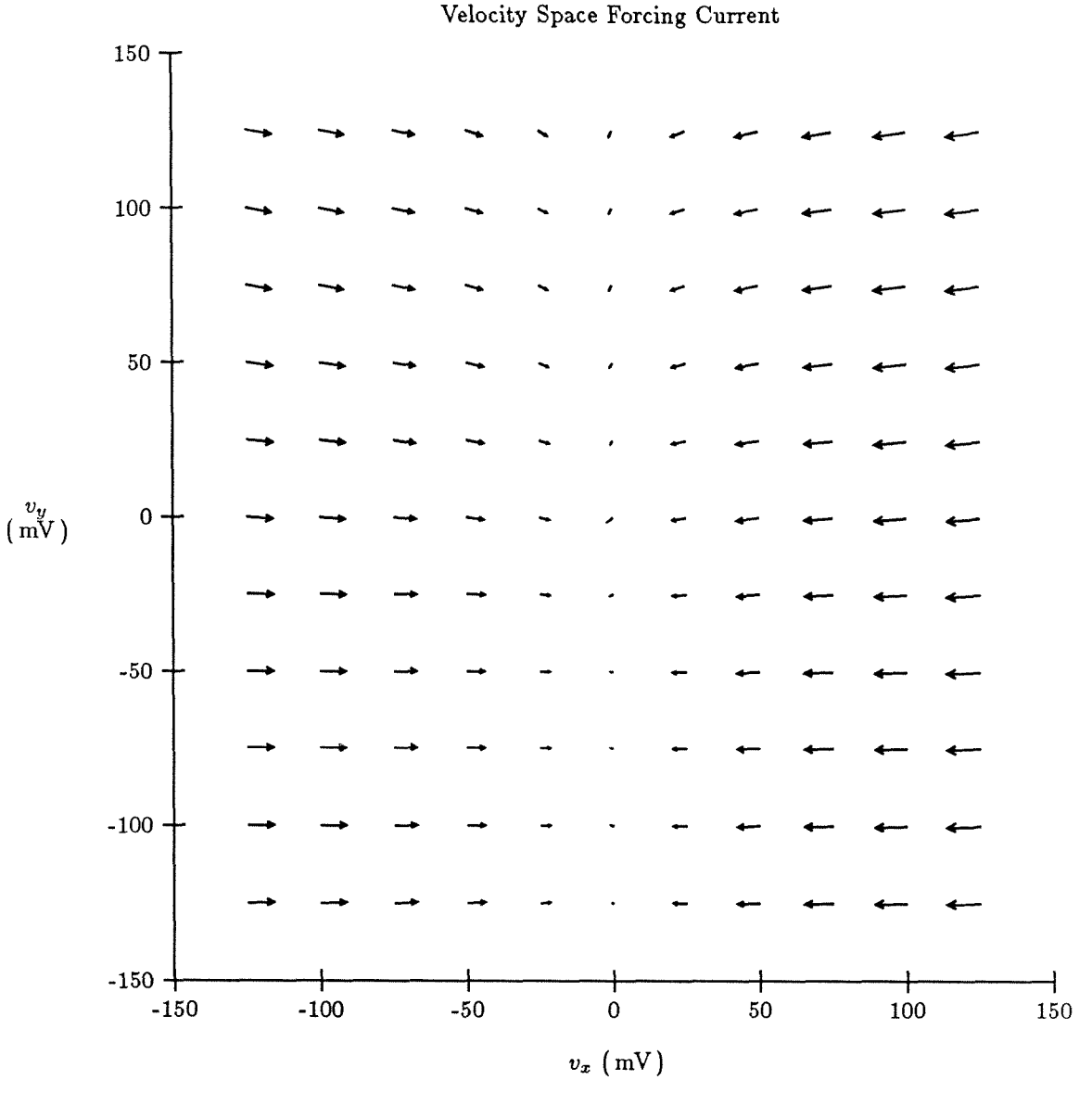


Figure 6.9: Velocity space map of the restoring forces generated by the motion chip in response to an edge at 90°.

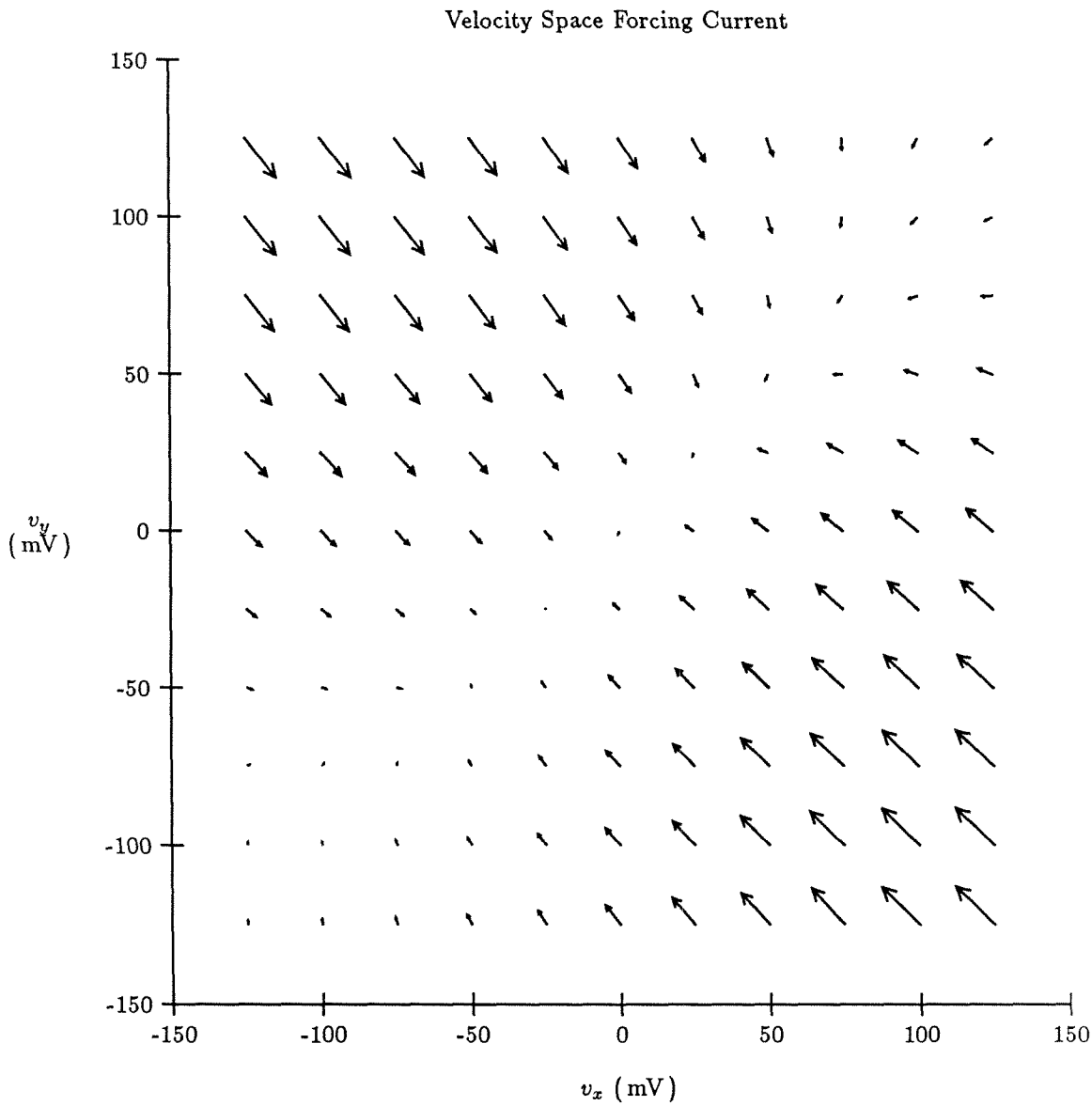


Figure 6.10: Velocity space map of the restoring forces generated by the motion chip in response to an edge at 45°.

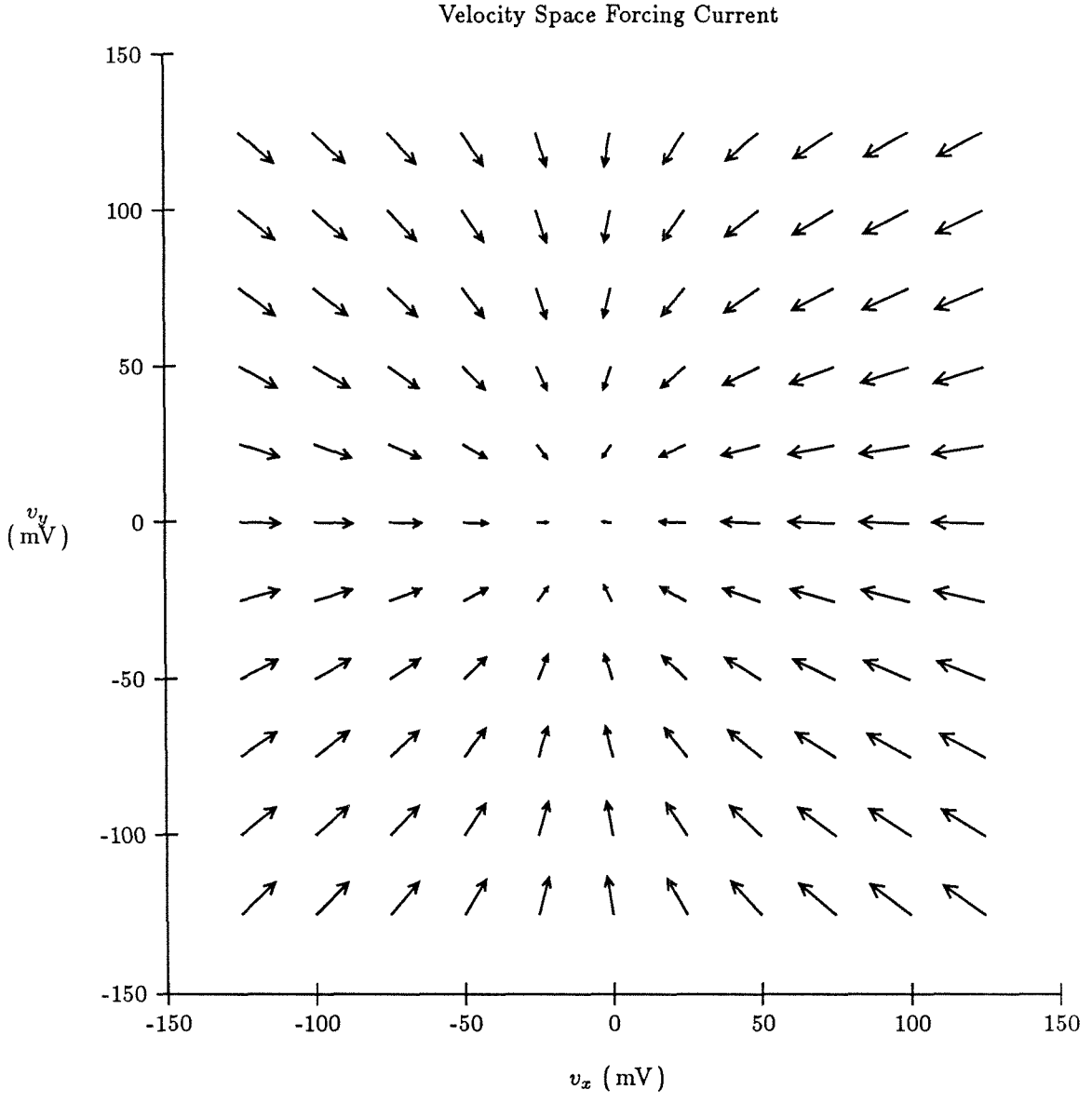


Figure 6.11: Velocity space maps of restoring force for a bright circle on a dark background. Edges of all orientations are represented.

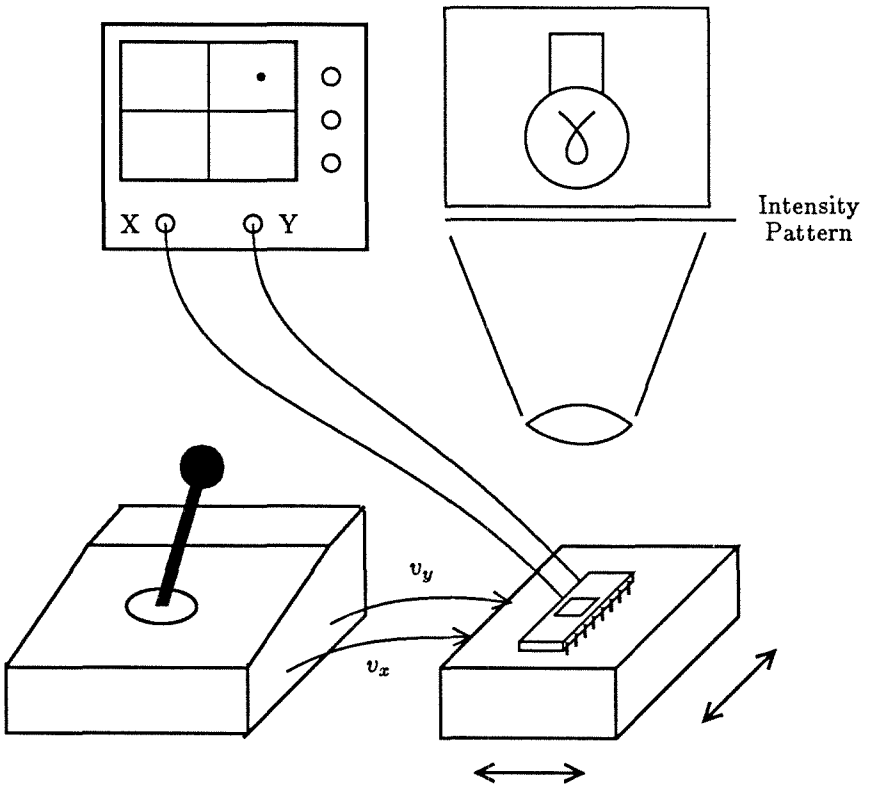


Figure 6.12: An interactive test set-up for the motion detector chip. The joystick controls the speed of an x-y motion table. The chip moves with the table under a fixed image projected from above. The chip outputs go to an oscilloscope to display reported velocity.

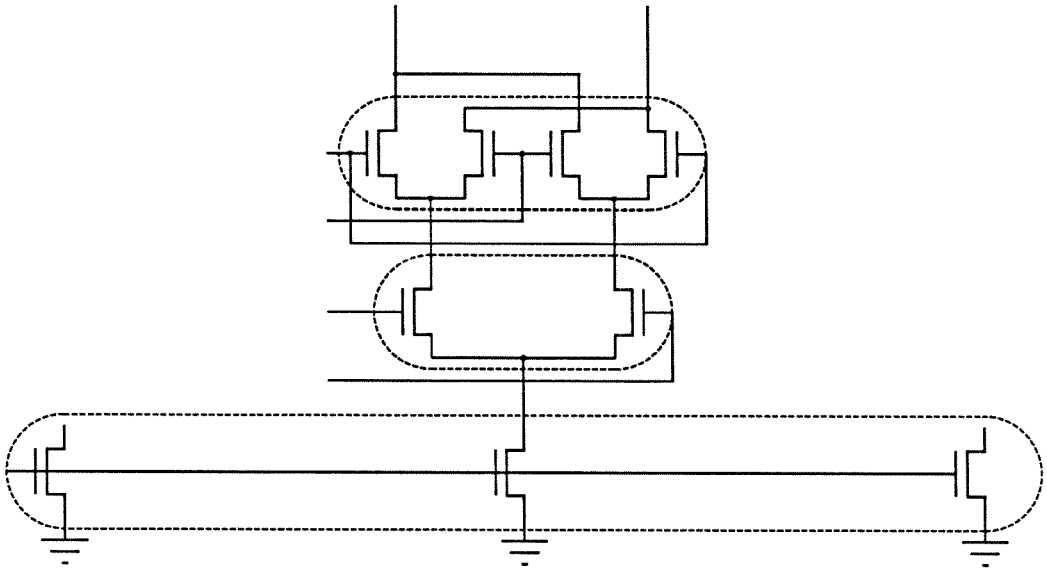


Figure 6.13: Multiplier transistors that must be matched. Each of the three pairs of transistors must be matched, the top two pairs must be matched to each other and the current source transistors of all the multipliers must be matched.

well matched were the transistors within a multiplier. The test circuit is shown in Figure 6.14. For both tests, the two differential inputs were set to zero with suitable common mode operating points. The gate of the current source transistor was set to 0.7 volts. The two output currents for each multiplier were recorded. The multipliers were arranged in a 6×9 two-dimensional array. One array had multipliers made with all transistors $2\times2\lambda$ and one with $16\times16\lambda$ transistors. Both were fabricated on a MOSIS 3 μm ($\lambda = 1.5\,\mu$m) run of a CMOS-Bulk process.

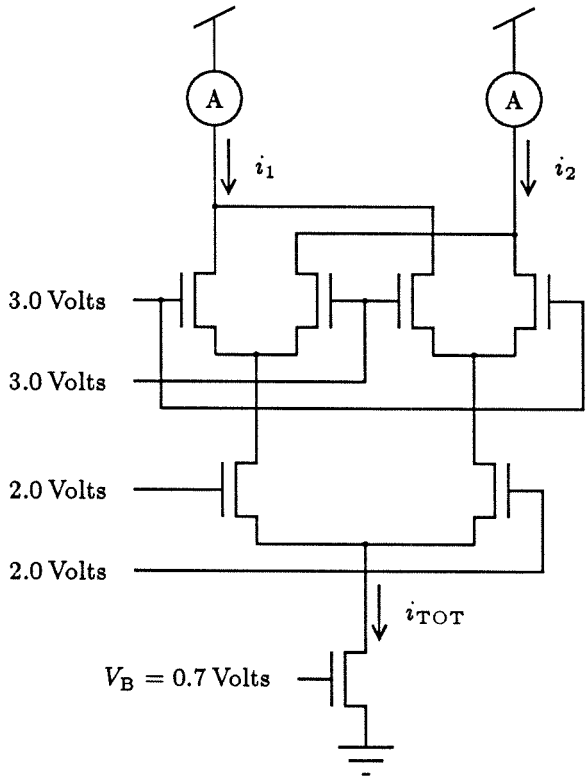


Figure 6.14: Transistor variation array test setup.

The sum of the two measured currents indicates the total current set by the bottom transistor. Figure 6.15 plots this total current as a function of position in the two-dimensional array. Each line is a scan across a row. Shown superimposed are the lines for the 6 rows. Besides noting a general upward trend in the envelope of these data lines, we see that there is periodic variation that repeats about every two or three multipliers (about 250λ). The same data is displayed in another way in Figure 6.16 to achieve a three-dimensional plot of control current as a function

of both spatial axes. To achieve this display each of the successive rows of data is offset vertically by the same increment. The absolute vertical axis scale is correct then for only the lowest line. We can then quite clearly see that along with some random variation there are ridges and valleys running diagonally across the chip at an angle of about 30° off from the vertical axis.

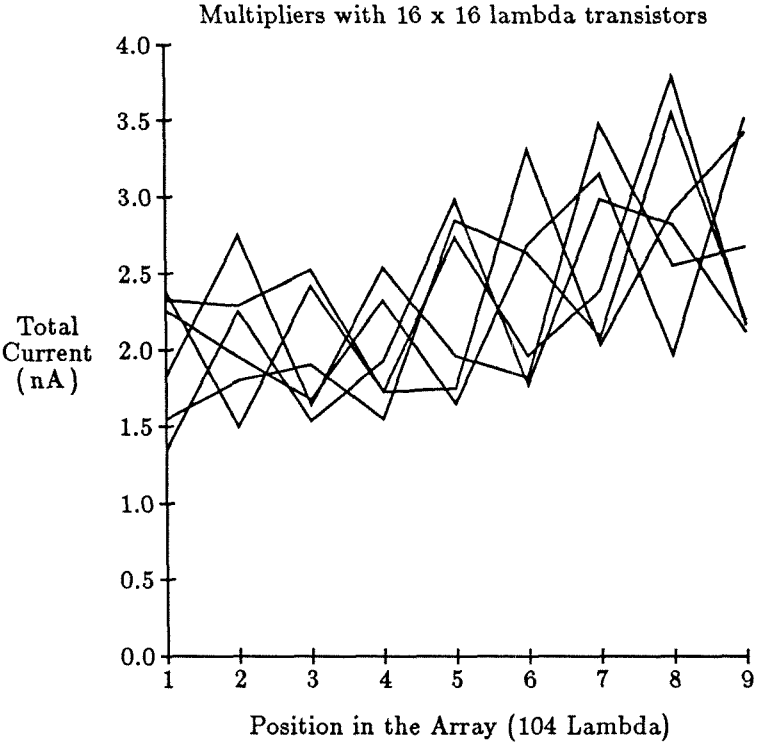


Figure 6.15: One-dimensional plot of total current as a function of position in the array.

These periodic variations in transistor parameters may be due to the coarseness of the raster scan used to implant impurities in the active layer of the silicon. If the direction of this scan relative to the axes of the designed chip is known, there is a straightforward way of eliminating the mismatch of transistors due to the scan. Each matching pair of transistors can be designed to lie next to each other parallel to the scan pattern. As less is known about the orientation and spatial frequency of the variation, more and more complex schemes can be designed to compensate at the expense of simplicity and chip area. Such compensation schemes have been

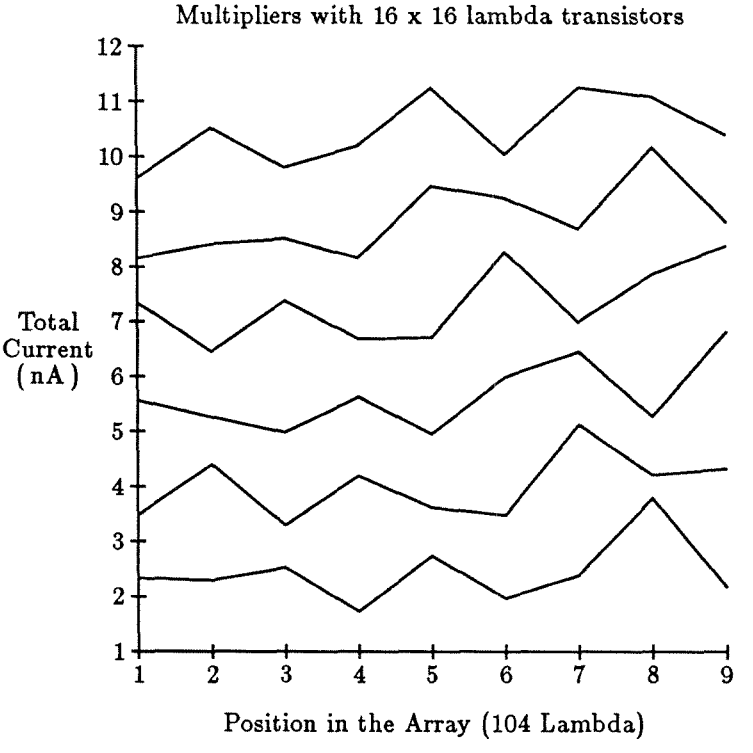


Figure 6.16: Two-dimensional plot of total current as a function of position in the array.

used to fabricate well-matched MOS capacitors [16]. We are currently pursuing another attack on this problem by opening a dialog with the fabrication service through MOSIS. Perhaps they can reduce this variation to below measurable levels by modifying their fabrication process.

To see what effect these transistor variations have on multipliers, the same test data was used in another way. Since the differential inputs to the multipliers in our test setup were set to zero, their differential current output should be zero. The currents i_1 and i_2 from the test setup shown in Figure 6.14 should be the same. The actual currents were measured and and the difference between the two currents, expressed as a percentage of the total current, were plotted as a histogram in Figure 6.17. We can see that a significant number of multipliers made with $16\times16\lambda$ transistors have errors of more than 40%. For the smaller transistors (Figure 6.18), the errors are considerably worse. These tests indicate that transistor variations are a real problem that must be considered by the analog designer.

6.5 Effects of Transistor Variations on the Motion Detector

A variation in the doping of a transistor results in a behavior that still follows the subthreshold relation of:

$$i_{\mathrm{DS}} = e^{kV_{\mathrm{GS}}},$$

but with a different threshold constant k. Differential pair circuits made with transistors with different thresholds will then behave the same as a circuit with matched transistors with a fixed DC voltage added to one of the differential inputs. This DC voltage is referred to as an offset voltage V_{OFF}.

We now analyze a simplified version of the divider circuit for its behavior in the presence of offsets. Figure 6.19 shows the schematic.

The equations describing the circuit become:

$$\begin{aligned} i_1 &= (\tfrac{\partial I}{\partial x} + O_{1A})(-M + O_{1B})G_1 \\ D &= i_1 + (\tfrac{\partial I}{\partial t} + O_2)G_2 \\ i_{\mathrm{OUT}} &= (D + O_{3A})(\tfrac{\partial I}{\partial x} + O_{3B})G_3. \end{aligned}$$

where O_x are the offset voltages. Substituting we get:

$$i_{\mathrm{OUT}} = (\tfrac{\partial I}{\partial x} + O_{3B})G_3[O_{3A} + (\tfrac{\partial I}{\partial t} + O_2)G_2 + (\tfrac{\partial I}{\partial x} + O_{1A})(-M + O_{1B})G_1].$$

M is constant when $i_{\mathrm{OUT}} = 0$ which occurs when:

$$\tfrac{\partial I}{\partial x} = -O_{3B}$$

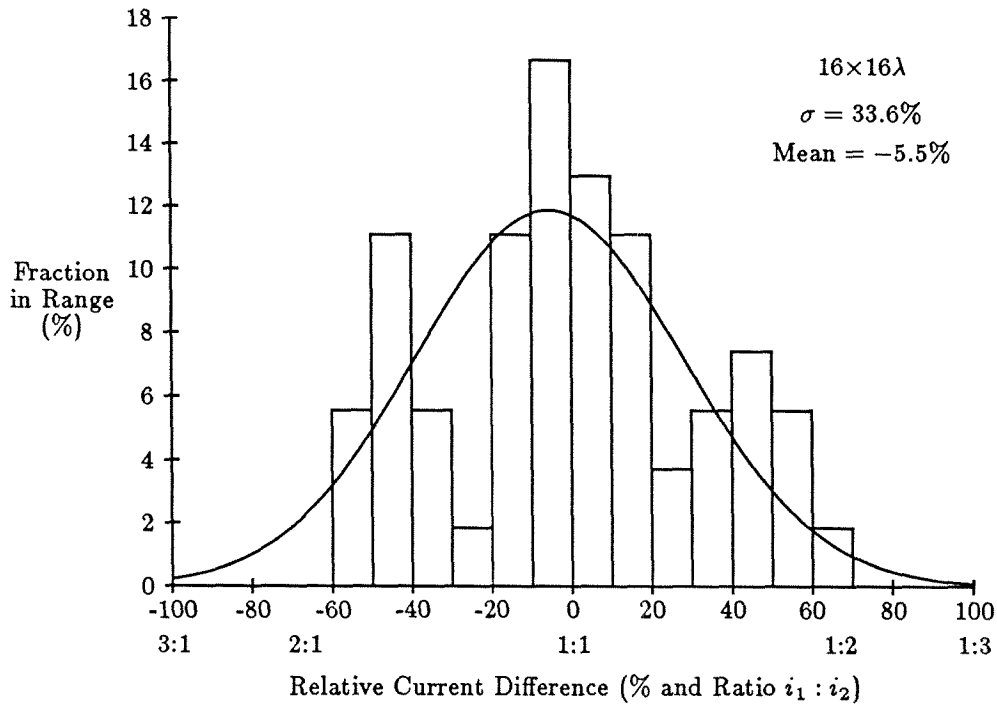


Figure 6.17: Histogram of the differential current output of the multipliers for zero differential input as a percent of average current. 16×16λ transistors.

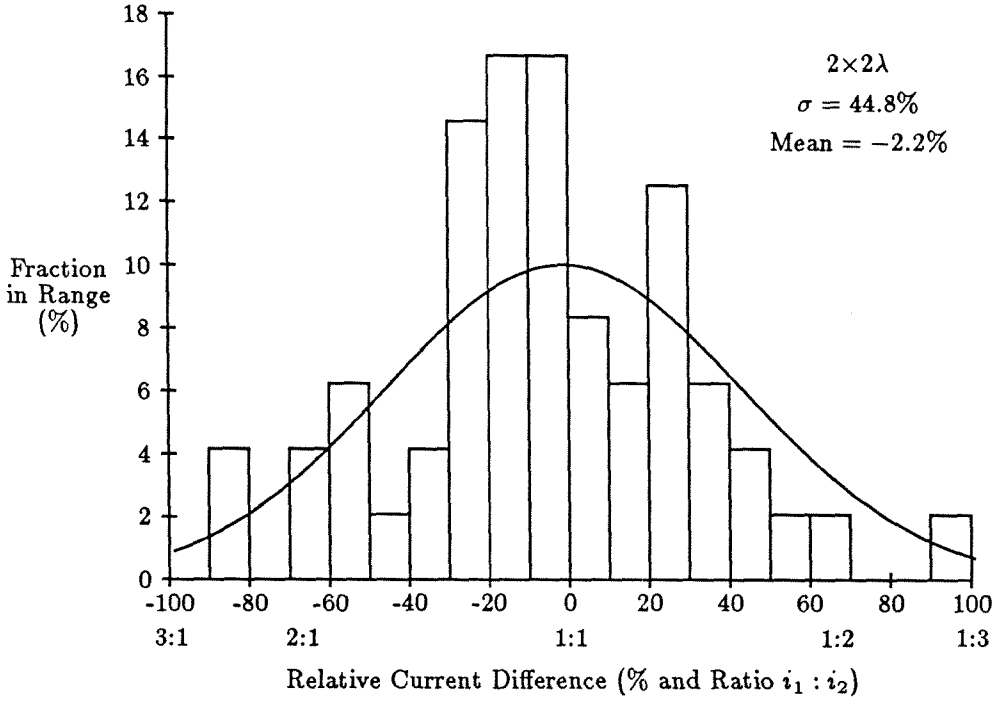


Figure 6.18: Histogram of the differential current output of the multipliers for zero differential input as a percent of average current. 2×2λ transistors.

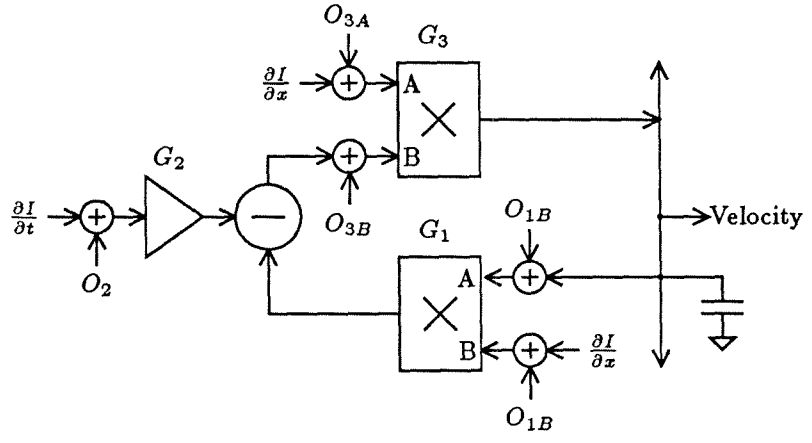


Figure 6.19: The divider circuit schematic taking into account offset voltages due to transistor variations.

or when:

$$M = \frac{O_{3A} + (\frac{\partial I}{\partial t} + O_2)G_2}{(\frac{\partial I}{\partial x} + O_{1A})G_1} + O_{1B}.$$

If all the offsets happened to be zero, and we ignore the various constants (as we have been doing all along) this equation reduces to the familiar:

$$M = -\frac{\frac{\partial I}{\partial t}}{\frac{\partial I}{\partial x}} = v$$

For non-zero offsets, their effect on the output depends on their position in the circuit. As is usually the case in feedback systems, any offsets within a negative feedback path show up in the output. This offset is not particularly good but could be worse—none of these offsets is multiplied by any of the gains throughout the circuit. The offsets that fall into this category are O_{1B} and O_{3A}. The O_{1A}, O_2 and O_{3A} offsets are not in any feedback loops and are therefore multiplied by gains in any amplifiers encountered on the way to the output including the amplifier with the offset in question.

Unamplified voltage offsets are generally acceptable. A current offset of 40%, as typical in the test data above, translates into a voltage offset of only 125 mV. For reasonable signal levels, this may not affect the signal to noise ratio significantly. Voltage values from the photosensors vary by about 320 mV per decade of light intensity. A voltage difference of 125 mV corresponds to a contrast ratio of about 2.5:1. For contrast ratios greater than 5:1, the measured distribution of voltage offsets may be acceptable. If these offsets are amplified, the result is generally significant and often devastating. Individual circuits in the motion detector cell by themselves do exhibit errors of this magnitude. These errors make it difficult to test parts of the motion cell such as a multiplier and load devices in isolation. Without the negative feedback, small offset voltages result in offset currents that as high impedance current sources tend to drive the outputs all the way to their limits near the power supply rails. Adding the feedback reduces the offsets to their unamplified levels in most cases.

The remaining offsets when feedback is added are in evidence for single motion cells but are barely noticeable in an 8×8 array. As the contributions from many local cells are aggregated, the signals due to intensity input add up, reinforcing each other, while the offsets due to random threshold variations tend to cancel each other out. This increased reliability and accuracy with larger arrays is an important property of the motion detector.

6.6 Summary of Results

An 8×8 version of the motion detector chip has been tested extensively in the lab. Electronically simulated motion was used to characterize the response of the chip. These tests showed that the chip reports velocity over a range of significant contrasts. With low contrasts the output gracefully degraded to another form of motion. As the contrast goes to zero, the reported motion goes to zero. Using real images projected onto the chip, the constraint line behavior of the motion algorithm was verified. Velocity space maps illustrated the correction force as a function of error distance. Threshold variations within a chip were demonstrated to be significant. Circuit feedback in some cases can reduce the effects of transistor variations. The aggregate property of large arrays improves the accuracy and reliability of the resulting motion output. output for

Chapter 7

Discussion

The following sections relate the velocity detector theory and implementation described in previous chapters to work in the other related fields of Hopfield neuron modeling, AI and computational vision, and biological vision modeling.

The correspondence problem of matching images in successive frames is avoided by both the motion sensing systems.

The velocity detector is an example of a Hopfield neural net [6,7]. The Hopfield neuron model is a recent powerful model for describing the behavior of highly connected neural nets. It is an important example in the emerging field of collective computation. Section 7.2 gives a brief description of the fundamental model and shows how the motion detector can be described in this model. Input-defined connection strengths, although a potentially powerful computing technique, are generally not considered by the Hopfield model due to their mathematical complexity. The motion detector is an example of a network with input-defined connections that is well behaved.

Motion detection (optical flow) has traditionally been done by the Artificial Intelligence community using conventional TV cameras and sequential digital computers. Some researchers are beginning to investigate and appreciate the power of analog networks for the solution of many early vision processes. In Section 7.3 I relate how the motion sensor chip fits into the computational vision paradigm and how the motion sensor architecture can easily be modified to accommodate some of the more complex motions studied by researchers in the computational vision field.

The implementation proves that a motion detector can be built using a very well-founded theory *and* using a simple, regular structure. Although there may not be a direct correspondence between the motion detector's electronic parts and neurons in a biological vision system, knowing the operation of the electronic version may allow the biologists to develop a new class of vision models that are more firmly based

on first principles than are the present ones. As an example, in Section 7.4, a well known psychophysical experiment is examined to see how my motion detector would respond—i.e., can the motion detector model explain the results of the experiment?

7.1 Avoiding the Correspondence Problem

The usual way researchers attempt to reconstruct velocity from image input is to match up features or objects in successive images. This approach has two problems:

- Features or objects must be extracted in advance. This prevents motion cues from being used in the feature or object extraction process.
- Matching must occur over large distances in the image, creating a difficult global problem out of an easy local one.

Advance feature extraction can be avoided by calculating velocity directly from the intensity information in the image. These direct techniques can only be used when the sampling rate is sufficiently high relative to motion in the scene.

The sampling process is also responsible for increasing the complexity of the problem. TV cameras generate images at a rate of 60Hz or 30Hz. In the time between successive frames, an object can move many pixels. Since the information associated with any intermediate positions is lost, the motion reconstruction algorithm must perform a match over an area large enough to cover the range of possible motions during the frame time. The coarseness of the sample makes a very difficult and computation intensive task from an inherently local problem.

The correlating motion detector of Chapter 2 avoids the correspondence problem by operating fast. The clocking rate must be high enough so that during one cycle time, the image does not move more than one pixel. This corresponds to a correlation window of 3 pixels. At room light levels the detector cycles at a rate of about 10KHz. This rate will handle image speeds of up to 2 meters/sec. To handle the same image velocity as this sensor, a system cycling at 60Hz would have to perform the correlation over a window 166 pixels wide. The slower system must then perform $166/3 = 55$ times the computing per cycle. For a two-dimensional detector, the computational load goes with the square of the linear window distance. The 60Hz system must perform 55^2 or more than 3000 times the computation of the faster system in each cycle. The computational bandwidth for the slower system must be 18 times that of the faster system. When more degrees of freedom of motion are considered, such as rotations or smoothly varying velocities, the space to be searched for matching features increases at a rate faster than the square of the window size.

The analog motion sensor also avoids the correspondence problem. Since this sensor operates continuously, the loss of information associated with discrete time sampling does not occur.

It is well known that two-dimensional velocity cannot be unambiguously determined from strictly local information. The need for longer range interactions does not require edge detection, feature extraction or the communication of individual intensity values over distances the scale of the interaction distance. The analog motion detector described in this thesis has none of these properties.

Data representation is the key to the efficiency of the motion detection algorithm. The velocity vector representing the result of the computation is also the means of interaction between cells. The global behavior of the motion detector is the solution of constraints from cells separated by a distance. This behavior occurs not because intensity values or individual local velocity values are communicated throughout and compared, but because each cell performs its own computation locally. Each local computation contributes to the final velocity result according to its local information. The velocity representation with its two degrees of freedom, the x and y components, allows the interaction of many cells each of which has only an ambiguous one degree of information. The local computation serves to reduce the raw data tremendously, with a rich long range interaction occurring over a very narrow communication channel. This channel of interaction is the set of wires (or resistor network) carrying the velocity vector.

The analog motion detector avoids the correspondence problem. The long range interactions necessary to disambiguate two-dimensional motion are solved by an appropriate choice of global data representation and local computation. Computationally intensive global pattern matching is not inherent in the motion detection problem and is not done by the analog motion detector.

7.2 Neuron Modeling and Energetics

The optical motion sensor shows some of the collective properties of a Hopfield neural net. Both systems exhibit a robustness which allows them to operate in the presence of defects and incomplete information. For both neural nets and the velocity detector, an abstract energy can be defined that is decreased by the system as time passes. The velocity sensing algorithm can be written below in terms of the same equations of the Hopfield neuron model so the motion detector is an example of a Hopfield neural net.

The two systems do have one important difference: static versus input-defined synapses. Following is a brief description of the Hopfield model and the motion

detector cast in this model, and a discussion of the importance of the difference between the two.

In the Hopfield model, the output of each neuron is connected to the input of each neuron by a set of connection weights or synapse strengths as shown in Figure 7.1. The set of N neurons defines the vector $\mathbf{V}$. For complete connection, there are N^2 connections or synapses each with its own connection strength or weight. Each neuron sums the outputs of all the other neurons according to its own connection weights. It then thresholds the resulting sum and outputs the mostly digital value. The system of neurons will, under the right conditions, fall into a stable state. The set of these possible stable states or "memories" is determined by the interconnection strengths. These strengths can be viewed as a matrix $\mathbf{T}$, and the memories as the fixed-points of the non-linear equation:

$$V = f(\mathbf{T} \cdot \mathbf{V} + \mathbf{U}),$$

where f is the thresholding function and $\mathbf{U}$ is a vector of threshold values [6].

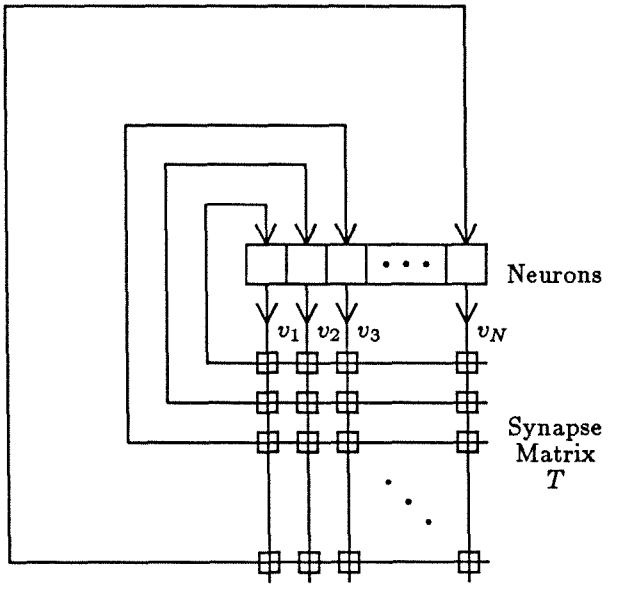


Figure 7.1: The Hopfield neural net model with vector $\mathbf{V}$ outputs from N neurons with feedback through the $\mathbf{T}$ matrix of weights.

In an elegant proof [7], Hopfield shows that for symmetric matrices $\mathbf{T}$, the system will converge to a fixed-point. This proof utilizes the idea of an energy for the system. The neuron network traverses its multi-dimensional state space ever decreasing its

energy until it reaches a fixed-point at a local energy minimum. The shape of this state space energy function and thus its minima are determined by the synapse matrix **T**. These connection strengths are presumed to be fixed or to change slowly with time by some learning process.

Figure 7.2(A) shows a cross-section of the state space energy diagram for the simplest two-neuron net, the flip-flop. It has two stable states that correspond to the two low points in the diagram. A flip-flop started near the middle, near but not at its metastable point, will progress away from metastability. As the flip-flop moves from the center, the slope of the energy function gets steeper. Correspondingly, the force increases and moves the circuit even farther in the direction away from center. The decision as to which of the energy minima the flip-flop will come to rest is determined long before the minimum is even approached. The only reason the system even has its fixed-points is because eventually the state variables approach the limits of their values near the power supply rails. The signals at this point can be conveniently thought of as digital values, 0 and 1.

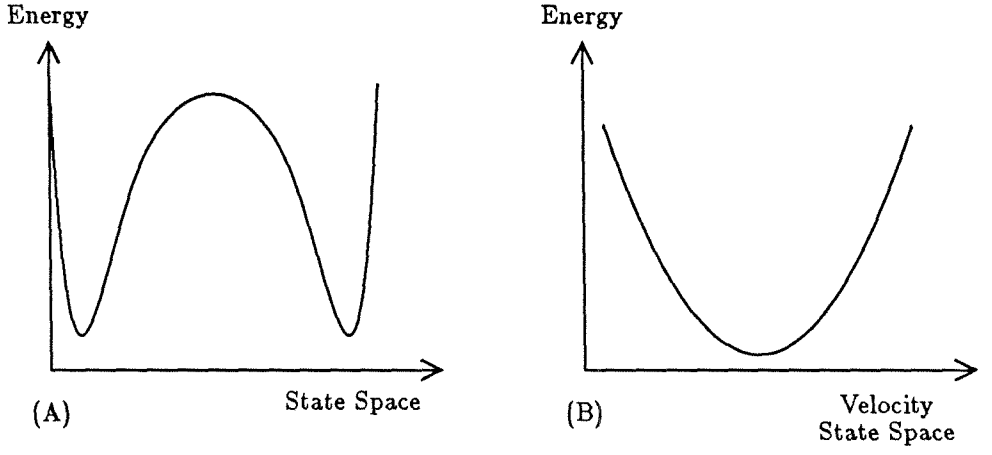


Figure 7.2: State space energy functions for (A) a simple Hopfield neural net and (B) a velocity detector.

In operation, the neural net must be cycled. It is first forced into some initial state near metastability and then released to fall into one of its stable states near the signal extremes. The inputs to the net determine the initial state of the system which biases the final state to be one that is "nearest" the initial state in state space.

It is straightforward to use an energy concept for the optical velocity sensor. The local computational elements of each cell produce a tendency for the global velocity

to move into closer agreement with its locally measured constraint line in velocity space. I loosely referred to this tendency earlier as a force. The idea now becomes a bit more formal. This force was defined earlier for one dimension as:

$$F = (\frac{\partial I}{\partial t} + \bar{v}\frac{\partial I}{\partial x})\frac{\partial I}{\partial x}.$$

This force defines the energy of the system. Energy is the integral of force over distance. Distance in this case is in velocity space, not physical space. For one dimension:

$$E = \int F\, dv.$$

Since the individual forces from the local cells are each linear, their sum is also linear and the energy is quadratic. The parabolic energy function has single minimum at the velocity as in Figure 7.2.

Expanding the physical analogy, the state of the global velocity can be represented as the horizontal position of a ball that rolls down the parabolic energy curve due to gravity and comes to rest at the low point. For a system with only a single cell, the low point or energy minimum will be at the local velocity. For a collection of cells working together, the minimum will be at the weighted average of all the local velocities. The circuits construct the appropriate energy well by computing the associated forces.

In two-dimensions, the forces produced by one cell make up the linear vector force field given by:

$$\begin{aligned} F_x &= (\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial x} \\ F_y &= (\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t})\frac{\partial I}{\partial y}. \end{aligned}$$

The energy becomes:

$$E = \int \mathbf{F} \cdot d\mathbf{v},$$

where $\mathbf{F} = \langle F_x, F_y \rangle$. The energy function is a two-dimensional curved surface and is generally a parabolic bowl as shown in Figure 7.3(A). An image that contains information equally in orthogonal directions will produce a circularly symmetric bowl. Images with non-symmetric information content will produce ellipsoid parabolic curves as in Figure 7.3(B). Any cross-section of the surface parallel to the velocity plane will be an ellipse with its long axis parallel to the direction of the greater number of edges in the image. In the special case of an ambiguous velocity of a one-dimensional image such as a stripe pattern, this bowl becomes a parabolic trough. Here, as shown in Figure 7.3(C), there are infinitely many lowest points, each of

which is consistent with the image information. A vertical cross-section of the velocity detector's energy function in any direction is a parabola. The curvature of such parabolas is not necessarily the same in all directions.

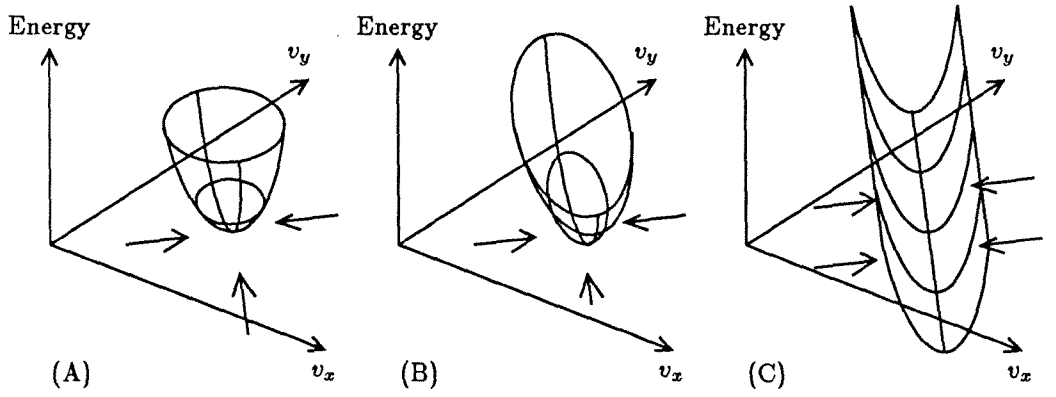


Figure 7.3: Velocity space energy diagrams for the velocity detector. Forces, shown as arrows, give rise to the parabolic energy surfaces. (A) Circular cross-section. (B) Elliptical cross-section. (C) Parabolic trough.

For the velocity detector, at any one time there is one and only one minimum in the energy function. For a Hopfield neural net there are many, one for every possible memory. On the other hand, a Hopfield net energy function stays fixed as the inputs change. The inputs only affect the initial position in the state space, not the energy function. For the velocity detector, the exact shape of the energy surface is determined by the input image as described above, but, except for the ambiguous case, has only one energy minimum. The position of the energy surface in state space moves as the input image changes velocity such that the energy minimum is always at the global velocity of the image. The reported velocity of the chip continuously tracks this energy minimum with no artificial clocking. The velocity detector has, over time, an infinite number of possible stable points which are directly a function of the input image.

The equations for the behavior of the velocity detector can be cast in the same manner as those for the Hopfield net. Feedback is a major part of both systems.

Each system has processing elements that take in the current state, perform a computation based on this state and additional information (either image input or predetermined connection weights), and produce a result that may affect the current state. An abstract energy can be defined for both systems that is only reduced by this computation. Energy minima correspond to fixed-points and both systems will move toward these fixed-points as their processing proceeds. The Hopfield fixed-point equation is:

$$\mathbf{V} = \mathbf{T} \cdot \mathbf{V} + \mathbf{U},$$

where the non-linear thresholding function has been changed to the linear identity function. Although in the real velocity detector chip, non-linearities can occur and for various practical reasons can be quite useful, this analysis works fine for linear systems. The operation of Hopfield neural nets depends on the non-linearities to come to a fixed-point.

For the velocity detector, velocity represents the state of the system so the state vector, $\mathbf{V}$, above corresponds to the velocity vector, $\mathbf{v}$, so:

$$\mathbf{V} \equiv \mathbf{v} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}.$$

For the velocity detector the fixed-point equation becomes:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} 1 + \sum \left(\frac{\partial I}{\partial x}\right)^2 & \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & 1 + \sum \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} + \begin{bmatrix} \sum \frac{\partial I}{\partial t}\frac{\partial I}{\partial x} \\ \sum \frac{\partial I}{\partial t}\frac{\partial I}{\partial y} \end{bmatrix}, \tag{7.1}$$

where the summations involving $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$, and $\frac{\partial I}{\partial t}$ are occurring over the array of cells each with its own local derivatives obtained from image inputs.

Alternatively, the local variables, one in each cell, could be included in the state space vector. These local variables are an intermediate result in the computation performed in each cell and represent the error distance, D, between the global velocity and the locally determined constraint line. Expressing the local intermediate distance variables, D_i as state variables, the fixed-point equation becomes:

$$\mathbf{V} \equiv \begin{bmatrix} v_x \\ v_y \\ D_1 \\ D_2 \\ \vdots \\ D_N \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{\partial I}{\partial x_1} & \frac{\partial I}{\partial x_2} & \cdots & \frac{\partial I}{\partial x_N} \\ 0 & 1 & \frac{\partial I}{\partial y_1} & \frac{\partial I}{\partial y_2} & \cdots & \frac{\partial I}{\partial y_N} \\ \frac{\partial I}{\partial x_1} & \frac{\partial I}{\partial y_1} & 0 & 0 & \cdots & 0 \\ \frac{\partial I}{\partial x_2} & \frac{\partial I}{\partial y_2} & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial I}{\partial x_N} & \frac{\partial I}{\partial y_N} & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ D_1 \\ D_2 \\ \vdots \\ D_N \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{\partial I}{\partial t_1} \\ \frac{\partial I}{\partial t_2} \\ \vdots \\ \frac{\partial I}{\partial t_N} \end{bmatrix}. \tag{7.2}$$

The Hopfield proof of fixed-points is one of the major strengths of the neural net model. The proof makes somewhat tractable a potentially very difficult problem—the behavior of an arbitrary fully connected non-linear neural net. The proof gives us one important piece of information, stability, about a large set of neural nets—those with all symmetric connections. The same stability is the reason the neural net must be cycled for continued operation. During the active phase of the cycle, the range of state space values a neural net can take on decreases with time until it reaches a fixed-point where it would sit indefinitely until the next reset phase. Compare this behavior to that of a velocity detector that can remain near a fixed-point at all times yet traverse all of state space continuously and indefinitely as the inputs change. The velocity detector example illustrates the power of state space energy terrain defined by inputs.

The usual method of cycling a Hopfield neural net is to introduce an additional mechanism such as in Figure 7.4(A) to force the state of the net out of its minima. This forcing is done repeatedly forming a simple two-step cycle of force-release. It is interesting that to make a net useful, a cycle must be introduced, since the strength of the Hopfield proof was that the fixed point existence eliminated the possibility of cycles. As beautiful a system as a Hopfield net is, one must go outside the system in order to make it useful. The excursion outside the system is the addition of the circuitry to reinitialize the state in a cyclic manner.

We can, of course, view the reset circuitry in the framework of a state space energy function. The reset circuitry alters the state space energy terrain by removing the feedback and thus the fixed point behavior. This circuitry temporarily puts the system into a mode where the internal state tracks the input. The energy diagram for one possible input is shown in Figure 7.4(B). This is the simplest possible input driven connection function—unity. During the next phase of the cycle, the reset circuitry reconnects the feedback, restoring the fixed-point stuck kind of behavior. During the reset phase, even the simplest useful neural net has input defined fixed-points although it is usually not thought of in this way.

Fixed-point behavior is important. Random walks through state space are generally of very limited usefulness. Points in state space that are not only fixed-points but also have all neighboring points tending toward them are called attractors. Metastable points are fixed-points but are not attractors. The existence of attractors is important. The velocity detector has one attractor at all times, neural nets have attractors—many during the memory phase and one during the reset phase. Allowing the locations of these attractors in state space to change as a function of inputs to the system is a powerful extension to the neural network paradigm. This extension formalizes what is done outside the Hopfield model during the reset phase, and allows us to go beyond the simple two-phase network cycling scheme and

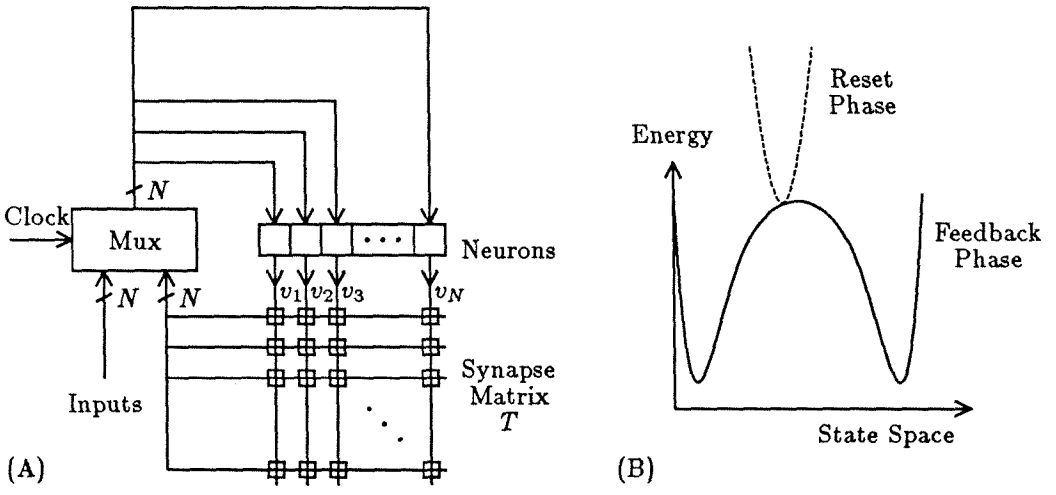


Figure 7.4: (A) Neural net with circuitry to cycle it. (B) Energy function for the reset phase shown as dotted line.

develop systems that smoothly traverse state space in response to inputs, with no artificial clocking imposed.

The bad news about input-driven connection matrices is that the nice stability property of the fixed-weight nets is not guaranteed. In general, if the connections are defined by the inputs, they may be symmetric and therefore yielding a stable network for all sets of inputs, some sets, or none at all. A system that is stable sometimes but unstable for some input combination seems very unsettling. In fact, with some unstable networks, energy is not defined. When a ball can continually run downhill and return to where it has been before, there is no longer a well-defined energy terrain.

The easiest approach to the stability problem is to retro-fit the new input-defined connections into the old stability criterion: Make sure that even though the inputs determine the connection matrix that they always make a symmetric one. The velocity detector algorithm uses this method. Examination of Equations 7.1 and 7.2 reveal that both matrices are symmetrical about the diagonal. This self-transpose property is another way of saying that the connection between any pair of neurons is the same going both ways. This symmetry can also be determined from the examination of the architecture of the velocity detector. All circuits are symmetrical

with respect to v_x and v_y.

Another method of achieving stability is to appeal to the special properties of the problem at hand. For example, in the case of the velocity detector, the network was designed to consist of one large attractor at the minimum of a single convex energy bowl. This property, along with an additional tendency to zero to handle the ambiguous case, results in a stable system. Admittedly, the velocity detector is a low dimensionality problem compared to even the smallest Hopfield neural nets considered, but the elegant smooth nature of the solution suggests that the approach may be well worth the extra effort required to examine its stability.

7.3 Artificial Intelligence and Computational Vision

A developing field in artificial intelligence is computational vision. One of its main goals is to develop systems that construct scene descriptions from input images. Computation of optical flow, the apparent velocity at each point in the image, is one of several early vision processes that extract some property of the scene from low level image data. Poggio, Torre and Koch [17] show that most of the early vision problems are ill-posed but that they can be made well-posed by adding *a priori* knowledge often in the form of a variational principle. A unified mathematical structure emerges called variational regularization theory that can be applied to early vision problems including optical flow.

The ambiguity resolution of my velocity detector is a well-posed problem for rigid translational motions in the absence of noise. The presence of noise, minor fabrication defects or small deviations of the image from strict translation cause the constraint lines in velocity space to converge on a region instead of intersecting at a point. Determining a velocity point from an approximate region of intersection is ill-posed. The variational principle of the energy associated with the rubber-band solution makes the problem well-posed.

Two important classes of image motion form extensions to rigid translation—those with smoothly varying velocities across the image and those with velocity discontinuities. An example of motions in the first category are rotations in the plane, that together with translation make up rigid motion within the image plane. Scalings of the image such as those that result from moving closer to or farther from the viewed object, also produce smooth optical flow fields. Smoothly varying motions can also result from the projection onto a two-dimensional image of three-dimensional objects rotating in three-space. Velocity discontinuities, however, arise from objects moving at different speeds occluding one another. For example, an object moving in front of a fixed background will have velocity discontinuities along

its boundaries in image space. Both of these types of more complicated motions can be handled by relatively minor extensions to the velocity detector architecture.

Horn and Schunck [8] and Hildreth [5] develop smoothness criteria for the 3-D motions projected onto 2-D images that turn them into well-posed problems that they solve by iterative solution methods. The area based criterion, which is most amenable to the parallel solution, is cast by Poggio *et al.* in the form of the regularization principle:

$$E = \int \left[\left(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t} \right)^2 + \lambda \left(\frac{\partial v_x}{\partial x}^2 + \frac{\partial v_x}{\partial y}^2 + \frac{\partial v_y}{\partial x}^2 + \frac{\partial v_y}{\partial y}^2 \right) \right] dx\, dy,$$

where λ is the regularization parameter and E is the total energy to be minimized by the system.

This formulation seems to allow a straightforward extension of the algorithm and architecture that detects only translational velocity. The first term of the equation, $\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}$, is a familiar quantity, the weighted distance of the velocity guess to the local velocity constraint line. The second part of the equation, $\frac{\partial v_x}{\partial x}^2 + \frac{\partial v_x}{\partial y}^2 + \frac{\partial v_y}{\partial x}^2 + \frac{\partial v_y}{\partial y}^2$, incorporates a smoothness cost function and requires the use of the difference in velocity components between neighbors in x and y directions. λ is the relative weighting of the smoothness criterion to the constraint-line criterion.

Poggio goes on to suggest that analog networks could be used to solve some of these regularization problems. The velocity detector already described in this thesis is an example of such a network. We have already designed, fabricated, tested, and shown in the laboratory a working velocity detector based on a simple regularization principle and utilizing an analog network implemented with active CMOS circuits. This integrated velocity sensor uses a simple regularization principle needed due to the noisy data of real world images. The more advanced regularization principle based on smoothness is a simple extension of the velocity detector architecture and is described below.

Figure 7.5 shows a proposed extension of the velocity detector architecture for smooth optical flow. What had been a single global velocity distributed to all cells is now a network of resistors. The resistors allow the velocities at each cell to be different while at the same time minimizing the departure from smoothness of the velocity values on the nodes between resistors. The force equations become:

$$\begin{aligned} F_x &= (v_x - \bar{v}_x) + \left(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}\right) \frac{\partial I}{\partial x} \\ F_y &= (v_y - \bar{v}_y) + \left(\frac{\partial I}{\partial x} v_x + \frac{\partial I}{\partial y} v_y + \frac{\partial I}{\partial t}\right) \frac{\partial I}{\partial y}, \end{aligned}$$

where $\bar{v}_x$ and $\bar{v}_y$ are the average velocities of the four neighboring nodes. These forces will minimize the local deviation from the constraint line and larger scale departures

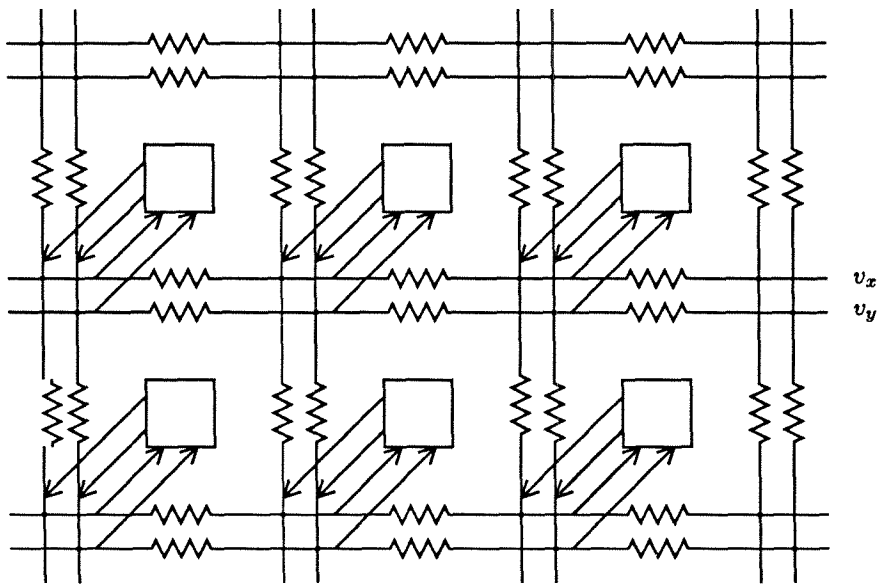


Figure 7.5: An extension to the velocity detection architecture for computing an array of smoothly varying optical flow. Each intersection point in the array represents the velocity field at that point. This velocity is computed by the same local circuitry as before, with the collective effect of neighboring cells reduced with distance away by the resistor network.

from smooth velocities. These equations follow from Horn and Schunk and use a somewhat simpler approximation for the Laplacian than theirs as a criterion for smoothness.

The simpler design described in previous chapters had a single global velocity that was distributed throughout the array. This architecture allowed only one velocity to be reported and had all local velocity information interacting globally with that of other cells regardless of distance away. The extension to the architecture has an array of nodes, where each node represents the local velocity. The rubber band model of constraint solving still holds for every node, but the strength of the force exerted by neighbors is reduced as the distance between the interacting cells increases. A slight practical problem is introduced by this velocity network. There are now an array of possibly different velocities that need to be communicated to the outside world. Until such time as the velocity map can be used by the next level of processing integrated on-chip, we must be content to monitor the array of velocity values off-chip. Since the pin limitation and wiring costs prohibit using a separate wire for each velocity value, circuitry to scan out the array sequentially can be integrated on-chip [21].

The resistors between cells can be implemented as CMOS active circuits as described by Mead and Mahowald [14] and reproduced in Figure 7.6. Here V_1 and V_2 are the voltages on two adjacent nodes of the network. A pair of current mirrors causes the current i_0 into the upper node to match the current out of the lower node. Any current out of node 1 must then flow into node 2. By symmetry, this current must be zero when the two voltages are equal and will be monotonically related to the voltage difference, $V_1 - V_2$. The value of the effective resistor implemented by this circuit is set by the current i_0 which is controlled by the current mirror input.

The second class of motions that are very important give rise to discontinuities in the optical flow. The flow field for an object moving with respect to a fixed background will have step discontinuities along its boundaries. These steps can be very useful in later stages of processing to aid in determining the object boundaries. If they are to be used in this way, the early vision optical flow extraction must preserve the discontinuities instead of forcing them into the mold of smooth flow. Koch [9] tackles a similar problem, that of reconstructing a smooth surface from sparsely sampled data while preserving the discontinuities of the surface. The basic idea is to allow neighboring values to interact in a way that yields smooth interpolation between them until the difference becomes great. At that point, a discontinuity is detected which causes the neighbors to be disconnected from each other. Then neighboring values can be quite different without affecting each other, as should be the case for two values straddling a step discontinuity. Koch proposes a hybrid digital-analog system to perform the computation using a cycled two-phase

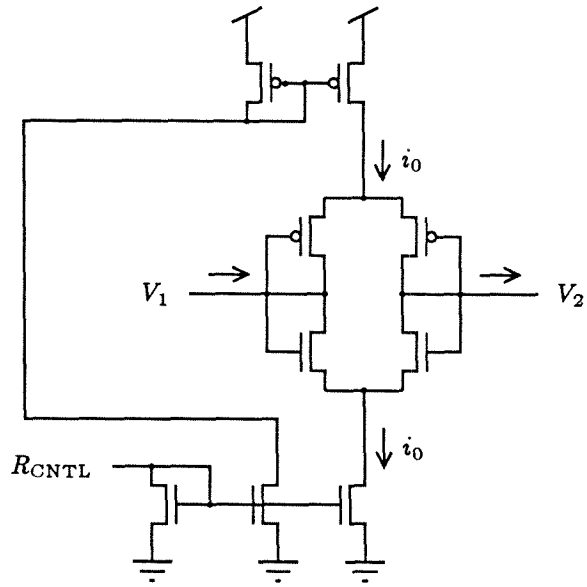


Figure 7.6: A CMOS active circuit used as a resistor.

approach with A/D and D/A conversion between phases. Alternatively, the integrated architecture of the velocity sensor could be easily adapted to perform this extension to the original computation. As well as solving the linear aspects of this problem, CMOS analog circuits are well suited to computing the necessary non-linear functions.

To allow discontinuities, we need a mechanism to suppress the interpolation mechanism when the difference between adjacent velocities becomes too great. One simple way to generate this behavior is with current limiting resistors between cells. Now, for small differences in velocity values (voltage) between neighbors, the connections are linear. As the voltages increase, the available current approaches a limit and the effective resistance begins to increase. The increased resistance causes the adjacent nodes to be less tightly coupled, as they should be when straddling a discontinuity, yet they are never detached entirely. This property is important so when the discontinuity moves or disappears, the circuit can recover to its smooth interpolation linear regime.

The resistor-equivalent circuit of Figure 7.6 has the current limiting property. The I-V transfer function is a hyperbolic tangent function. Plotted in Figure 7.7 is the I-V curve along with the resistance function. Near the origin, the circuit is linear and the resistance is constant. As the voltage increases, the current approaches its

limit at the value set by the current source of the amplifier. This limit is reflected as a rising resistance. If the determination of discontinuities is dependent only on adjacent differences, the resistor circuit with current limiting is an excellent implementation candidate.

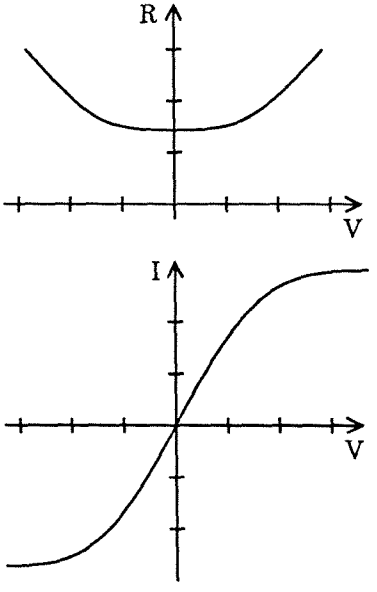


Figure 7.7: Transfer curve and effective resistance curve for the resistor circuit.

Koch *et al.* incorporate *a priori* knowledge about the nature of edges to penalize rare or non-physical combinations of nearby edges. They propose using digital hardware to compute the arbitrary nonlinear energy expressions associated with these penalties. Analog circuits can also perform non-linear computations. The desired penalty calculation in this case amounts to a coupling between nearby "resistors." This coupling could be implemented with analog or digital circuits or a mix of the two integrated together on a single CMOS chip. Figure 7.8 shows one possible configuration. What is shown in the diagram as resistors are computational elements that produce a current proportional to voltage for a range of voltages. For larger voltages, the elements perform a non-linear calculation that incorporates the *a priori* knowledge about edges. The current produced by the variable resistor in this regime of operation is a function of the neighboring discontinuities as well as the local voltage difference.

Extensions to the motion detector architecture seem to be viable implementations of a variety of the early vision algorithms under investigation in the AI field

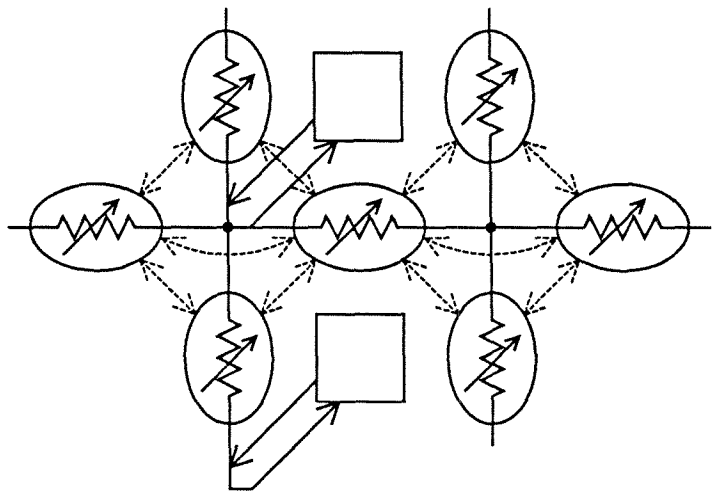


Figure 7.8: Incorporating *a priori* knowledge. Each variable resistor is actually an analog non-linear computation element that varies its "resistance" according to neighboring resistances and voltages. This coupling between neighbors is designed to encourage plausible arrangements of discontinuities and discourage non-physical ones.

of computational vision.

7.4 Biological Vision Modeling

It is a general property of collective systems that a collection of many local units working together, each doing a simple function, gives rise to a complex global behavior. This property is extremely useful for the system designer. My motion detector illustrates this property by globally resolving ambiguities when possible (a complex behavior) through a system of interconnected simple local cells each of which is performing a simple local computation. The complex-from-simple property can make biological vision research difficult. Researchers must derive clues about collective vision systems from observing their very complex behavior. Movshon *et al.* [15] performed psychophysical experiments with moving gratings and developed a visual motion model. This section investigates the suitability of my optical motion detector as a model for biological vision systems by comparing its response to the same input stimuli used by Movshon to the response of the biological systems.

The inherent motion ambiguity of one-dimensional images such as stripes is well recognized by researchers in the biological vision field. The consensus is that

the structures in the low levels of vision each deal with only one component of information, namely motion perpendicular to the edge and that the ambiguities that are unresolved because of this low level treatment are resolved at higher levels of processing. My motion sensor, in contrast, resolves much more of this ambiguity at the lowest levels of processing. To resolve ambiguities at a higher level does not require a different kind of algorithm or structure, just the same one applied to a larger field of view.

The motion detector explicitly represents the final disambiguated velocity at the lowest level. This representation allows the local computational elements to adjust their calculation based on the results of neighboring calculations. Alternative vision models have been proposed where each local cell independently makes a calculation and passes it to a higher level for processing. The highly interconnected cooperative system that results from the explicit representation of the aggregate answer, seems to have much more of a biological flavor than a strictly feed-forward system.

One-dimensional patterns such as bars or gratings are widely used by biologists for stimulating vision systems. Since such patterns appear so infrequently in nature on the scale of the entire visual field it is unlikely that biological vision systems evolved to explicitly handle these cases. My chip also was not specifically built to handle these ambiguous cases, but has a secondary effect described in Section 6.1 that causes the reported velocity to tend toward zero in the absence of information in the image. This effect will also influence the sensor output for one-dimensional patterns.

Figure 7.9 shows the rubber band model for the case of a stripe image input to the electronic motion detector. A true velocity detector would be content to report equally well any velocity that lay upon the single constraint line or multiple coincident constraint lines. The motion detector with its additional tendency toward zero velocity brings the reported velocity as close to zero as it can while still on the constraint line. This is the point where the reported velocity is perpendicular to the edges of the image. In the absence of disambiguating information, the motion sensor will report the component of velocity normal to the edge. This is not because the normal component is the fundamental low level representation of velocity but because the chip is being presented with an unusual situation that it is handling in a reasonable way.

Applying a one-dimensional grating to an electronic vision system is operating it out of the range for which it was designed. One could wonder if applying 1-D patterns to a biological vision system might be operating it out of the range for which it evolved. If so, the conclusions drawn from such experiments should be tempered with this knowledge.

Movshon et al further display two one-dimensional gratings superimposed and

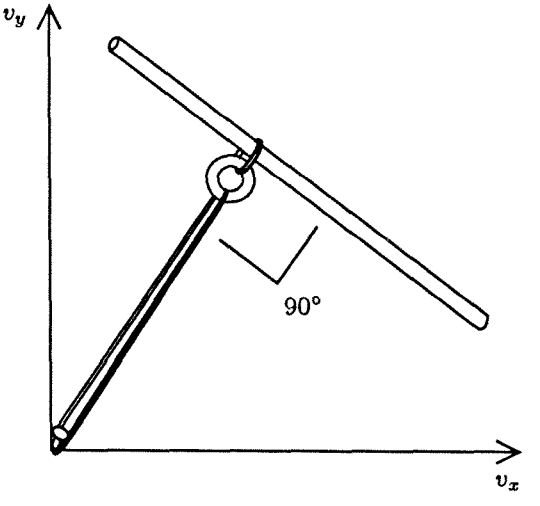


Figure 7.9: Physical model for the constraint solver doing the best it can with ambiguous input. The constraint solver tends to report zero velocity when no information is present. This tendency, modeled as a rubber band to the origin, causes ambiguous velocities to be reported as perpendicular to the image edge.

investigate when the observers perceive the two separately sliding over each other and when they perceive a single coherently moving plaid. The electronic motion detector has a single global velocity that it reports. As a result it will always "perceive" the coherent motion. The extension to the motion detector to allow smooth variation in motion described in the previous section could simultaneously compute different velocities at different points in the velocity network. For sufficient contrast in both dimensions for every neighborhood in the net, all velocities will be computed as the coherent plaid motion.

As the contrast of one of the gratings is reduced, the secondary effect of the motion sensor to tend toward zero velocity will become greater than the effect of the weak grating, yielding a reported velocity that is perpendicular to the higher contrast grating and decoupled from the weaker one. Movshon reports a region for biological systems between the threshold of detection of the weaker grating and the threshold of coherent motion of the plaid. This region suggests the importance the visual system puts on the higher contrast signals, yielding a velocity that is consistent with them even in the presence of detectable lower contrast signals. The zero tendency of my motion system is a secondary effect with a fixed small magnitude. On the other hand, for biological systems or more advanced electronic ones, this effect may scale with the largest contrast signal so it could be contrast level independent. Contrast gain control in biological vision systems is already an accepted idea. It has been observed in the cat retina by Shapley [20].

The motion sensor model can easily explain the transition from a coherent percept to a separate motion perception of the higher contrast grating as the relative contrasts of the two gratings changes. To explain the perception of motion of the weaker grating after the coherence has ceased requires a look at the spatial frequency of the gratings.

The second experimental result reported by Movshon is the dependence of relative spatial frequency of the two gratings on the threshold of coherence. Matched spatial frequencies provide the best coherence with the threshold rising as the frequencies differ. Let us look at the effect of the spatial frequency of a plaid on the behavior of the electronic motion detector with the network extension described in the previous section.

A simplified velocity network is shown in Figure 7.10 that includes the load resistors R_L at each node. The influence of the velocity information on a node in the network on a neighboring node diminishes with distance away, due not only to inputs to the net from other more local constraint solvers but also from the tendency-to-zero at each intermediate node produced by R_L. The decay of influence is an exponential function of distance. We will call the distance where the voltage representing velocity drops to one-half its center value, the radius of influence. This

radius distance is a constant that is dependent upon the ratio of the resistors between nodes, R, to the resistors within nodes, R_L.

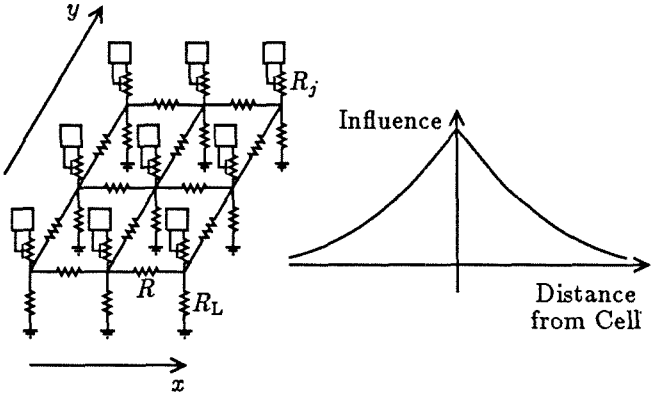


Figure 7.10: A simplified velocity network and a plot of the influence of the velocity voltage of a node as a function of distance from the node. The constant in the exponent is a function of the resistor ratio $\frac{R}{R_L}$.

The radius of influence due to R_L can be made quite large by choosing a large R_L. This radius sets the maximum distance that a node's influence can be felt. For reasonable images, the influence distance will be reduced significantly, due to the inputs from the constraint solvers at each node. The output resistances of the local circuits, R_j, are generally smaller than R_L but are highly image dependent and vary from place to place in the array. In areas where the image is of uniform intensity the output conductance of the cell is zero, so the velocity information from higher contrast areas will propagate a long way, limited only by the R_L radius of influence. For regions with high contrast edges, the influence of neighbors is greatly reduced. Here the output resistance, R_j, is relatively low so the locally computed values have a much greater effect on the node voltage.

An additional complexity not captured in Figure 7.10 is that the values on the nodes are vector quantities and the output resistances of the local constraint solvers reflect the interrelationship between the voltages representing the x and y components of velocity. For a high contrast nearly one-dimensional image, node voltages will have little influence on their neighbors in a direction perpendicular to the stripes or edges. Parallel to the stripes, however, the influence of a node voltage can extend much further to the limit imposed by R_L.

Putting the above ideas back into the frame of the psychophysical experiments, we see that each location in the motion sensing array sees only a single ambiguous

intensity gradient. In the presence of influence from nearby locations that are sensing a different intensity gradient, the result will be the solution of constraint lines. As the spatial frequency decreases so that the distance between contributions from non-parallel gradients exceeds the radius of influence, the result will be the reporting of separate velocities from each region. In this case, the tendency to zero is greater than the influence from cooperating regions far away, so each local cell will report the component of velocity normal to its local edge. This mechanism can account for the separation of the coherent motion into two separately perceived motions as spatial frequency decreases. As spatial frequency increases past the sensor spacing frequency, the integrating nature of the sensors will average out the intensities, effectively filtering out any information contained in the higher frequencies. This property is very important to prevent aliasing but will reduce the contribution to velocity sensing of these spatial frequencies and thus the coherence would drop off for higher frequencies.

The arguments given above for reduced motion coherence for high and low spatial frequencies are relative to sensor spacing. Movshon's experiments show this behavior for frequencies of the two grating relative to each other for more than one frequency relative to the sensor spacing. This observation suggests that biological visual motion systems may have a hierarchy of network motion sensors each operating at a different spatial scale. This change in scale could be implemented electronically using different resistor ratios for an array with identical structure or by making structures with similar resistor ratios but with a sparser cell grid and longer interconnections.

The motion detection algorithm described in this thesis is probably not a direct match to any biological systems. Although it may be possible to identify which populations of neurons correspond to each of the electronic circuits of this model and to determine how current and voltage map into nervous system representations, it will most likely take modifications and extensions beyond those suggested above to form a consistent, believable biological vision model. My intent was to draw the attention of vision modelers to an example of a simple, elegant, constraint-solving architecture. I have given an existence proof that such a system can be well founded mathematically, yet exhibit some of the same collective behavior as is often found in biological systems. I hope that this system or similar ones can be incorporated into future biological vision models.

7.5 Analog VLSI Systems

The working motion detector demonstrates that large collections of analog elements can be combined on a single chip. The resulting benefits are high density and low power consumption. While the moderate speed of analog circuits operating in subthreshold is not as great as digital devices, a digital multi-bit multiplication requires many propagation delays or clock cycles.

Threshold variations between adjacent transistors are significant and limit the accuracy of local analog computations. Feedback methods can often be used to limit the errors due to device variations. The collective nature of the motion detector architecture increases accuracy and reliability with larger arrays. The system produces answers that are more reliable than those of its constituent parts. Conventional digital systems, by contrast, are considerably less reliable than their component parts.

Usually analog circuits are thought to be less flexible after manufacture than digital circuits. Often the opposite is true. The control voltage on the current source provides a convenient means to "program" an analog circuit, varying its operating characteristics over orders of magnitude. Judicious use of these control points by the system designer allows a great deal of flexibility to accommodate a variety of real world unknowns. It is easy to go overboard with analog knobs and end up with a system so complex that it becomes difficult to find a state for all the programming where the system will operate as intended. Analog control points also provide convenient places for feedback in the form of automatic gain control. As the design of a system progresses, knobs that were set manually for purposes of exploration and experimentation, become set by the system itself, often as slowly varying functions of input data. In this way systems can be made continuously self-regulating.

Some digital logic systems will never be replaced with analog ones. Digital designs are best for problems that have complete well-defined inputs and demand precise outputs and reliable long term storage. Analog systems, on the other hand are useful for solving problems with partial information as input and require only approximate results with moderate accuracy. As we gain experience with large analog systems and more fully recognize their benefits, we may find that our attention is turning more toward the imprecise, ambiguous kind of problems that were difficult to solve using only digital techniques. Hybrid systems combining analog and digital circuits show great potential for future implementations.

Chapter 8

Conclusions

I have designed and built the first of two new generations of integrated motion sensors. The first uses a correlation technique for motion detection. A one-dimensional version has been demonstrated.

A second algorithm and architecture emerged from the desire for more collective behavior through the aggregation of locally derived quantities. This design utilizes closely coupled analog photosensors and analog computational elements. It gains performance benefits through the parallel operation of large arrays of sensors and computing elements made feasible by the use of small analog circuits. The system makes extensive use of local intensity information and is thereby resistant to the global gradient problems of other approaches. This locality also eliminates the need for any prior higher level processing like edge detection or object recognition. The circuits used in this design have been demonstrated and characterized in the laboratory.

This integrated motion system can be viewed as an example of a Hopfield neural network. As such it demonstrates the feasibility of an important extension to the Hopfield theory—input-driven synapses.

The motion chip is also a first example of an analog network used for early vision processing. These networks are just now becoming popular in the AI/computational vision field. The motion detector design can easily be extended to handle more complex motion fields.

The motion detector architecture, along with local analog computational elements, provides a dense, reliable means of processing high bandwidth visual data. The motion detector chip demonstrates the suitability of analog VLSI circuits for processing of sensory data.

The motion detector architecture is one of the first of a growing class of systems that employ collective computation. The property of increased reliability and ac-

curacy with larger arrays, which the motion detector exhibits, is one of the clear benefits of collective computation. As we gain experience with these types of systems, the range of application may expand beyond that of processing sensory data. The motion detector proved to be a good first example. While having a crisp, solid mathematical foundation, the motion sensor exhibits the collective behavior that is so necessary in the fuzzy real world.

References

[1] Gary Bishop and Henry Fuchs. The *Self-Tracker:* a smart optical sensor on silicon. In *1984 MIT Conference on Very Large Scale Integration*, pages 65–73, 1984.

[2] D. Cohen and G. Lewicki. MOSIS – the ARPA silicon broker. In *Proceedings from the Second Caltech Conference on VLSI*, pages 29–44, California Institute of Technology, Pasadena, CA, 1981.

[3] A. K. Dewdney. Computer recreations. *Scientific American*, 18–29, June 1985.

[4] Barry Gilbert. A precise four-quadrant multiplier with subnanosecond response. *IEEE Journal of Solid State Circuits*, SC-3(4):365–373, 1968.

[5] Ellen Catherine Hildreth. *The Meausurement of Visual Motion.* The MIT Press, Cambridge, 1984.

[6] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences USA*, 79:2554–2558, April 1982.

[7] J. J. Hopfield. Neurons with graded response have collective properties like those of two-state neurons. *Proceeding of the National Academy of Sciences USA*, 81:3088–3092, May 1984.

[8] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[9] Christof Koch, Jose Marroquin, and Alan Yuille. *Analog "Neuronal" Networks in Early Vision.* Artificial Intelligence Laboratory Memo 751, MIT, Cambridge, Jun 1985.

[10] Richard F. Lyon. The optical mouse, and an architectural methodology for smart digital sensors. *CMU Conference on VLSI Systems and Computations*, 1–19, 1981.

[11] C. A. Mead and L. A. Conway. *Introduction to VLSI Systems.* Addison-Wesley, 1980.

[12] Carver Mead. A sensitive electronic photoreceptor. *1985 Chapel Hill Conference on VLSI*, 463–471, 1985.

[13] Carver Mead and Mary Ann Maher. A charge-controlled model for submicron MOS. In *Proceedings of the Colorado Microelectronics Conference*, May 1986.

[14] Carver Mead and Michelle Mahowald. *An Electronic Model of the Y-System of Mammalian Retina.* Technical Report 5144:DF:84, California Institute of Technology, June 1984.

[15] J. Anthony Movshon, Edward H. Adelson, Martin S. Gizzi, and William T. Newsome. The analysis of moving visual patterns. In C. Chagas, R. Gattass, and C. G. Gross, editors, *Study Group on Pattern Recognition Mechanisms*, Vatican Press, Rome, 1984.

[16] Russel Newcomb. Considerations in MOS ratioed capacitor layout. *VLSI Design*, 1981.

[17] Tomaso Poggio, Vincent Torre, and Christof Koch. Computational vision and regularization theory. *Nature*, 317:314–319, Sep 1985.

[18] Werner Reichardt, Tomaso Poggio, and Klaus Hausen. Figure-ground discrimination by relative movement in the visual system of the fly. *Biological Cybernetics*, 46:1–30, 1983.

[19] C. L. Seitz. Ideas about arbiters. *Lambda - The Magazine of VLSI Design*, 10–14, First Quarter 1980.

[20] Robert Shapley and Christina Enroth-Cugell. *Progress in Retinal Research*, chapter Visual Adaptation and Retinal Gain Controls. Volume 3, Pergamon Press, Oxford, England, 1984.

[21] Massimo A. Sivilotti. *A* CMOS-*Bulk Imaging Array.* Technical Report 5226:TR:86, California Institute of Technology, 1986.

[22] John E. Tanner and Carver Mead. A correlating optical motion detector. *1984 MIT Conference on Very Large Scale Integration*, 57–64, 1984.