AD-A119 859

CAI BARRINGTON IL

FLIR STABILIZATION STUDY.(U)

APH 82 D DEFOE, T GOODRICH, J MARTIN

UNCLASSIFIED CAI-7901-A002

F/G 17/5

DAAK70-81-C-0255

NL

Technical Report ____________

FLIR STABILIZATION STUDY

CAI, a Division of Recon/Optical, Inc.
Barrington, Illinois 60010

April 1982

Technical Report for Period October 1981 - March 1982

Approved for Public Release, Distribution Unlimited

NIGHT VISION AND ELECTRO-OPTICS LABORATORIES
Fort Belvoir, Virginia 22060

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A119 859	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) FLIR Stabilization Study		5. TYPE OF REPORT & PERIOD COVERED Final 15 Oct 81 - 15 March 82
		6. PERFORMING ORG. REPORT NUMBER CAI 7901-A002
7. AUTHOR(s) D. DeFoe, T. Goodrich, J. Martin, C. Schleifer		8. CONTRACT OR GRANT NUMBER(s) DAAK70-81-C-0255
9. PERFORMING ORGANIZATION NAME AND ADDRESS CAI, a Division of Recon/Optical, Inc. 550 W. Northwest Highway Barrington, Il. 60010		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Night Vision and Electro-Optical Laboratories Fort Belvoir, Virginia 22060		12. REPORT DATE April 1982
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Stabilization, MTI, Correlation, Passive Ranging, Dynamic Sampling, Pipeline Architecture

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Precision pipeline correlation has been shown to be a superior method for stabilizing FLIR imagery. A pipeline correlation stabilization module provides better performance than gyro transducers, is inherently tuned to image parameters, can be remotely located from the sensor and can be used for passive or active stabilization of the image. In addition, the pipeline correlation module will provide edge enhanced imagery and a measure of focus of the sensor. With the addition of one board, the module will provide a motion map or, given two separated boresighted sensors, a range map of of the scene. A special function module can be connected to provide stabilized, scaled,

20 (cont.)

integrated and dynamically-sampled imagery as desired.

SECTION I
INTRODUCTION AND SUMMARY

SCOPE OF STUDY

This is the final technical report for the FLIR Stabilization Study Program (Contract No. DAAK70-81-C-0255). This document describes the result^r of a 5-month technical effort to study, analyze and conduct preliminary design of a precision pipeline correlation for FLIR stabilization applications. The period covered by this report is 15 October 1981 to 15 March 1982. Mr. Peter Durenec is the contract monitor.

The overall objective of this program is to define the appropriate architecture and design of a pipeline correlator for FLIR stabilization, analyze expected performance and investigate other potential applications. The pipeline correlator is an implementation of a precision correlation algorithm, proprietary to CAI, which measures image shifts to accuracies beyond the resolution of the sensor/detector.

SUMMARY OF RESULTS

Correlation stabilization is essentially different from inertial stabilization, in that with the former, the imaged scene itself is tracked, as opposed to sensor inertial rotations, as with the latter. This is shown to provide significantly improved performance at lower cost.

The salient results of this program are as follows:

- A modularized approach has been defined which allows various configurations dependent upon application, and is capable of expansion from 60 to 120 to 180 elements per line.
- A basic stabilization module has been defined and designed (preliminary) which:

  a) Provides scan and cross scan direction interframe offsets on a pixel-by-pixel and line-by-line basis, respectively

b) Can be used with fixed or servo-controlled sensors

c) Can replace up to three gyro transducers and/or a scan position sensor

d) Provides significantly better performance than gyro transducers
   - higher bandwidth
   - better accuracy
   - no limit to positional excursion
   - wider range of angular rates
   - no drift
   - lower cost

e) Provides edge video output

f) Provides a measure of focus

- The basic stabilization module contains two cards for pipeline correlation, one frame store controller card, and one and one-half cards per 60 elements of detector array length.

- The addition of one moving target indication (MTI)/range card and one control receiver card to the stabilization module and a remote control panel provides added capabilities. This configuration will provide video format outputs for display or use by another processor of the following:

  a) MTI

  b) Either relative motion mapping or (given detector size, focal length and range inputs from the control panel) absolute velocity mapping

  Given two boresighted sensors, but not requiring the frame storage cards, this same configuration will output passive:

  c) "Within range" indication

  d) Either relative range mapping or (given detector size, focal length and baseline separation inputs from the control panel) absolute range mapping

- A special function frame store module has been defined and designed (preliminary) which:

  a) Provides video putput in any combination of:

- Passively stabilized for translation, dilation, rotation and skew (update occurs only for those portions of the stabilized scene currently within the FOV of the sensor; other portions are frozen with last valid video)
- Integrated in time to improve signal-to-noise ratio
- Scaled 1:1, 2:1 or 4:1
- Dynamically sampled to improve resolution up to 3.5x

b) Provides a window superimposed on sensor video to define special video location and coverage

c) Is used with a plug-in addition to the remote control panel and the control receiver card option in the stabilization module

d) Consists of one controller card plus one card per 60 elements of detector array length.

OPTIONAL CONFIGURATIONS

Figures 1.1 through 1.5 illustrate the several configurations possible with these modules and optional boards.

GUIDE TO THE REPORT

A detailed description of the stabilization module and a comparison of its performance to that of gyro transducers are given in section II. This section also presents some correlation concepts pertinent to this report for the reader unfamiliar with correlation. Also included in section II is a description of other outputs provided with the basic module and an explanation of the need for a dedicated frame store.

One optional configuration which provides motion mapping of passive ranging is given in section III, together with a short explanation of passive ranging.

Section IV is a brief discussion on the relative merits of active and passive stabilization.

BASIC STABILIZATION

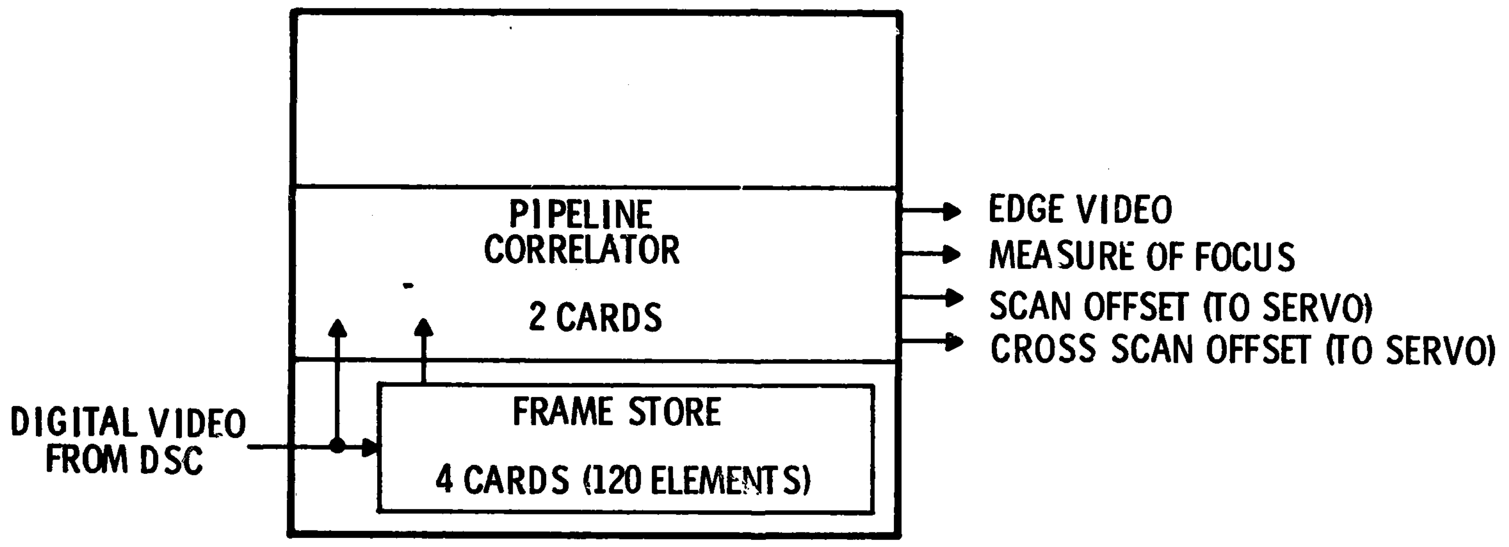


FIGURE 1.1

STABILIZATION PLUS MTI

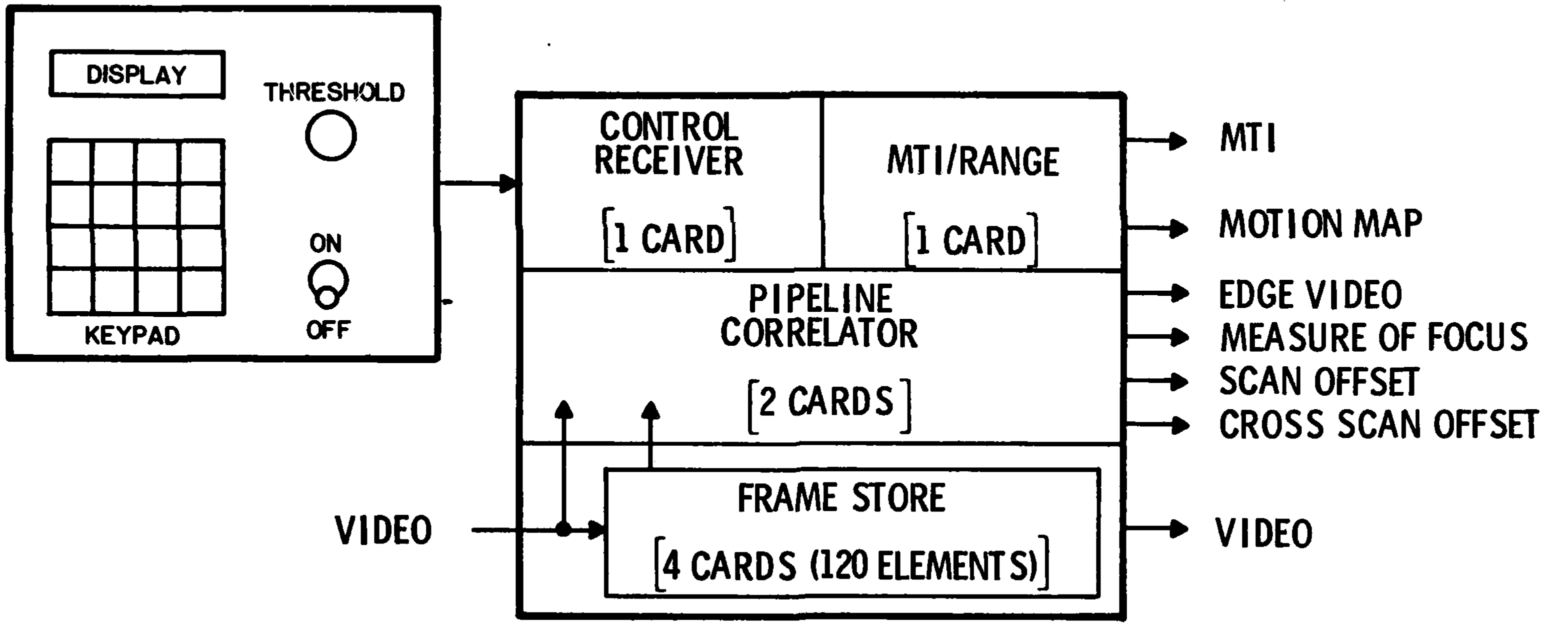
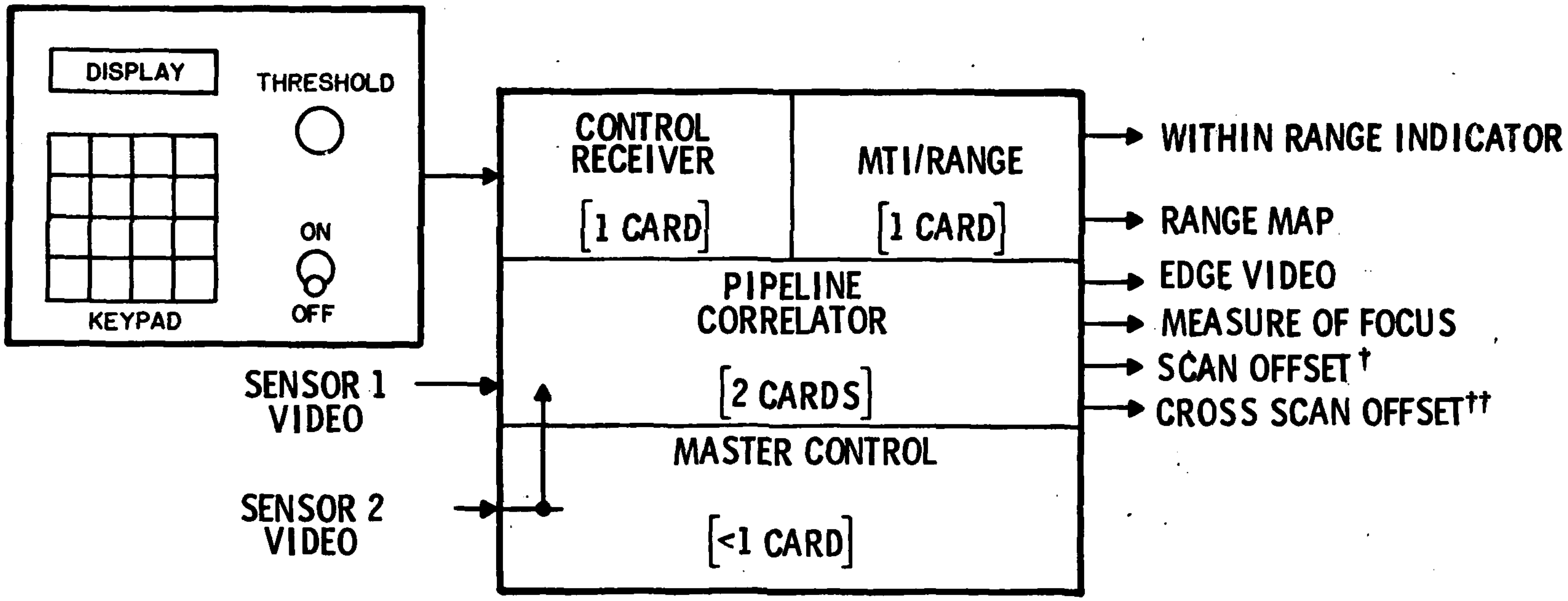


FIGURE 1.2

PASSIVE RANGING

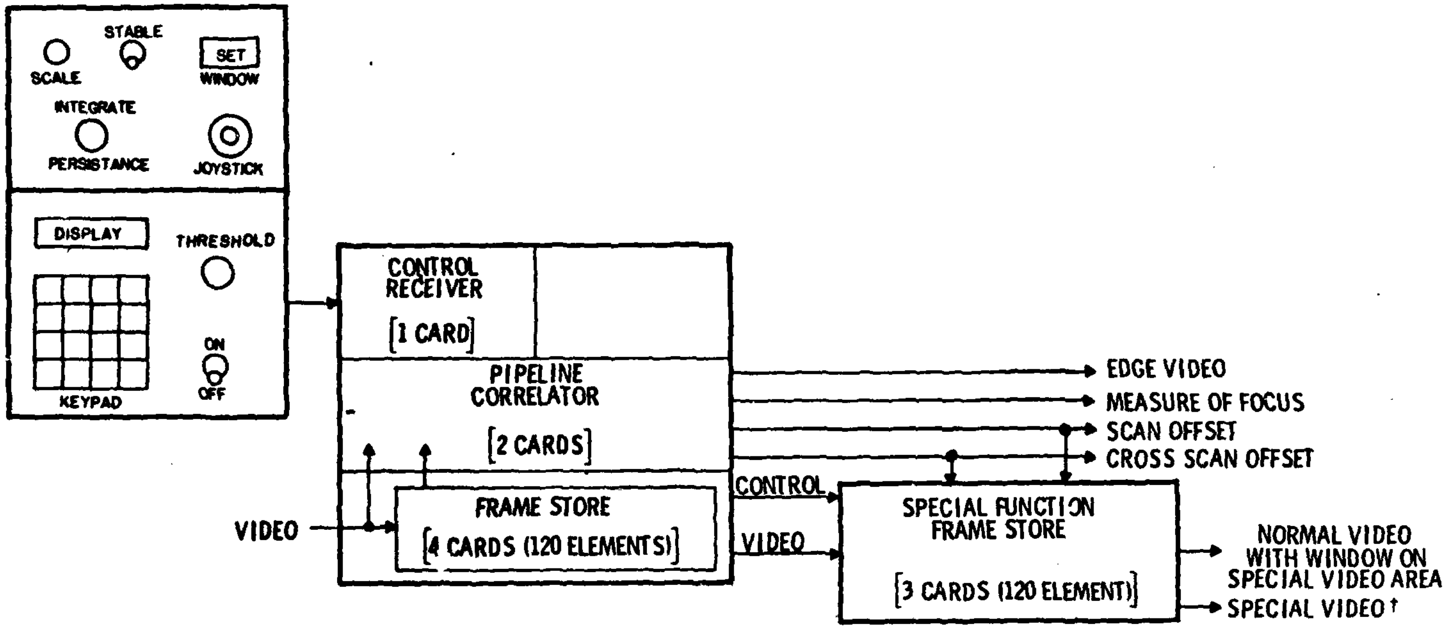


† TO SCAN MIRROR SYNCHRONIZER

†† LINE BY LINE RANGE

FIGURE 1.3

STABILIZATION AND SPECIAL FUNCTION



†STABILIZED, SCALED, INTEGRATED, AND/OR DYNAMIC SAMPLED

FIGURE 1.4

STABILIZATION WITH SPECIAL FUNCTION AND MTI

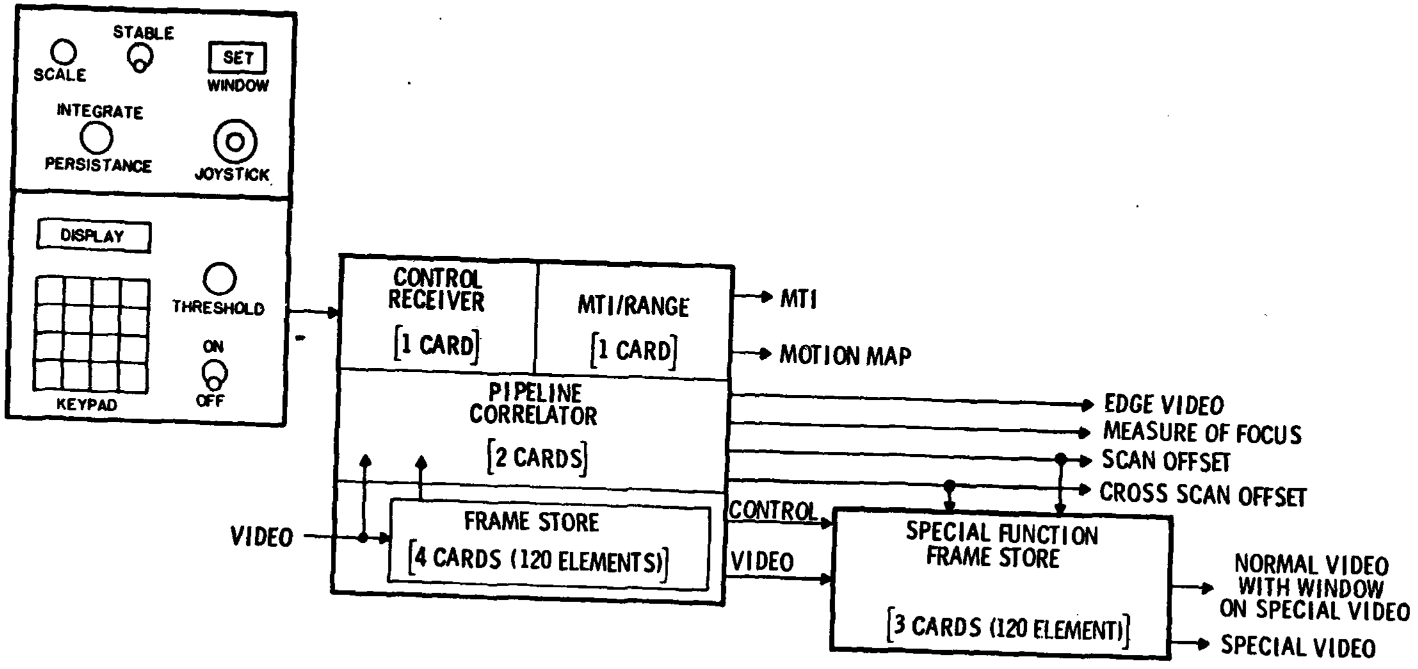


FIGURE 1.5

A description of the features offered by the optional special function module is given in section V.

In section VI, the design consideration and the conclusions of this preliminary design effort are presented. A more detailed discussion of the recommended brassboard demonstration system, together with the specific card design, is in section VII, while production design considerations are discussed in section VIII.

Conclusions and recommendations are given in section IX.

SECTION II
STABILIZATION MODULE DESCRIPTION

BENEFITS

The essential feature of correlation stabilization is that the stabilization results are derived from the scene, rather than an inertial frame of reference. This is the source of its many advantages over gyro transducers. These advantages are summarized in table II-I.

For example, since gyro transducers measure only sensor rotations, they do not account for sensor translation. Therfore, on a moving platform, they are not effective in tracking a particular portion of the environment. Correlation stabilization conversely tracks the scene itself, and therefore would command the sensor to rotate and maintain that scene in the field of view as the platform translates relative to the scene. Note that if the scene is a moving target, correlation stabilization will track the target.

Because it operates on the image, correlation stabilization is inherently tuned to the sensor and image parameters. A change of focal length has no effect on performance relative to resolution, except for the necessity to scale any feedback to servo mechanisms. Accuracy is rated in terms of image resolution and, therefore, the necessary precision to stabilize a scene is maintained regardless of sensor/detector parameters.

Correlation provides more information for stabilization than do gyro transducers. It provides data on translation motions in two axes and rotation. Hence, it actually takes the place of three gyro transducers. Furthermore, it tracks skew and dilation or scaling of the image in two axes and can passively stabilize the image for these effects, which is impossible with inertial devices.

The B-2 digital scan converter for FLIR imagery includes a scan position sensor (SPS) module to measure the position of the scan mirror in common module FLIR's. This module is used to assure accurate location of lines for reformatting

PIPELINE CORRELATION FOR FLIR STABILIZATION ADVANTAGES

- TRACKS SCENE; EFFECTIVE ON TRANSLATING PLATFORMS AND MOVING SCENES
- INHERENTLY TUNED TO IMAGE PARAMETERS
- ACCOUNTS FOR SCENE TRANSLATION, DILATION, ROTATION AND SKEW
- REPLACES GYRO TRANSDUCERS
- CAN REPLACE SCAN POSITION SENSOR FOR B2 DIGITAL SCAN CONVERTER
- HIGHER PERFORMANCE THAN GYRO TRANSDUCER
- REMOTELY LOCATED FROM SENSOR
- USED FOR ACTIVE AND/OR PASSIVE STABILIZATION
- LOWER COST THAN GYRO TRANSDUCERS
- MORE RUGGED THAN GYROS; NO MECHANICAL COMPONENTS
- PROVIDES OTHER FUNCTIONS

CAI

TABLE II-I

to video standards. Correlation stabilization can replace this module. However, it should be understood that correlation precisely locates lines in relative terms, on a frame-to-f'ame basis. It will not correct any geometric distortion which may have been present in the first frame. Therefore, when such distortion caused by the scan mirror is not considered important, the correlation module will obviate the need for the SPS. It will "correct" subsequent frames to the original distortion, if any.

In every performance figure, correlation stabilization is significantly better than gyro transducers. It is more accurate and has higher bandwidth and greater range. It can also be located remotely from the sensor in a protected environment, therby allowing smaller, more rugged sensor designs.

Furthermore, the results can be used actively by feeding back to servo mechanisms on the sensor, or passively by correction of pixel locations in frame store or on display. Finally, as will be detailed later, correlation stabilization provides many other important functions.

CORRELATION CONCEPTS

These paragraphs are intended to give the reader unfamiliar with correlation some basic understanding of the process, so that the pipeline correlator may be better evaluated. A more detailed treatment is given in appendixes A and B.

Correlation is a measure of match between two signals or functions. For the application considered here, the signals are video waveforms representing images. Consider two images on transparencies. Holding the transparencies, one on top of the other up to some light, the best match occurs when, overall, the most light is transmitted by the pair of images. For example, if the images are of the same scene, but there is a relative shift or translation between the two, then by sliding one transparency over the other and measuring the total light transmitted, the relative shift can be measured by noting where the peak in transmitted light occurs.

Correlation is merely the mathematical implementation of this process. It is the multiplication of the transmittance (or intensity) of a point on one image by the transmittance of the point on the second image which overlays the first point.

The multiplications are take point-by-point and summed over the whole image. As one image is shifted over the other, new multiplications and sums are taken and plotted as a funtion of shift. Locating the peak correlation value defines the shift between the images as shown in figure 2.1.

At the peak of the correlation plot or curve, all spatial frequencies match up and contribute to making the peak what it is. As the images are shifted away from this optimum shift, the highest spatial frequencies no longer match up, but lower frequencies do still match fairly well and contribute to an above-average correlation value. Thus, the correlation curve as a function of shift slopes more or less gradually down from the peak, depending on the frequency content of the image.

If the two images are strictly copies of one another, the correlation curve obviously peaks at zero shift. This is called autocorrelation. Similarly, if the two images are identical but shifted versions of one another, the correlation curve peaks at the relative shift position but is identical in shape as shown in figure 2.2.

Obviously, correlation cannot truly be taken point by point since there are an infinite number of points. The signals must be sampled to obtain a finite number of discrete points. However, as soon as the signals are sampled, the full correlation curve can no long be produced. Only a sampling of the curve results. For autocorrelation, the zero shift sample is still located at the true peak of the curve, but this true peak on the cross correlation curve is sampled only if the relative shift is an integer number of sample spacings (pixel dimensions). Since the object is to find the shift, it cannot be known a priori that the shift is indeed an integer number of pixels.

Simply locating the peak sample only defines the shift to an accuracy of ±1/2 pixel. However, since it is known that the autocorrelation curve samples and the

CORRELATION PROCEDURE

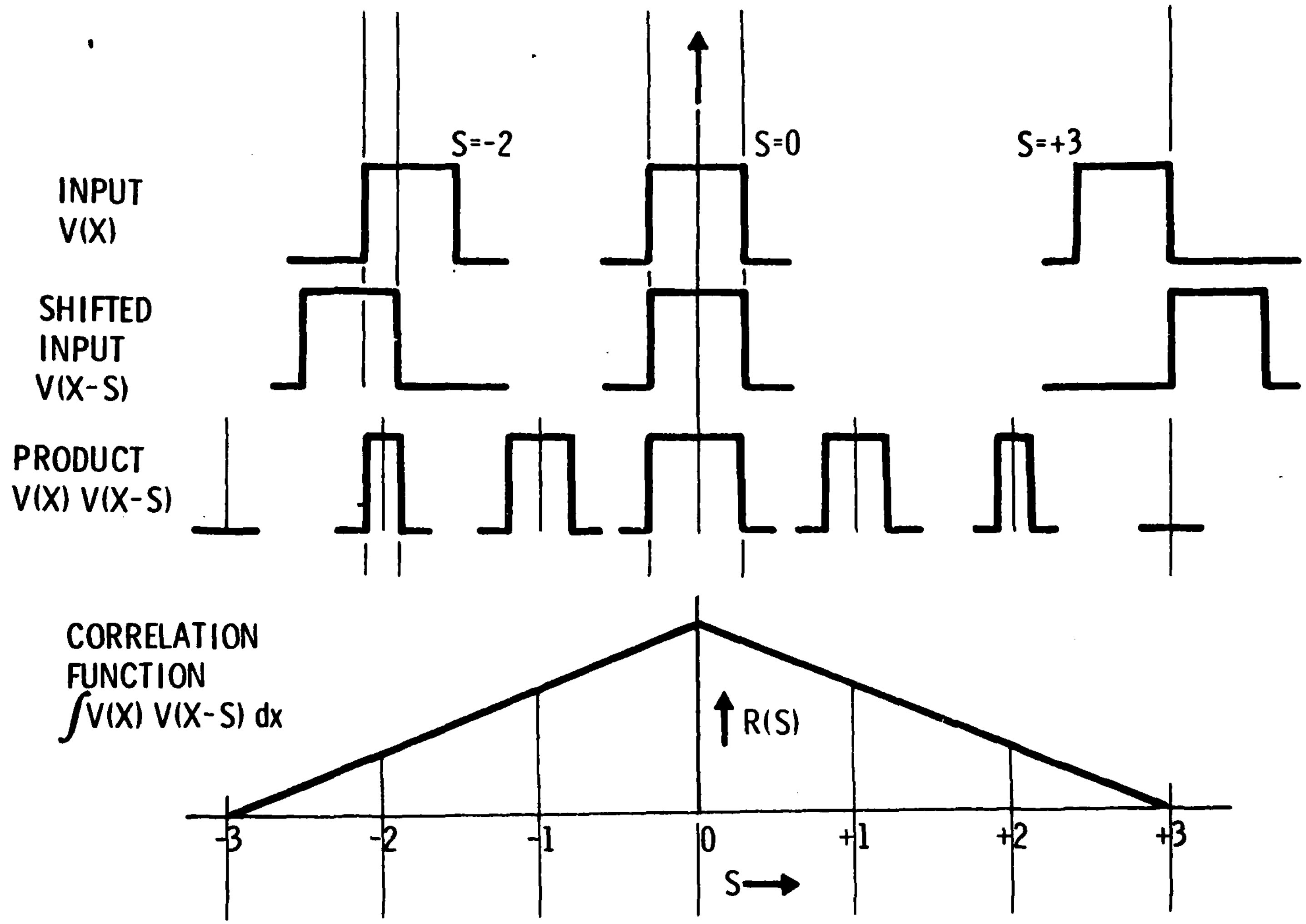
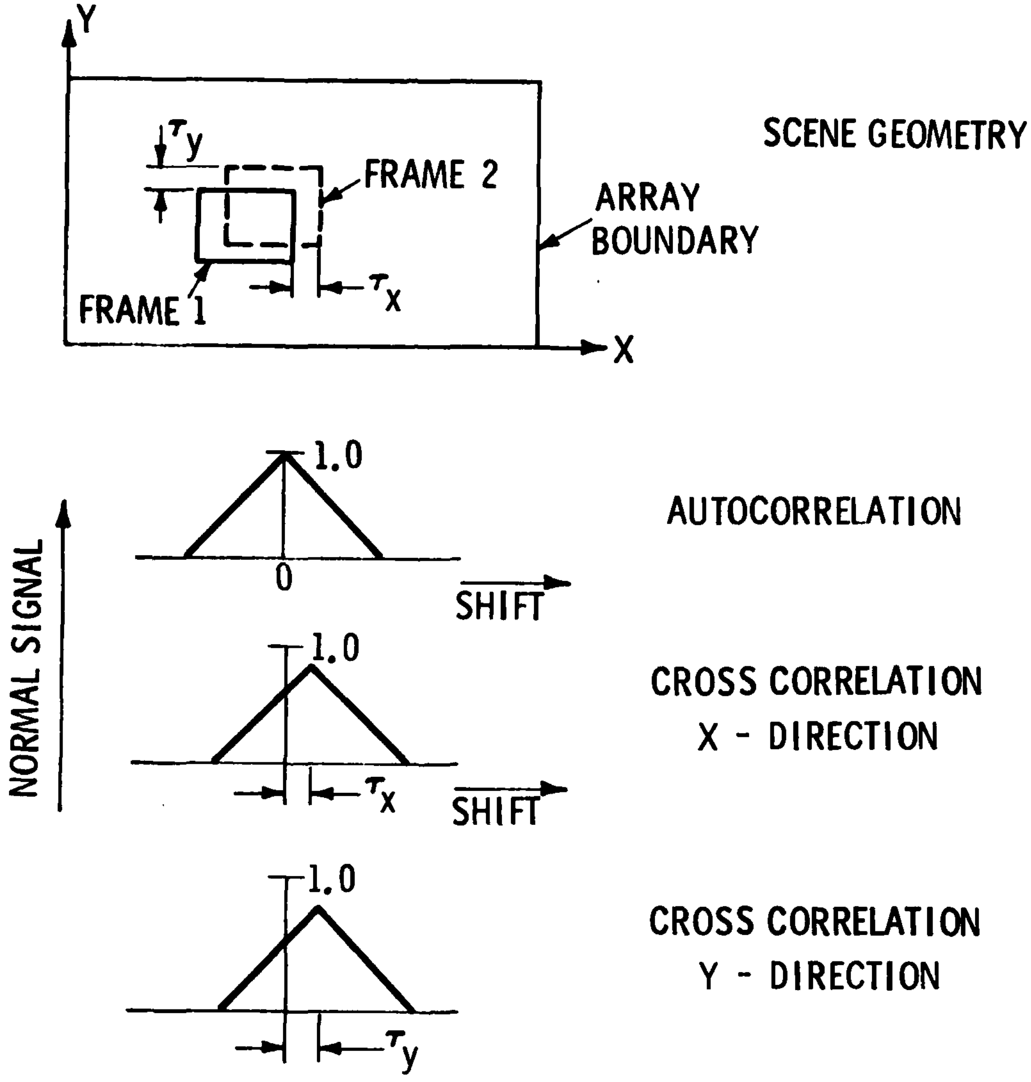


FIGURE 2.1

CORRELATION CONCEPT



SCENE GEOMETRY

AUTOCORRELATION

CROSS CORRELATION
X - DIRECTION

CROSS CORRELATION
Y - DIRECTION

FIGURE 2.2

cross corrrelation curve samples are samples of the same curve, it is possible to curve fit the samples to obtain a far more precise estimate of peak position on the cross correlation curve, that is, to a smaller fraction of a pixel dimension. Thus, it is not high spatial frequencies that are matched in precision correlation, but the lower spatial frequencies of the scene. A summary of the correlation process is given in table II-2.

In fact, precision pipeline correlation implements only this curve fitting operation. Thus, there is an implicit assumption that the peak is actually within ±2 pixels of shift from where it is expected. This is of little concern for the application of stabilization, since the shift can be tracked from line to line and on this basis, it is nearly impossible to shift more than 2 pixels per line (i.e., within 35μs).

One final point is that the correlation curve sharpness defines a measure of focus. If the scene is in focus, there are high spatial frequency components to match up and produce a high zero shift autocorrelation value. Conversely, if the zero shift value is not much larger than the neighboring shift values, there is no high-frequency information, or it has been destroyed by loss of focus. As a natural by-product of pipeline correlation, the autocorrelation width is calculated and can be made available to a focus drive mechanism in the sensor as shown in figure 2.3.

PIPELINE CORRELATION STABILIZATION

Precision pipeline correlation is a hardware implementation of correlation matching wherein each operation is sequentially performed along a bus. Each pixel datum enters the bus in sequence, is operated upon, and is passed to the next operation. Each operation may use constant operands and/or results from other operations. Upon exiting the bus, the datum has effectively been transformed into a measure of offset or misregistration relative to the datum for that pixel from the previous frame as shown in figure 2.4.

TABLE 11-2

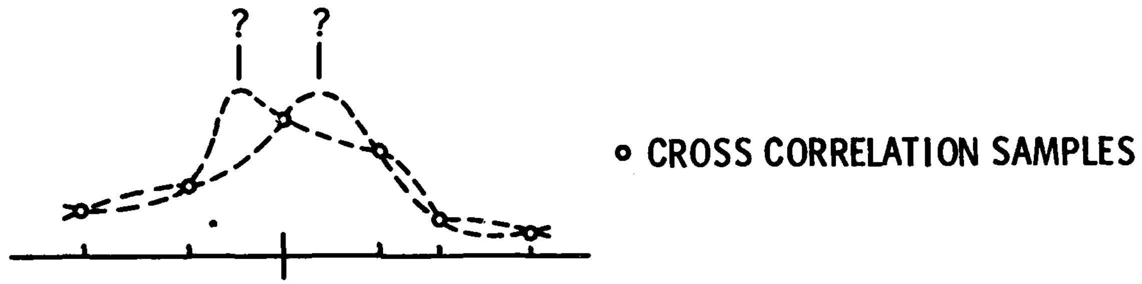
CORRELATION

- A CORRELATION CURVE IS SIMPLY A PLOT OF THE MEAN SQUARED DIFFERENCES OF INTENSITIES AS A FUNCTION OF X-Y SHIFT BETWEEN TWO IMAGES (CROSS CORRELATION)
- THE SHIFT AT WHICH THE MEAN SQUARED DIFFERENCE IS LEAST IS THE POSITION OF BEST MATCH
- (MORE PRECISELY, CORRELATION PRODUCT IS A CONSTANT MINUS THE MEAN SQUARED DIFFERENCE; THE BEST MATCH IS THEN WHERE THE CORRELATION PRODUCT PEAKS)
- TARGET MOTION PRODUCES SHIFT ON SEQUENTIAL IMAGES IN ONE OPTICAL SYSTEM
- FINITE TARGET RANGE PRODUCES SHIFT ON THE IMAGES IN TWO DISPLACED PARALLEL OPTICAL SYSTEMS; NOTE THAT THE IMAGE PLANES MUST MAINTAIN PRECISE ALIGNMENT TO ONE ANOTHER
- LOCATING THE PEAK CORRELATION OF THE TWO IMAGES MEASURES THE SHIFT

TABLE 11-2

CORRELATION (CONT)

- WITH SAMPLED IMAGES, A SAMPLED CORRELATION CURVE IS OBTAINED



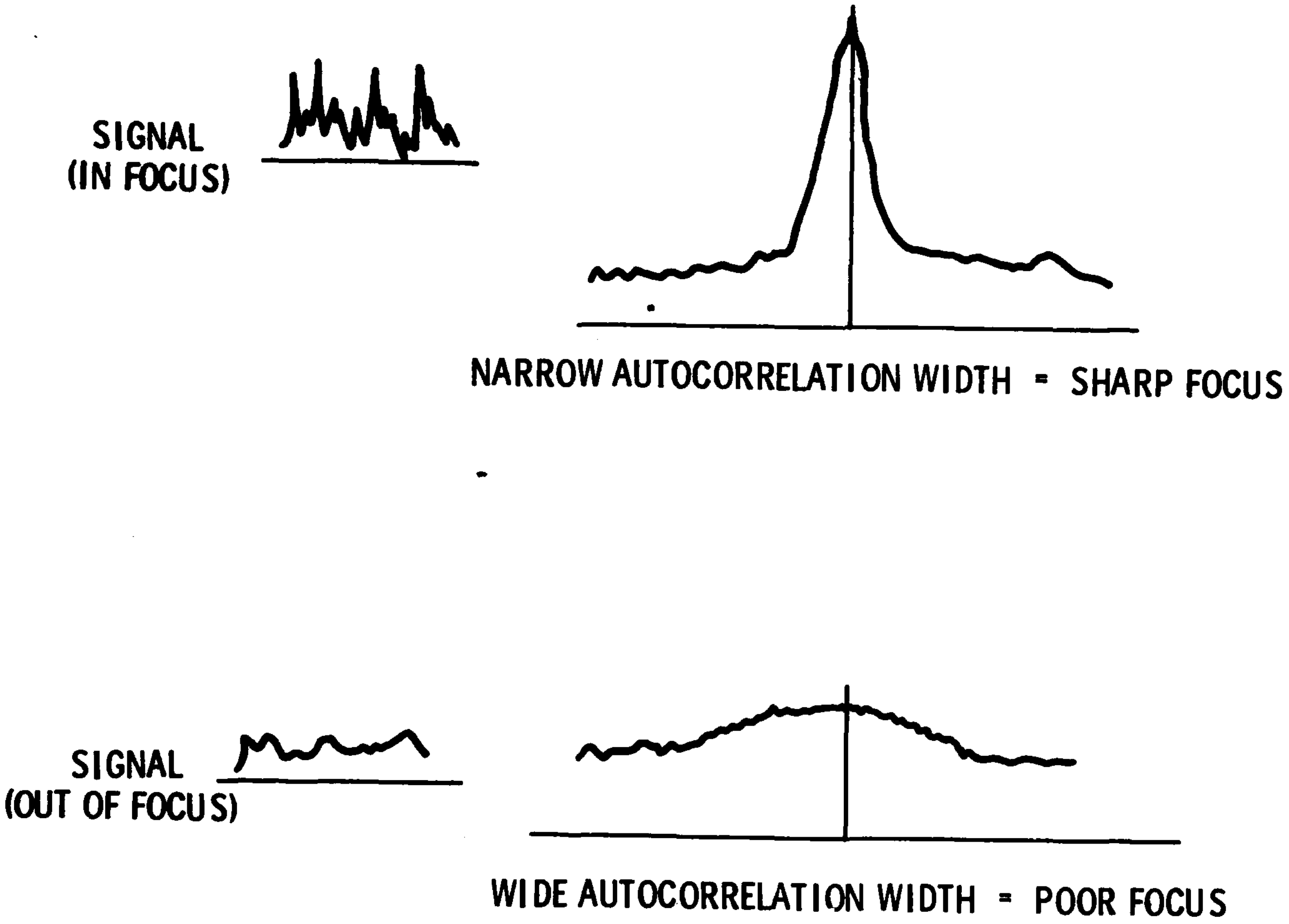
- THE PEAK SAMPLE IS EASILY LOCATED, BUT WHERE IS THE REAL PEAK OF THE FULL CURVE?

TABLE II-2

CORRELATION (CONT)

- BY FITTING A CURVE TO THE CORRELATION SAMPLES, THE CROSS CORRELATION SAMPLES CAN BE LOCATED ON THAT CURVE AND THE SHIFT DETERMINED
- BACKGROUND REJECTION IS OBTAINED BY CORRELATING ONLY A "WINDOW" AROUND THE TARGET
- MISALIGNMENT BETWEEN THE TWO RANGING OPTICAL SYSTEMS IS MINIMIZED BY DESIGN; STATIC RESIDUALS ARE COMPENSATED BY FIXED OFFSET TO MEASURED SHIFTS, AND DYNAMIC RESIDUALS ARE MEASURED BY AUTOCALIBRATION

MEASURE OF FOCUS



SIGNAL (IN FOCUS)

NARROW AUTOCORRELATION WIDTH = SHARP FOCUS

SIGNAL (OUT OF FOCUS)

WIDE AUTOCORRELATION WIDTH = POOR FOCUS

FIGURE 2.3

PIPELINE ARCHITECTURE

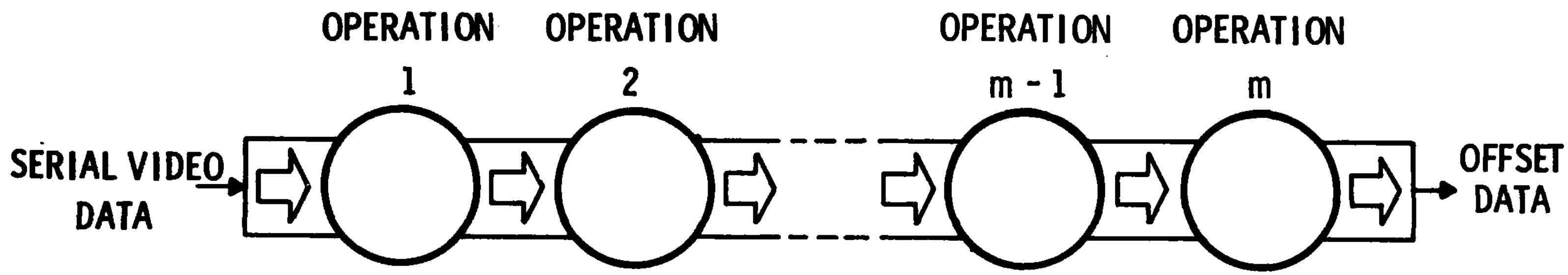


FIGURE 2.4

Correlation is defined over the bounds of some window less than or equal to the field of view. This window may be arbitrarily set, depending on the application. In pipeline correlation, the transformation of data is broken at a specific point in the pipeline to be summed over the desired window before proceeding to a final processing stage to produce an offset datum. If the summation is implemented as a running sum, always subtracting the partial results calculated n cycles ago, results can be produced at video rates. This is illustrated in figure 2.5. In fact, each result corresponds to a window centered about each pixel in the original video as shown in figure 2.6.

The geometry of common module FLIR imaging is particularly suited to pipeline correlation stabilization in several respects. Essentially, it consists of a linear array of up to 180 detectors which are scanned back and forth across the scene. The scan mirror is tilted slightly at either end of a one-directional scan to provide interlaced fields. Each one-directional scan defines a field and two interlaced fields define a frame.

Note that no particular control is used in the pipeline architecture to separate lines, fields and frames. The operation simply accepts a continuous stream of video data. This, of course, can produce confusion at the boundaries. The back and forth scanning of common module FLIR's obviates this problem at least in one direction. More importantly, this scanning method allows tracking of offsets from line to line across field boundaries. This is very desirable for pipeline correlation as referred to previously.

Because the sensing elements are in a linear array format, effects of motion are common to an entire line. This means that cross scan offsets may be calculated on a line-by-line basis and, in fact, there are sufficient detectors in the line to permit the window for cross scan correlation to be one line wide. The result is that, for the cross scan direction, there is no need for a running sum to be taken, and a simplified design may be implemented. However, for the scan direction, possible skews and rotations require a block window calculation as shown in figure 2.5. Thus, the architecture required for FLIR stabilization is as shown in figure 2.7. Note that edge images and measures of focus are natural by-products of pipeline correlation.

PIPELINE PRECISION CORRELATION FOR m X n WINDOW

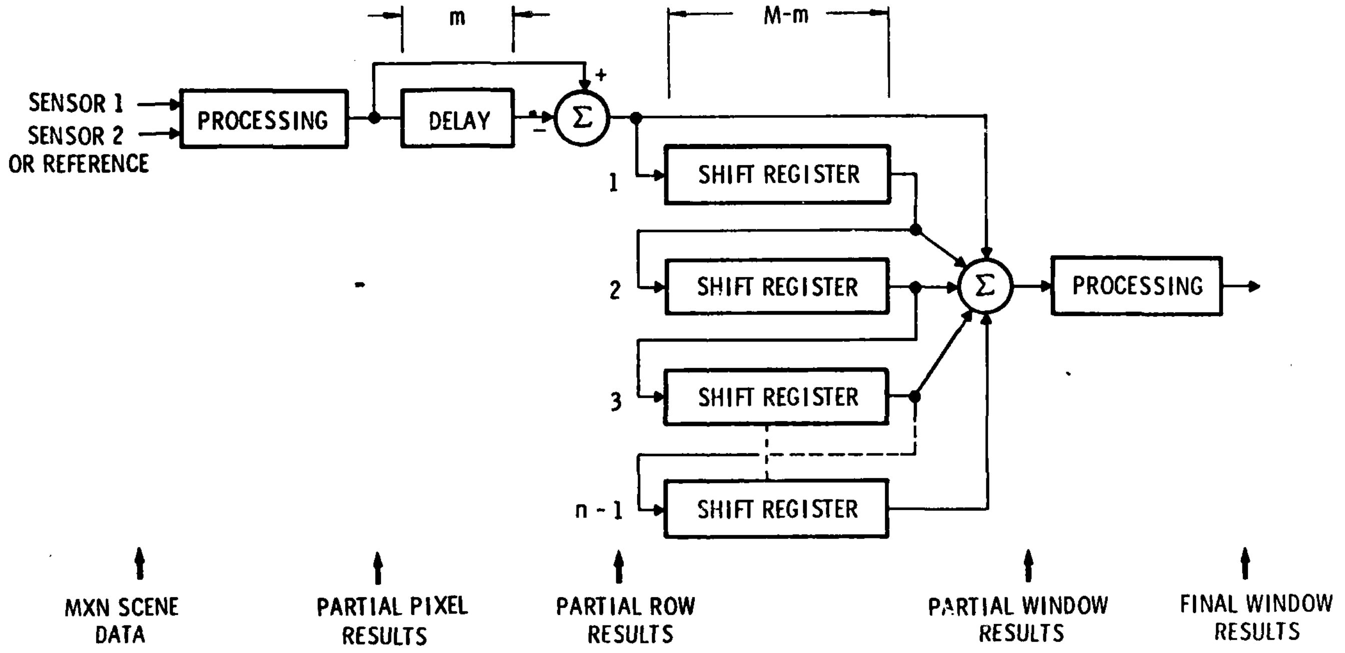


FIGURE 2.5

PIPELINE SUBFIELD PROCESSING

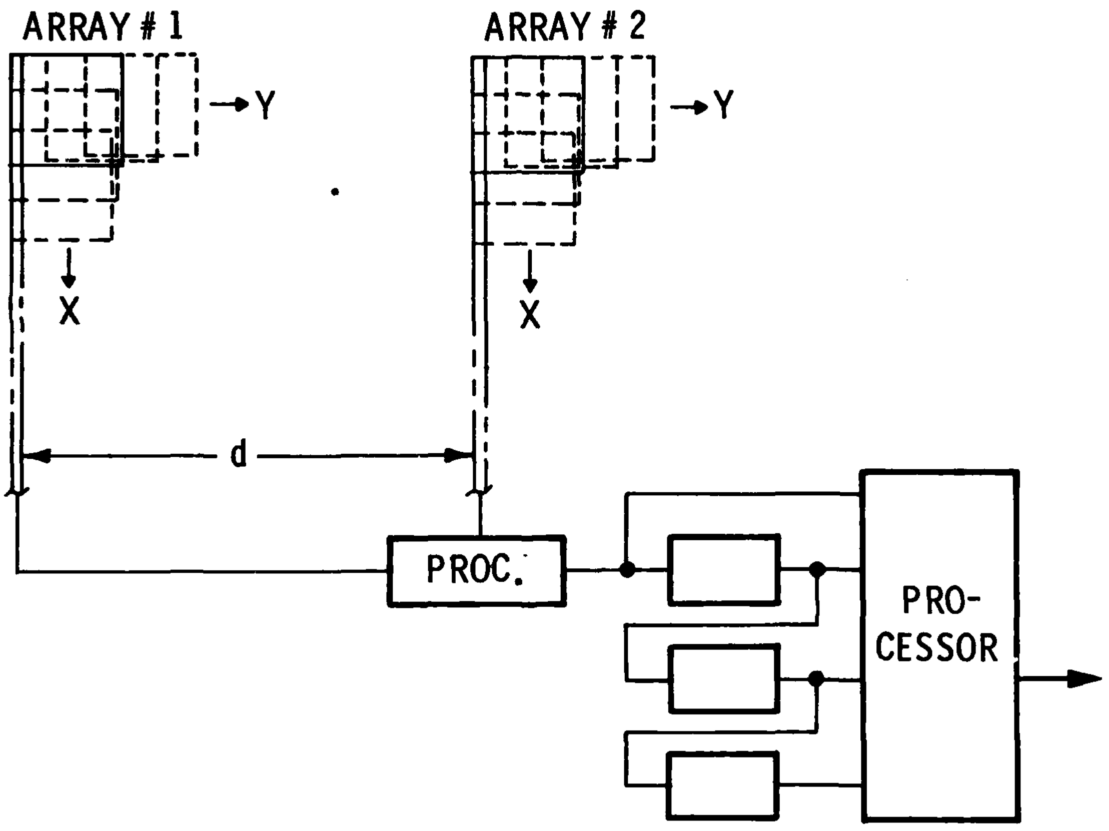


FIGURE 2.6

FLIR STABLIZATION MODULE
PIPELINE CORRELATOR BLOCK

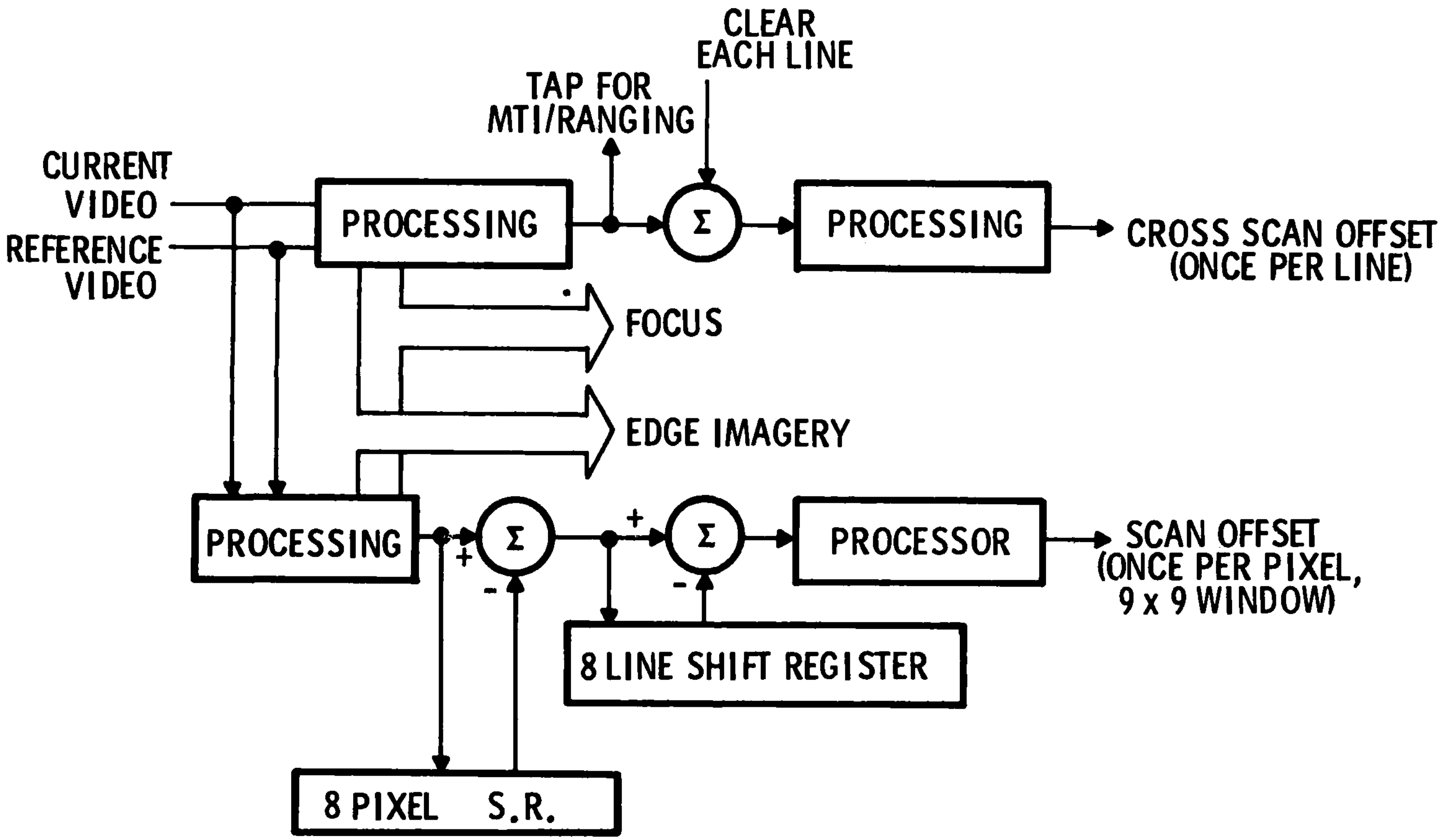


FIGURE 2.7

GYRO STABILIZATION

Gyroscopes are devices which measure the change in angular position or (not both) the angular rate to which they are subjected. Modern miniature gyros are small canisters which contain a small spinning mass (rotor). In the angular positon measuring mode, if the device is tilted, the rotor tends to remain at its original orientation and the difference in orientation between the rotor and the canister is measured by pickoffs. In the angular rate mode, the rotor orientation is locked relative to the canister and only the torque of attempted deflection resulting from an angular rate is measured.

Gyro performance is related to its cost. For example, a simple, miniature, rate-integrating gyro, which costs about $2000, typically has G-sensitive drifts to 25° to 50°/hr/G and -G-sensitive drifts of 30° to 60°/hr/G. Angular rates of response on these gyros are in the range of 100° to 200°/s.

More sophisticated gyros cost about $10,000 and have non-G-sensitive drifts of 15°/hr/G and G-sensitive drift of about 14°/hr/G. A portion of this high cost is attributed to the gas spin-axis bearings which give a mechanical time constant of a little over 1 ms, that is, the time in which a gyro responds to an excitement. The gas bearings also eliminate much of the gyro noise. Such gyro's can handle rate inputs up to 115°/s. The rate limiting factor in most gyros is the torque motor current limit due to wire size.

Non-G-sensitive drift is due primarily to spring in the internal leads that carry the electrical signals into and out of the gyro. This drift can be in a positive or negative direction and varies from gyro to gyro. It can be measured and trimmed to zero electrically or magnetically.

G-sensitive drift is due to a mass unbalance incurred in the mechanical construction of the gyro. For example, if on the floated rotor of a gyro there is a heavy spot on the side marked as an "A" in figure 2.8, it will cause the rotor to rotate about its output axis due to the effect of gravity or acceleration. G-sensitive drift can be negated by mounting a gyro so its output axis is parallel to the forces of acceleration. Another way to cancel these forces would be to use an accelerometer to supply a nulling signal. As each gyro is different, the scaled signal must be differentially tuned for each device.

G-SENSITIVE DRIFT

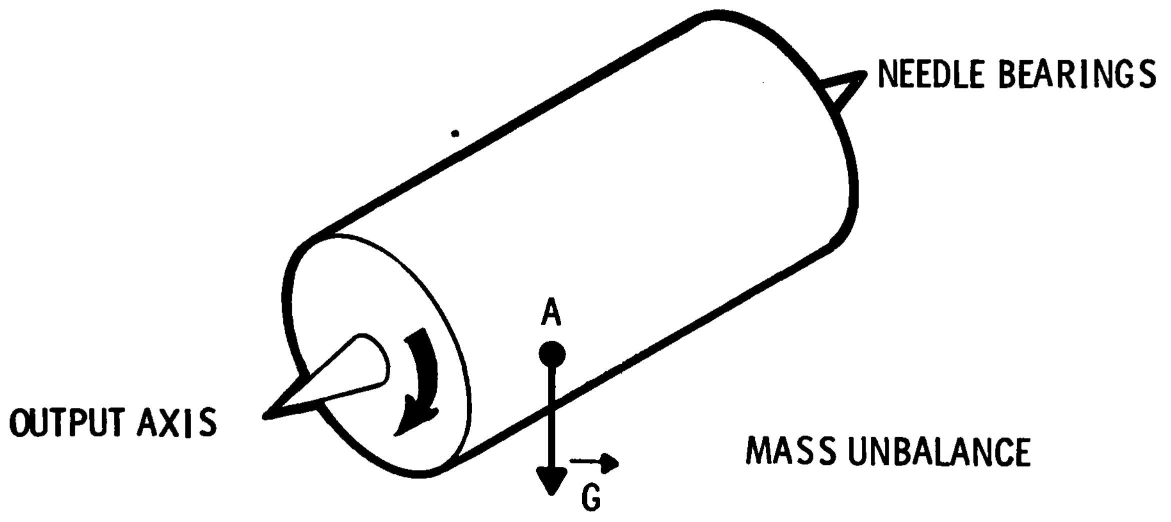


FIGURE 2.8

Other causes of gyro drift are a result of bearing frictions, magnetic influences and internal thermal gradients. These influences are normally considerably less than the G and non-G-sensitive drifts, but still must be considered in the design of an accurate system. Nearly all can be minimized, if not eliminated by careful circuit design and gyro construction, as is done for the inertial navigation systems on aircraft and missiles requiring extreme accuracy.

The floated rate-integrating gyro is ruggedized and usually can withstand an impulse acceleration of 200-300 G's before being damaged. When a gyro is damaged, it generally involves the output axis bearings. These bearings are either of the needle-jewel type or small precision ball or roller bearings. It should be realized that a gyro sitting on end and falling over to its side on a hard table can easily experience shocks of 300-350 G's, so that the use of the word "rugged" is relative. This is illustrated in figure 2.9.

The schematic of a typical gyro is shown in figure 2.10. A typical block diagram for an angular rate sensing gyro is shown in figure 2.11. The gyro and its circuitry should be visualized simply as one block of a servoed system normally entitled "single axis angular rate transducer". Figure 2.12 is a block diagram of an angular position transducer using a gyro. Figures 2.11 and 2.12 are only representative and, depending upon the magitudes of the quantities to be measured, may be implemented quite differently. Table II-3 lists some of the advantages and disadvantages of gyros.

COMPARISON OF TECHNIQUES

Inertial stabilization is a considerably different technique than scene correlation. Nevertheless, they can be quite easily compared. For the most part, measures of gyro transducer performance are fixed by the device. The only real exception is positional accuracy and maximum excursion in the position mode. While there are physical limits within the device on excursion (2° to 3°), it is possible to induce a centering torque and effectively scale down the rotational deviation of the rotor relative to the canister. Thus, greater excursions can be allowed at the expense of accuracy. This also means that the indicated position will drift back to zero while the device actually remains at its new orientation.

GYRO SHOCK DAMAGE

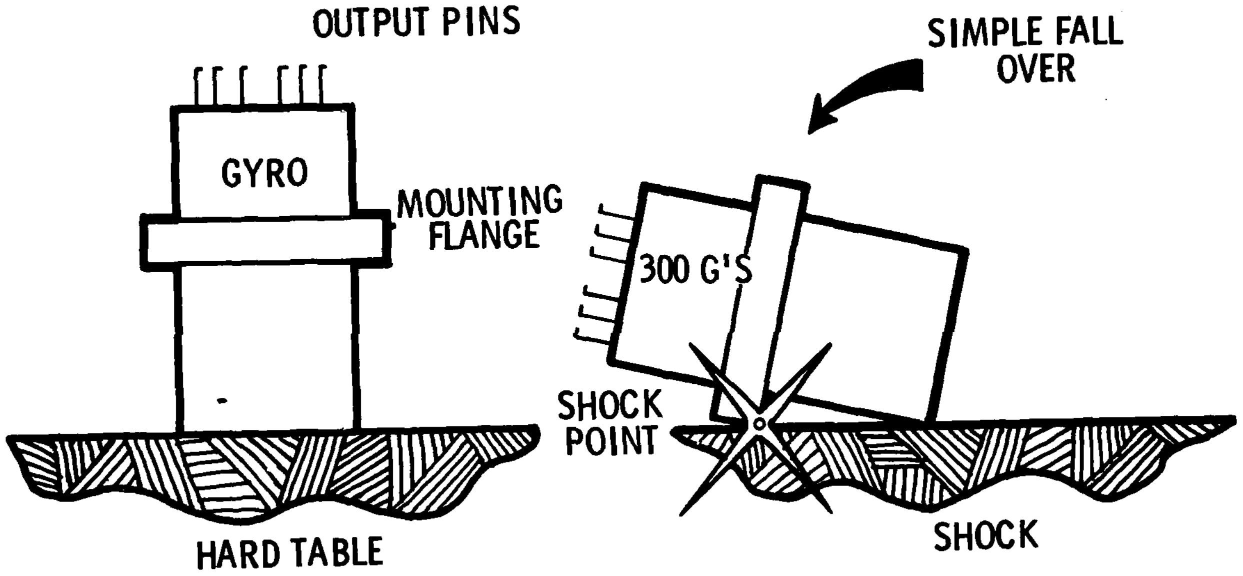


FIGURE 2.9

GYRO SCHEMATIC

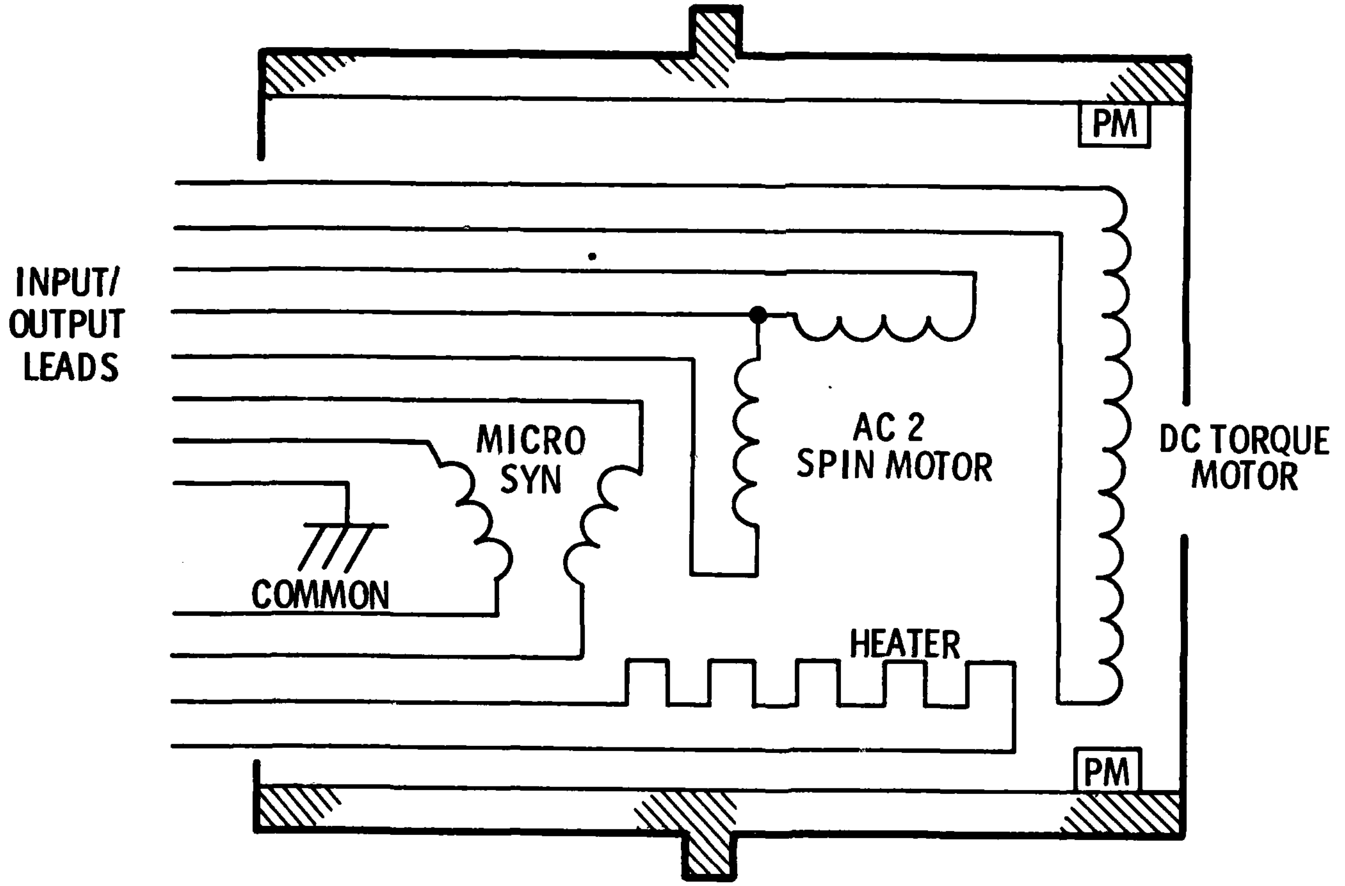


FIGURE 2.10

GYRO ANGULAR RATE TRANSDUCER

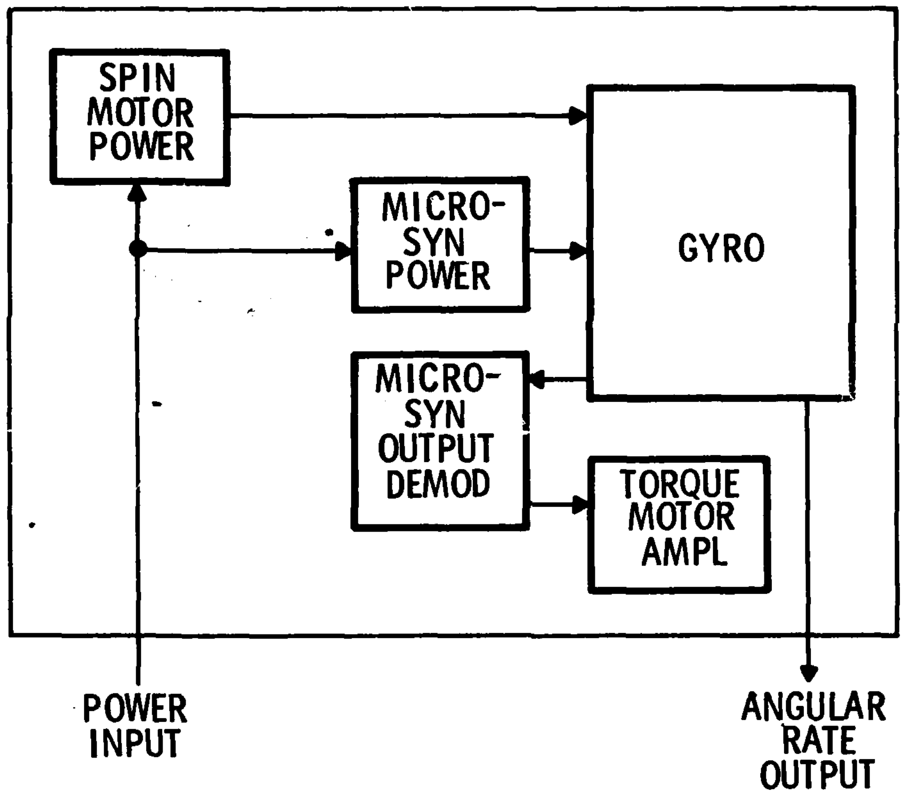


FIGURE 2.11

GYRO POSITION TRANSDUCER

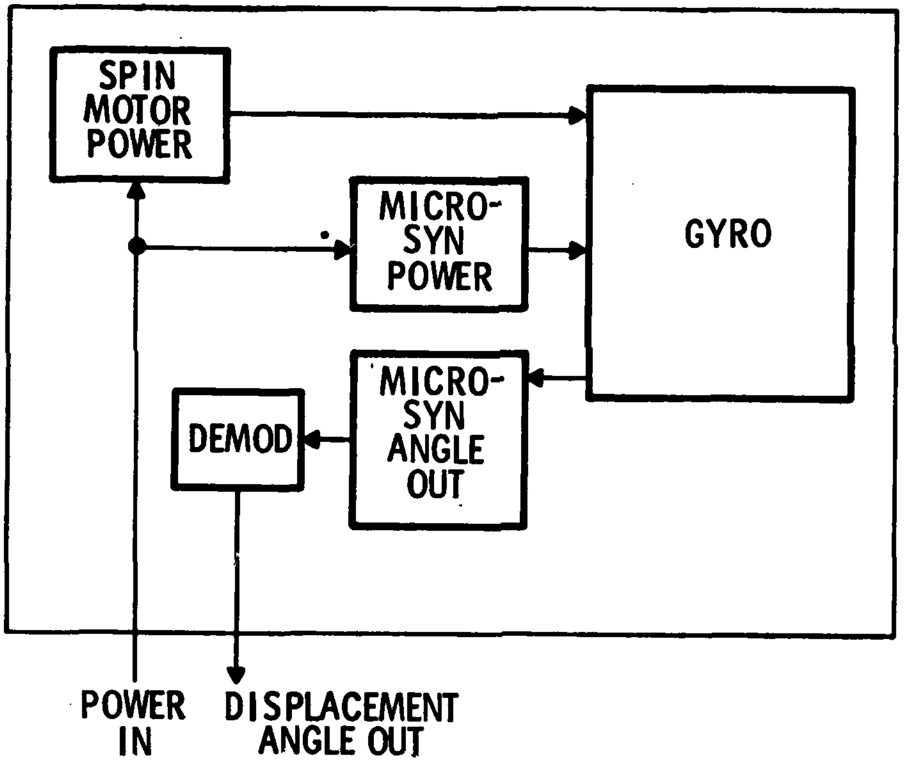


FIGURE 2.12

TABLE 11-3

GYROSCOPES

ADVANTAGES

- MULTIUSE FOR POSITION, ANGULAR RATE, ACCELERATION
- SMALL, LIGHTWEIGHT
- GOOD ACCURACY WITH KNOWN DRIFT CORRECTION
- CAN BE USED IN A MULTIAXIS SYSTEM
- DOES NOT REQUIRE VISUAL OR RADIO CONTACT

DISADVANTAGES

- NO TWO ARE EXACTLY ALIKE
- DRIFT ANOMILIES MUST BE CORRECTED
- REQUIRES TEMPERATURE CONTROL FOR PRECISE USE
- MECHANICALLY BANDWIDTH LIMITED (TYPICAL IS 100 Hz)
- EASILY DAMAGED; REQUIRES VERY CAREFUL HANDLING
- HAS BOTH ELECTRICAL AND MECHANICAL CONSTRUCTION PROBLEMS
- VERY EXPENSIVE (FOR A GOOD ONE)
- REQUIRES COMPLEX CIRCUITRY TO OPERATE
- MUST BE LOCATED AT THE UNIT BEING STABILIZED

Performance figures for scene correlation, on the other hand, are intimately related to the imaging sensor/detector. A direct comparison is acheived by specifying imaging parameters and must be repeated on a case-by-case basis. This is something of a moot point in the stabilization of imaging sensors, since the required performance is itself a function of these parameters.

The computer simulation of pipeline correlation, discussed in appendix C, has shown that the rms accuracy of this technique is conservatively on the order 1/20 of a detector dimension. This assumes at least 4-bit digitization, a signal-to-noise ratio of no less than 10:1 in the video signal, and a window of at least 64 elements. Even a signal-to-noise ratio of 2.5:1 allows accuracy to 1/8 pixel. The rms error of 1/20 pixel is used as the measure of performance for angles and angular rate accuracies. The mean of errors were shown to be virtually zero over 100 calculations, indicating that there is no drift with the correlation technique. Finally, the fact that pipeline correlation can lock on to offsets up to ±2 pixels allows calculation of maximum rate, since the offset must deviate by more than 2 pixels from line to line to prevent tracking of offsets. Using these results, table II-4 presents a comparision of gyro and correlation stabilization.

Several other considerations have been mentioned, such as the ruggedness, remote location, accounting for translation and skew, moving scenes and cost. In each case, the argumenτs favor pipeline correlation. The only penalties occur in size, weight and power. These must be offset against the extra capabilities provided in addition to stabilization, such as motion mapping, passive ranging and dynamic sampling.

In general, a pipeline correlator, as compared to gyro transducers, offers better performance, lower cost and greatly expanded capabilities at the price of size, weight and power.

TABLE 11-4

PERFORMANCE COMPARISON

	GYRO TRANSDUCER	PIPELINE CORRELATION (2.5 MHz)		
		IFOV		
MEASURE		.5 MRAD	.167 MRAD	
BANDWIDTH	100 Hz	14.4 kHz	14.4 kHz^{+}	(LINE FREQUENCY)
POSITION MODE* ACCURACY	±1 TO 11.MRAD	±24 μRAD	±8 μRAD	(0.05 IFOV)
MAXIMUM EXCURSION	2.5 TO 28^{0}	NO LIMIT	NO LIMIT	
RATE MODE* ACCURACY	0.2%	±.042^{0}/S	±.014^{0}/S	(0.05 IFOV x FRAME RATE)
MAXIMUM RATE	100^{0} TO 200^{0}/S	1650^{0}/S	550^{0}/S	(2IFOV x LINE RATE)
DRIFT	15^{0} TO 50^{0}/HOUR/G	NONE	NONE	

* GYRO TRANSDUCER MAY MEASURE POSITION OR RATE, NOT BOTH SIMULTANEOUSLY

+ EFFECTIVE 3.2 kHz IN SCAN DIRECTION DUE TO DELAY

OTHER OUTPUTS

The pipeline correlation process performs several calculations which are useful for applications other than stabilization.

For example, a measure of edge is performed in both horizontal and vertical axes. These measures are tapped for possible use as edge maps of the video. If two-axis edge mapping is desired, the edge outputs may be thresholded and the results used as inputs to a logical OR gate. The results are output in the same video format as the sensor output.

A second function calculated is a measure of autocorrelation width which is equivalently a measure of focus, as noted in the discussion on correlation. There are other causes of broadened autocorrelation peaks, such as image smear. However, the object of this module is to eliminate such motions; thus, the first task is to stabilize the image and then, if the image is still blurred , the measure of focus output will permit sharpening the focus by means of a servo-controlled focus drive on the sensor. The measure of focus outputs are given in both axes. A mismatch in signals indicates some effect other than defocus to be the cause of blur. Thus, only the signal indicating sharpest focus should be used in correcting focus.

Several other features require optional boards and/or modules and will be discussed in later sections.

NEED FOR FRAME STORAGE

Because correlation stabilization is based upon comparing sequential frames, it is necessary that the video signal be stored for comparison with the next frame. It is also necessary that the current video signal and the stored previous video signal be presented to the correlator in the same format. This fact leads to the requirement for two frame storage planes. If both inputs to the pipeline correlator are to be in the original FLIR sensor format, then one frame store to preserve a frame for correlation and a second frame store to reformat for display are needed. If, on the other hand, both inputs are to be in the final output display format, then two frame stores are still required: one to reformat and one to store the previous frame in reformatted format. Since two frame stores are required, regardless of configuration, the better approach is to use the FLIR sensor format. Not only is the back and forth scanning with vertical line format preferrable to simplify the implementation of pipeline correlation, but the delay of reformatting is avoided, allowing optimal bandwidth of the correction signal. Frame store configurations are presented in figure 2.13.

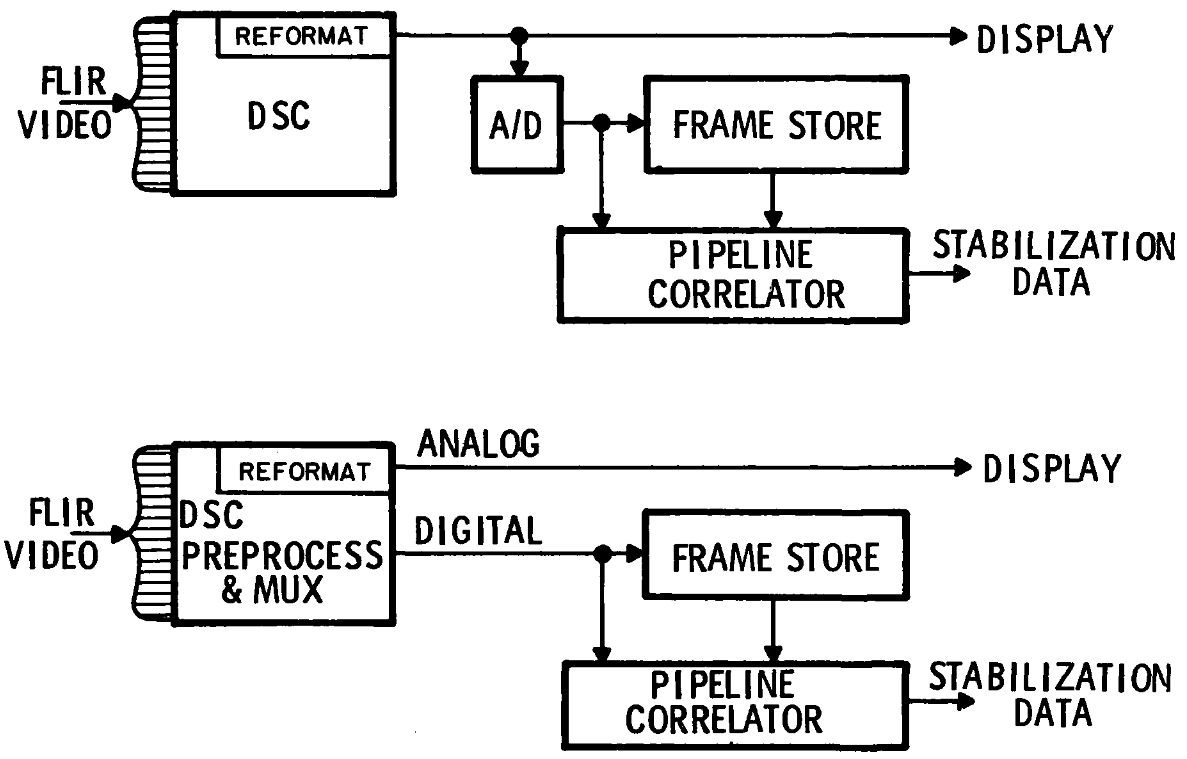
The question arises whether a DSC reformatter could be used as a simple frame store for correlation. Although attractive, such a solution would be, unfortunately, impractical. The preferred input for the stabilization module is digital; this, however, is a minor point since there is little doubt digital output from the reformatter could easily be supplied. The most serious defect in attempting to use a DSC reformatter is the rigidity of its operation. While it can be reprogrammed for specific video formats, it cannot easily handle changing address offsets on a pixel-by-pixel basis as required to track offsets between the current and previous frames. Finally, the amount of interfacing between pipeline correlator and frame store, as well as modification to the latter if it were a DSC reformatter would, in the end, result in a frame store dedicated to pipeline correlation.

It is, then, easier and more efficient to design a frame store specifically for this application.

FRAME STORE CONSIDERATIONS

The pipeline correlator tracks offsets from frame to frame. If there are sections of imagery which do not overlap the previous frame, the correlator can provide no usable offset data and an effective dead time occurs. Naturally, if the sensor is not actively stabilized, such occasions are bound to occur. Some dead time is tolerable. So long as the shift which occurs duing the dead time is within a few pixels, the correlator will be able to pick up where it left off as shown in figure 2.14.

FRAME STORE CONFIGURATION APPROACHES



NOT PREFERRED

- HIGH (EMPTY) BANDWIDTH VIDEO & LARGER FRAME STORE
- DELAY, LOWER BANDWIDTH STABILIZATION DATA
- POORER FORMAT FOR CORRELATION
- ANALOG OUTPUT OF DSC REFORMATTER

PREFERRED

- OPTIMAL BANDWIDTH VIDEO
- MINIMAL DELAY
- GOOD FORMAT FOR CORRELATION
- DIGITAL INPUT TO CORRELATOR

FIGURE 2.13

CORRELATION DEAD TIME

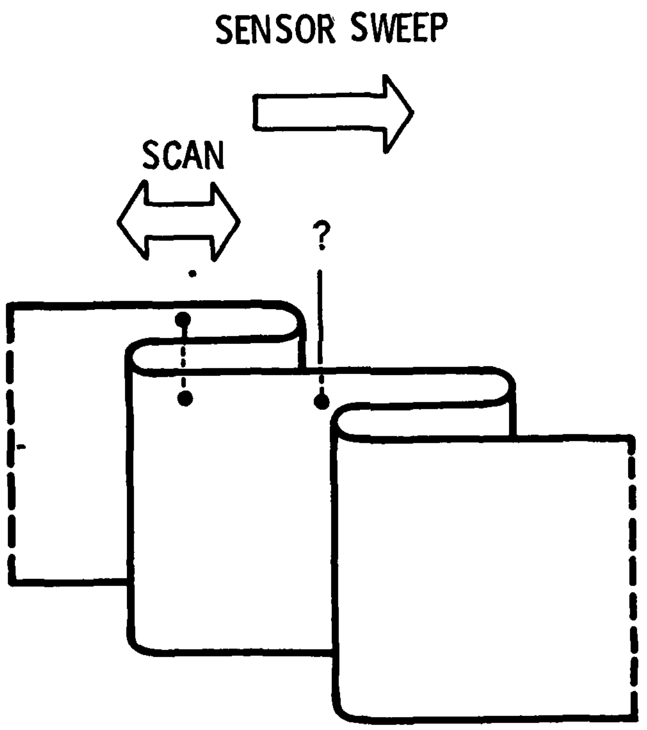


FIGURE 2.14

The requirement is that during the dead time, the line offset must not vary by more than 2 pixels. The dead time (τ_O) is given by:

$$\tau_O = 2 \frac{O_L}{N_L} t_{FLD}$$

where:

O_L = frame-to-frame offset in lines
N_L = number of lines per field
t_{FLD} = field time

The factor of 2 is due to the traceback. Now, O_L is in turn given by:

$$O_L = O_\theta / IFOV$$

where:

O_θ = angular offset
IFOV = instantaneous field of view of the line

Finally:

$$O_\theta = \omega_s \times t_{FR}$$

where:

ω_s = angular rate of the sensor
t_{FR} = frame rate

Note that offset is constant with constant angular rate.

Hence: $$\tau_O = \frac{2\omega_s t_{FLD} t_{FR}}{N_L \; IFOV} = \frac{\omega_s t_{FR}^2}{N_L \; IFOV}$$

Now, since the allowable change in offset is two lines, the allowable change in angular rate ($\Delta\omega_s$) is:

$$\Delta\omega_s = \frac{2 \; IFOV}{t_{FR}}$$

which must occur during the dead time. The sensor must then experience an acceleration (α_s) of $\Delta\omega_s / \tau_D$:

$$\alpha_s = \frac{2 \; IFOV^2 \; N_L}{\omega_s \; t_{FR}^3}$$

The condition which must be satisfied is $\alpha_s \omega_s$ Q where Q is given in table II-5 for the lines per frame and fields of view considered.

TABLE II-5

Q (degrees^2/second^3)

	WFOV	NFOV
360 lines	15,900	1,770
480 lines	21,200	2,360

Assuming sinusoidal excitation, the peak product of acceleration and velocity occurs at 45°. Therefore, the maximum angular rate would be $\sqrt{2}$ times the square root of the product. Table II-6 gives the maximum allowable angular rates of sinusoidal excitation, above which offset lock may be broken.

TABLE II-6

Maximum Sinusoidal Rate

	WFOV	NFOV
360 lines	138°/s	59.5°/s
480 lines	206°/s	68.6°/s

A similar condition occurs at initiation when there is no previous frame against which to correlate. This is equivalent to an effective infinite angular rate. To assure initial offset lock, the system merely operates as though the second field was a reversed copy of the first field. With the back and forth scan of the scan mirror, there is only a 1/2-pixel cross scan offset at turnaround. Offsets may easily be tracked from a near null condition by this method. Therefore, even if the conditions become sufficiently severe to break offset lock, the system will simply reinitialize automatically.

All of the foregoing analysis is based upon frame-to-frame correlation. Clearly, if this frame store were allowed a stabilized mode in which the scene is locked and the sensor is not actively stabilized, the system would quickly lose offset track and be unable to recover without locking in a fresh scene. Scaling an image in the memory would create a more severe situation, whether or not the sensor was actively stabilized since a long dead time is assured for this condition. For this reason, no options of passive stabilization or scaling are allowed with this frame store. Because passive stabilization is not feasible for this storage plane, neither is scene integration. The frame store must be a simple temporary storage.

Manipulations of the imagery suggested above will be handled by a separate frame store which is not used for correlation, but will use correlation results. This special function frame store will be described in section V.

Being unstabilized, video data is written into the basic frame storage just as it appears from the sensor, that is without offset. However, to track offsets, the memory is read with offsets dictated by the correlation results. Because cross scan offsets are calculated line by line and with no delay, the bandwidth of this offset is equal to the line rate reciprocal. The offset of each line determines the offset with which to read subsequent lines for correlation. In the scan direction, however, the windows are 9 lines wide. Thus, even though the data output rate is equal to the pixel rate, the results are delayed 4.5 lines. The higher rate of output allows correction for skew and rotation, but the delay results in a response bandwidth of the line rate reciprocal divided by 4.5.

These considerations are summarized in table II-7.

TABLE II-7

FRAME STORE RATIONALE

- DEAD TIMES MAY OCCUR BETWEEN FRAMES WITH WIDE ANGULAR SWEEPS
- HIGH ANGULAR RATES WITH HIGH ANGULAR ACCELERATION AT LOW FREQUENCY MAY CAUSE LOSS OF OFFSET TRACK
- IF OFFSET TRACK IS LOST, REBOOT IS REQUIRED
- STABILIZED IMAGERY CANNOT RECOVER AFTER REBOOT
- SCALED IMAGERY WOULD HAVE LONGER DEAD TIMES AND MORE SEVERE CONSTRAINTS
- CORRELATION FRAME STORE MUST BE SIMPLE, UNSCALED, UNSTABILIZED
- SCALING, INTEGRATION & PASSIVE STABILIZATION HANDLED WITH SEPARATE "SPECIAL FUNCTION" FRAME STORE

SECTION III
MTI/RANGING OPTION

MTI CONCEPTS

Scene correlation is really far more versatile than has been described in section II. Figure 3.1 illustrates several possible applications.

Correlation stabilization may easily be considered to be the tracking of the scene within the FOV as a whole. Due to the fact that perturbations may occur within a frame, the correlation stabilization described here actually maintains a running track on subportions of the field of view.

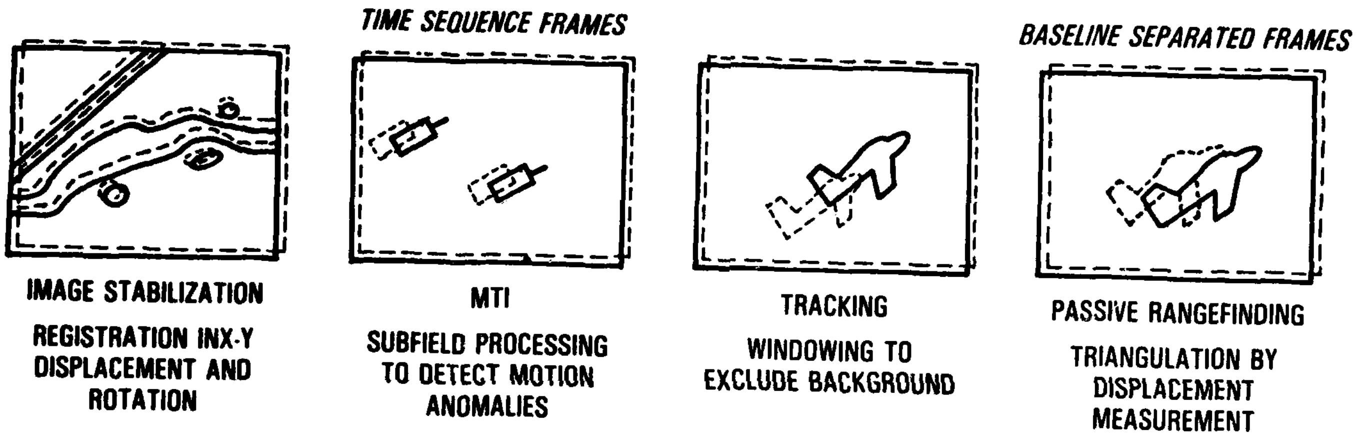
The perturbations are not necessarily due to sensor motion, but may in fact be due to scene motion. If one or more small portions of the scene continuously exhibit a perturbation considerably different from the scene as a whole, they may safely be considered moving targets. Thus, the correlation stabilization results may be used for moving target indication (MTI) by subtraction of the average frame-to-frame offset. The difference is then compared to a threshold value and any point sufficiently different is illuminated as a moving target. This concept is shown in figure 3.2.

At this time, it should be noted that the pipeline correlation described calculates cross scan offset on a line-by-line basis where the window is exactly one line in length and width. Clearly, this is of little value for detecting cross scan motion. More desirable is a square window. Thus, the partial results for the cross scan motion must be tapped off before summation over the 1-line window and instead summed over a square window. This provides running results for each pixel in the field of view.

The offset calculated by the pipeline correlator is proportional to angular rate of motion. There is, in fact, no need to subtract the average and threshold the differences if the application is more suited to a relative motion map than MTI. The video format offset results may be taken directly to provide a video image of motion where brightness is proportional to velocity. Alternatively the

526—25

IMAGE REGISTRATION TECHNIQUES


TIME SEQUENCE FRAMES
BASELINE SEPARATED FRAMES
IMAGE STABILIZATION
REGISTRATION INX-Y DISPLACEMENT AND ROTATION
MTI
SUBFIELD PROCESSING TO DETECT MOTION ANOMALIES
TRACKING
WINDOWING TO EXCLUDE BACKGROUND
PASSIVE RANGEFINDING
TRIANGULATION BY DISPLACEMENT MEASUREMENT

MTI

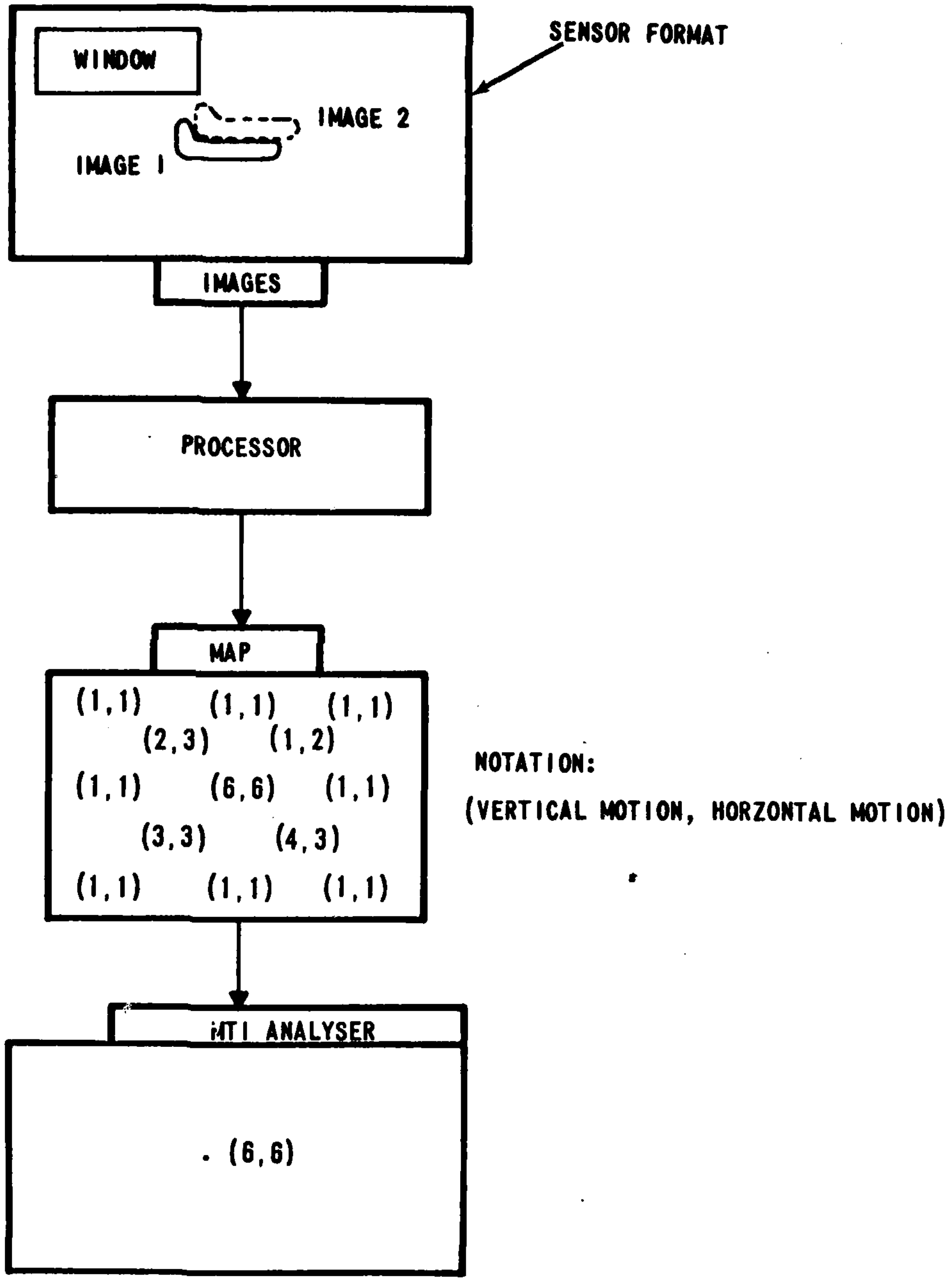
- DETECTS MOVING TARGETS
- STABILIZES/TRACKS MOVING TARGETS AGAINST FIXED BACKGROUNDS

SIGNATURE MATCHING

- IDENTIFIES TARGET BY COMPARISON TO "LIBRARY" IMAGE

FIGURE 3.1

MOVING TARGET DETECTION INFORMATION FLOW



SENSOR FORMAT

WINDOW

IMAGE 1

IMAGE 2

IMAGES

PROCESSOR

MAP

(1,1) (1,1) (1,1)
(2,3) (1,2)
(1,1) (6,6) (1,1)
(3,3) (4,3)
(1,1) (1,1) (1,1)

NOTATION:
(VERTICAL MOTION, HORZONTAL MOTION)

MTI ANALYSER

. (6,6)

FIGURE 3.2

physical velocity may be calculated by multiplying the offset by the constant of proportionality (C_v) which is given as:

$$C_v = \frac{P\ R}{f1} \times \dot{F}$$

where:

P = pixel size

fl = focal length

R = range to target

$\dot{F}$ = number of frames/s

The absolute velocities may be used either for another processor, or the system could actually be configured to display the velocity of a particular target on a control panel display (to be described later). Alternately, this velocity in relative terms could be fed to the sensor servomechanisms to implement tracking.

The possible ouputs which are made available by this simple addition to the stabilization module are shown in figure 3.3.

The performance of this option is easily calculated from the angular rate accuracy quoted above for stabilization performance. The accuracy may be thought of as noise equivalent angular rate. Neglecting factors of oblique viewing angles, this works out to .51 mi/h per 1000 ft of range for the wide field of view, and .17 mi/h per 1000 ft of range for the narrow field of view. Similarly, the maximum velocities tolerated are determined by the limit of 2 pixels of offset. These relationships are graphed in figure 3.4.

PASSIVE RANGING CONCEPTS

Thus far, only the correlation of time sequential images which leads to various forms of motion analysis has been discussed. The process could be conceived as the correlation of two video signals derived from two sensors displaced in time. Consider instead the correlation of two video signals derived from two sensors displaced in space; that is, separated by a fixed distance. If bo. sighted, they will image the same scene but from slightly different angles as

MTI PROCESSING

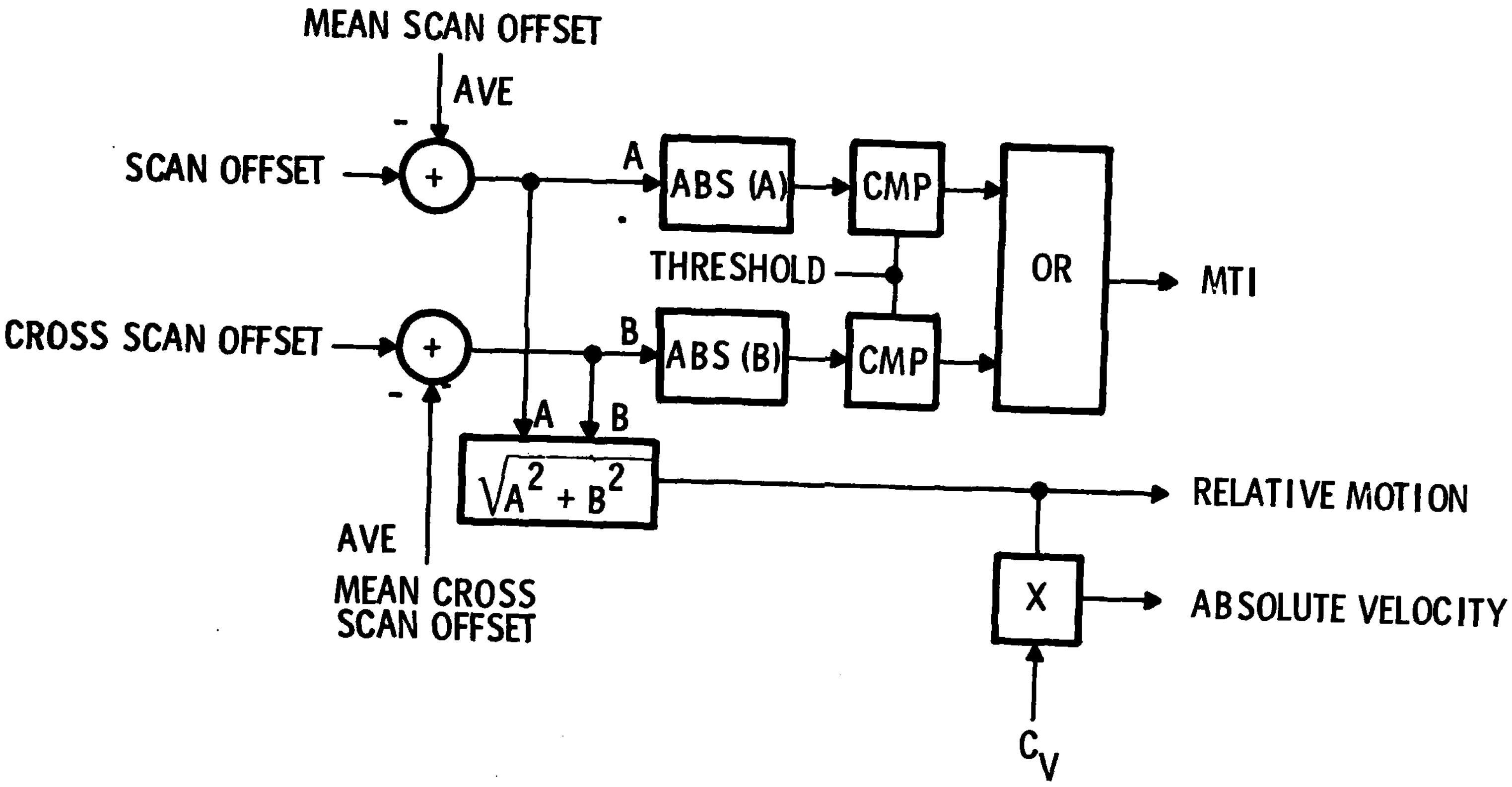


FIGURE 3.3

MTI OPTION – VELOCITY DETECTION LIMITS

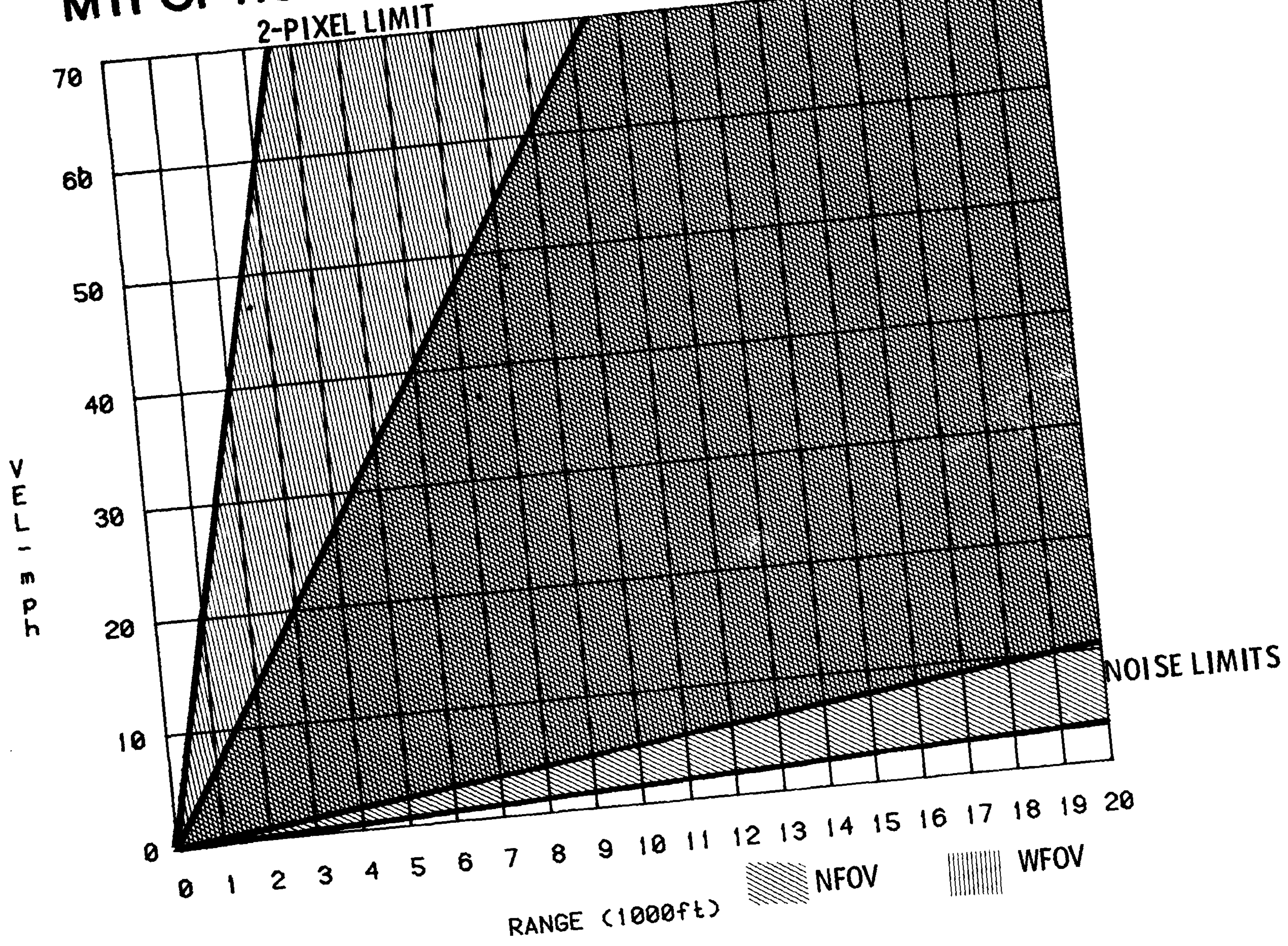


FIGURE 3.4

shown in figure 3.5. It is easily shown by similar triangles:

$$\frac{R}{B} = \frac{F}{D}$$

and: $$R = \frac{BF}{B}$$

where:

R = range to target

B = separation between lenses

F = focal length

D = apparent image shift

The absolute range to a target may thus be calculated using stereo sensors and maintaining precise alignment between the two sensors. Since the accuracies of offset measurement are approximately 8 µrad, such precision is clearly beyond the limits of mechanical tolerance. These misalignments may, nevertheless, be measured and corrected by presenting each sensor with an artificial infinity target. Such techniques are known as autocalibration, and are required only if absolute range calculations are desired. This concept is shown in figure 3.6.

It is important that the detectors have fixed geometric positions to the baseline axis, so that image offsets may be reliably calculated. For this reason, the baseline separation must be in the cross scan direction. The scan direction offsets are then used to synchronize scan mirrors.

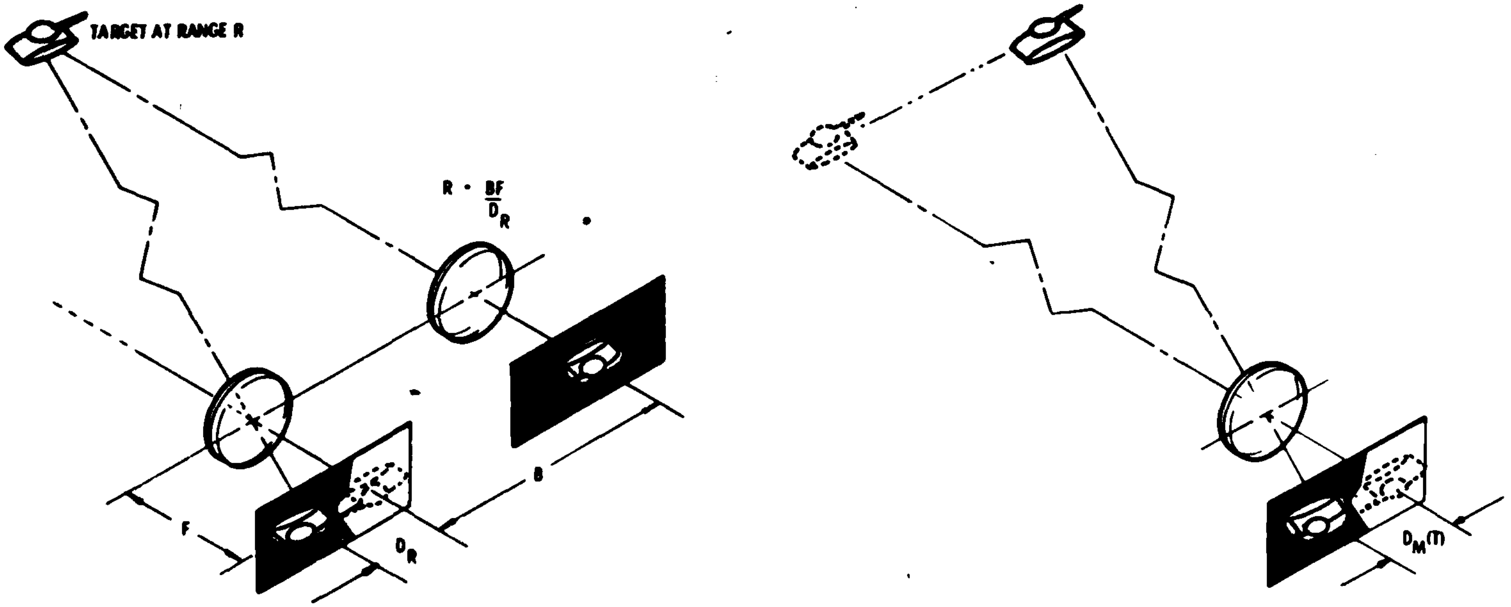
Since the baseline separation and focal length are essentially fixed quantities, range error (ΔR) is determined by differentiating R = BF/D:

$$\Delta R = -\frac{BF}{B}\Delta D = -\frac{R^2}{BF}\Delta D$$

where:

ΔD = error in measuring D the image shift

RANGING AND TRACKING GEOMETRY



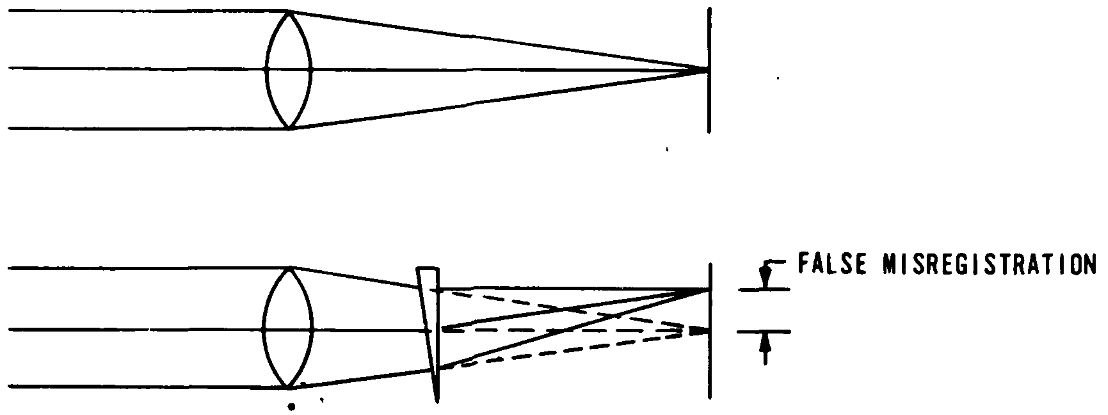
RANGING

TRACKING/STABILIZATION

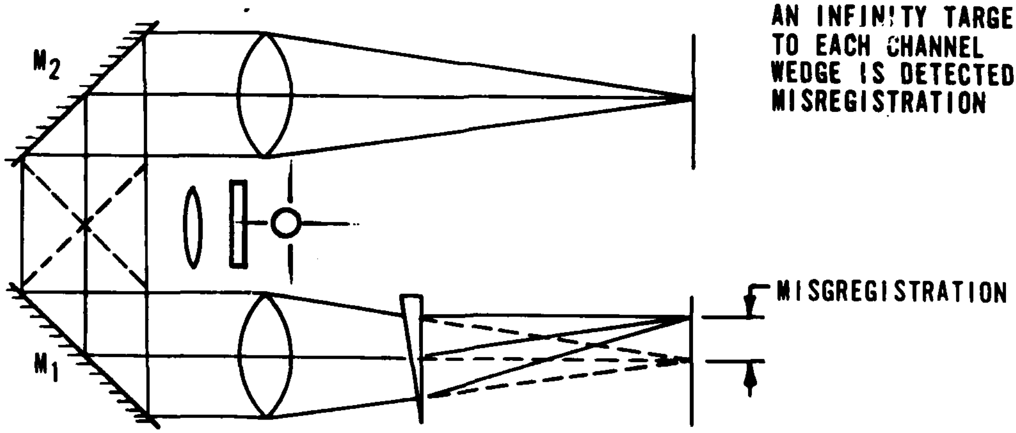
CAI

FIGURE 3.5

FALSE MISREGISTRATION RESULTING FROM AN OPTICAL WEDGE



AUTOCALIBRATION SYSTEM



AN INFINITY TARGET IS PRESENTED
TO EACH CHANNEL
WEDGE IS DETECTED AS A
MISREGISTRATION

FIGURE 3.6

Since D has been defined as offset (in pixels):

$$D = O \times P$$

and:

$$\Delta D = P \times \Delta O$$

where:

O = offset

P = pixel dimension

ΔO = error in offset calculation

Thus, the scaling factor for the reciprocal of offset is $C_R = \frac{BF}{P}$

and the ranging error is $\Delta R = -\frac{R^2}{B \times F} P \Delta O = -\frac{R^2 \, IFOV}{B} \Delta O$

Note that the same options are available for ranging as for motion detection. A threshold may be set such that only objects within a specific range are illuminated. Alternatively, as shown in figure 3.7, the offsets may be taken as is for relative range mapping or may be scaled appropriately to obtain absolute range values. Figure 3.8 shows expected ranging accuracies in such cases. The accuracy in ranging is clearly different than active methods which measure the time between the transmission of some signal and its refelected return. Active methods typically have fixed accuracies regardless of range, while passive stereometric ranging has a range squared dependence. While ranging performance is not exceptional because less than an optimal configuration is used, it is reasonable, especially compared to recognition range. There is little value in knowing precise range if it is not known what is being fired upon. Since the error in ranging is statistical rather than systematic, considerable improvement can be obtained by averaging results. Results are produced for each field, therefore a 10:1 accuracy improvement is obtained by averaging 100 samples yielding a ranging bandwidth of 0.3 Hz for a given target. This improved accuracy is also graphed in figure 3.8 by reading the accuracy scale on the right side. Absolute range values may be fed to another processor or displayed on the control panel by windowing a select target.

Since both video signals are supplied from separate sensors to the pipeline correlator, there is no need for frame storage. This is true only if passive ranging

RANGE PROCESSING

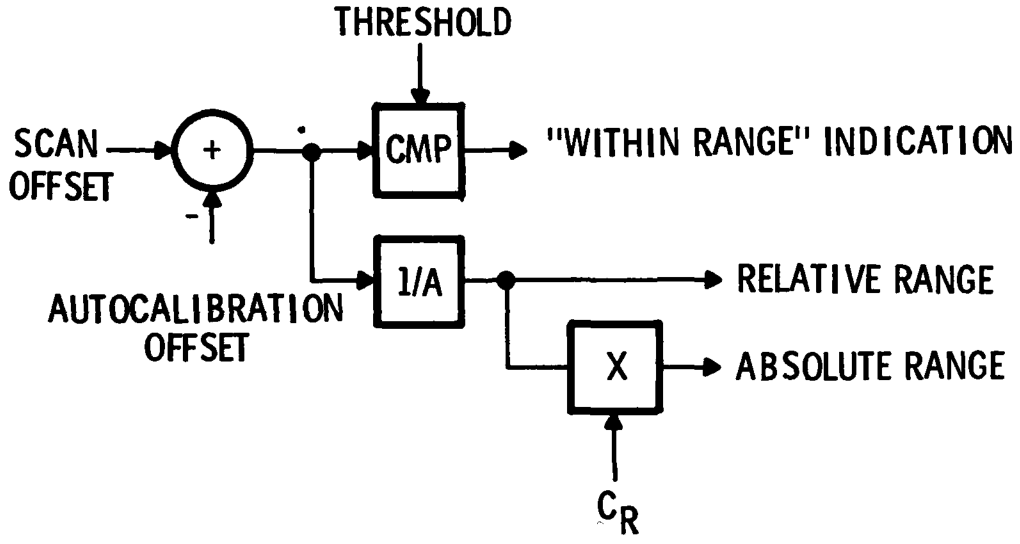


FIGURE 3.7

PASSIVE RANGING ACCURACY FOR COMMON MODULE FLIR'S

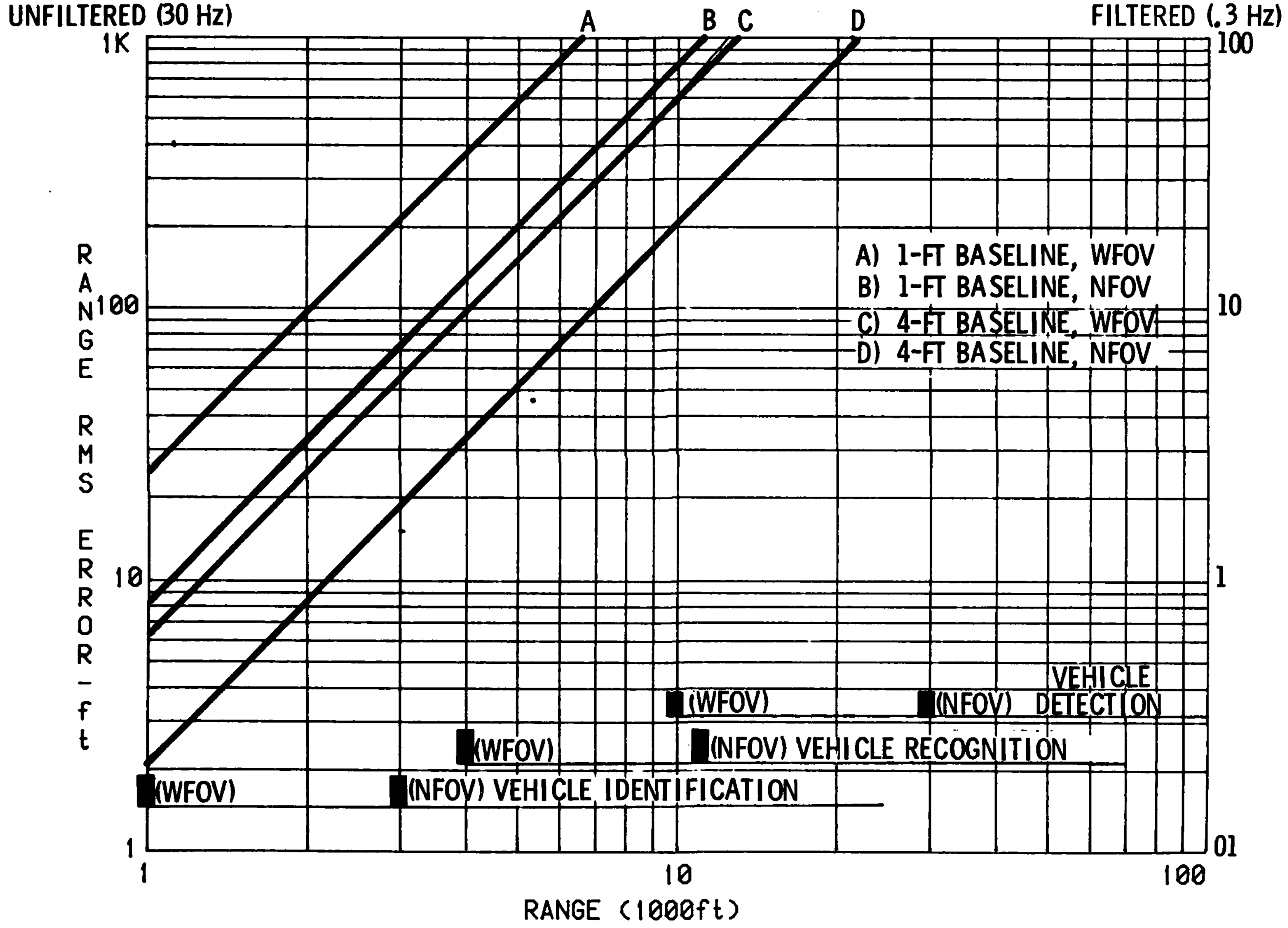


FIGURE 3.8

is the sole function. If ranging and some motion analysis are desired, the frame store, a third pipeline correlation card, and perhaps the second MTI/range card would be required.

CONTROL PANEL

For the basic stabilization module, very little control is required and no control panel is needed. With the MTI/ranging option, however, a remote control panel becomes more of a necessity. Even for the simple MTI/"within range" modes, some threshold must be supplied. When absolute values are desired, the appropriate terms to calculate a scale factor need to be supplied.

For these purposes, a simple control panel consisting of a keyboard, an alphanumeric display, a threshold potentiometer and an on/off switch is recommended. This panel allows remote operation. The display also allows alerting the operator to possible error conditions in the module functions.

An additional plug-in control panel can be used to window a selected target and display on the alphanumeric display, absolute velocities or ranges. This plug-in panel will also be used in conjunction with the special function module and will be described more fully in section V. For the purpose under discussion here, part of its function is to define a window on the video display and position that window to a desired location. The operator must locate the window over the selected target and request velocity and/or range of the target on the keyboard. The appropriate datum of the absolute value output from the motion/range map will be latched and displayed on the alphanumeric display. Figure 3.9 illustrates the several modes of operation.

When the control panel is used, there must also be a control receiver card in the stabilization module to interface the function.

MTI/RANGE OPTION MODES OF OPERATION

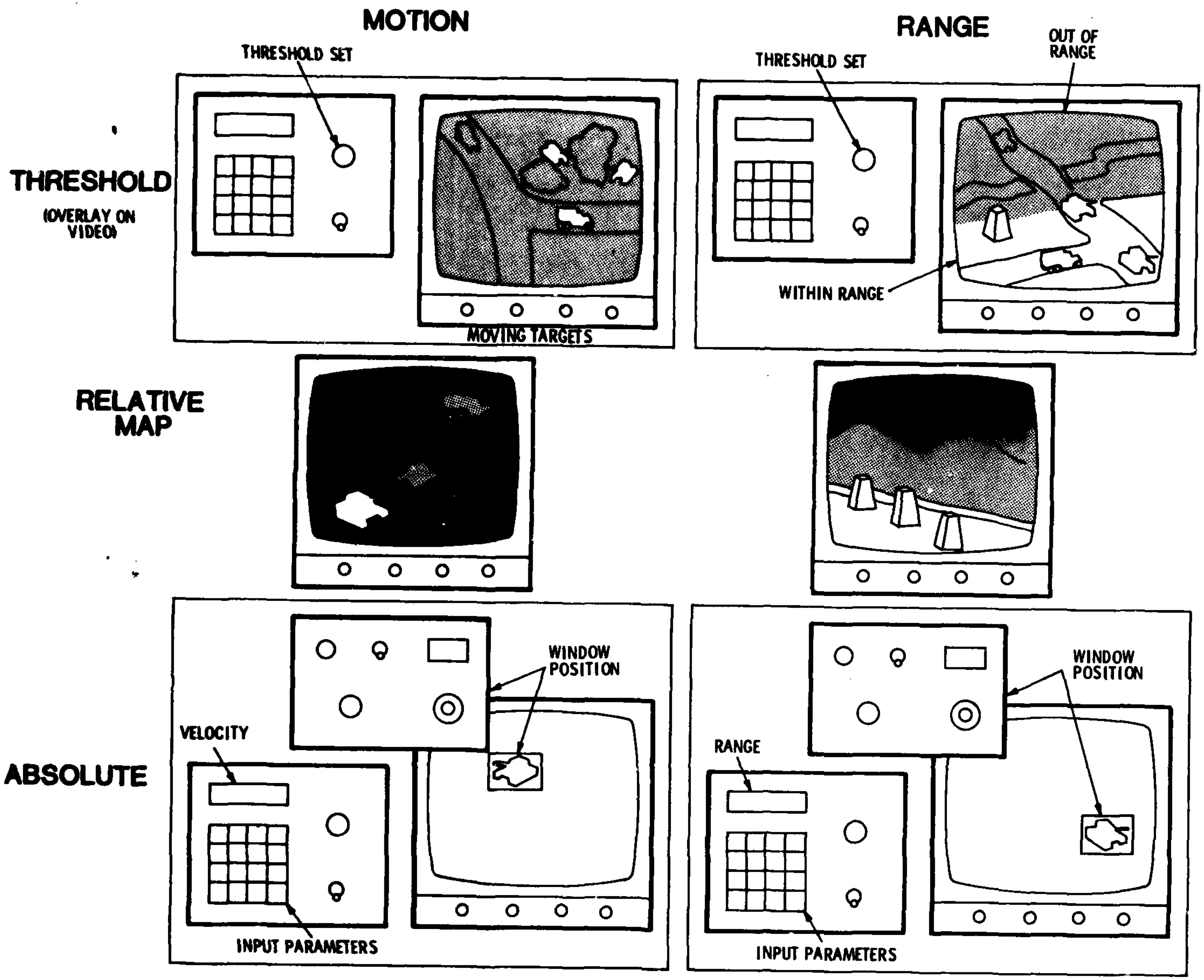


FIGURE 3.9

SECTION IV
ACTIVE VERSUS PASSIVE STABILIZATION

There are two means by which imagery may be stabilized:

- Active stabilization controls the sensor
- Passive stabilization manipulates the image after the fact

Active stabilization requires servo control of the sensor which results in larger, more complex sensors and higher probability of damage. The advantages are that it eliminates blur and allows continuous video update of the scene of interest. In addition, active control permits tracking of particular objects and proper search of a region larger than the field of view.

Passive stabilization also has advantages. It is not always feasible or even possible to equip a sensor with active stabilization. Passive stabilization permits remote location from the sensor. Furthermore, in some cases the image is already being manipulated for other reasons, and it may be a trivial addition to stabilize the image as well in memory or on a display. Passive stabilization can also stabilize for image motions outside the realm of those normally handled by active means, such as rotation, skew and dilation. Finally, a peculiar advantage of passive stabilization is the ability to stabilize one portion of the scene while allowing the sensor to wander about and simultaneously display other nearby areas of possible interest.

The latter point suggests that active and passive stabilization are not mutually exclusive and may in fact be advantageously used together in certain situations. Table IV-I summarizes the above points.

TABLE IV-I

COMPARISON OF STABILIZATION MODES

ACTIVE

- ELIMINATES MOTION BLUR
- MAINTAINS SCENE WITHIN FOV
- ALLOWS TRACKING
- ALLOWS SEARCH

PASSIVE

- REMOTE LOCATION
- NONMECHANICAL
- IMPROVED PROTECTION
- OTHER FUNCTIONS EASILY ADDED OR IF OTHER FUNCTIONS ALREADY EXIST, STABILIZATION IS EASILY ADDED
- STABILIZES FOR MOTIONS OTHER THAN IMAGE TRANSLATION
- ALLOWS DUAL DISPLAY: SCENE OF INTEREST AND SURROUND

MAY BE USED TOGETHER

CAI

SECTION V
SPECIAL FUNCTION MODULE - DESCRIPTION

BENEFITS

In many instances, imagery is of little or no value due to lack of stabilization, poor scale, poor signal-to-noise ratio, insufficient resolution or fleeting collection time. The special functions module provides improved imagery when it is not feasible or possible to address these problems with sensor modifications.

For example, if the sensor is fixed rather than servo controlled, imagery may still require stabilization. The special function module provides passively-stabilized imagery. If the area of interest is just a small area of the field of view, the area may be blown up in the special function module. Low signal-to-noise ratio can be improved by longer dwell or integration time. However, there are so many constraints involved with this parameter, it is generally impossible to change. The module provides an artificial dwell time increase by integrating frames in memory with no effects on system operation. Insufficient resolution is also addressed with the special function module as it includes dynamic sampling capability, the integration of detector outputs by phase. Finally, the problem of fleeting scenes is generally handled by freeze frame operation. This module incorporates and improves upon the operation by allowing image update so long as the sensor contains some or all of the frozen frame within its field of view.

In general, the special function module provides usuable imagery in even the most unfavorable circumstances. Its advantages are summarized in table V-I.

DESCRIPTION OF FUNCTIONS

The special function module is an optional module which can be easily connected to the stabilization module and provides passively-stabilized, scaled, integrated and/or dynamically-sampled imagery. Each of these functions can significantly enhance FLIR operation under degraded conditions.

TABLE V-I

SPECIAL FUNCTION MODULE ADVANTAGES

- PROVIDES PASSIVE IMAGE STABILIZATION
- NO SERVO CONTROL REQUIRED
- CORRECTS FOR TRANSLATION, ROTATION, SKEW AND DILATION
- UP TO 11.2TIMES IMPROVEMENT IN SIGNAL-TO-NOISE RATIO
- UP TO 3.5TIMES IMPROVEMENT IN RESOLUTION
- FREEZE FRAME ON (PORTION OF) SCENE NOT IN CURRENT FOV

CAI

The stabilization function makes use of the offset data provided by the stabilization module to write fresh video information into the appropriate memory location, i.e., that memory location which contained the particular scene element on the previous frame. For the first frame, the information is written in normal order. Thereafter, write addresses are modified by the correlation offset results. Reading video data to the reformatter for display is always in normal order without offset. Therefore, as shown in figure 5.1, the output continues to provide the orginal scene with regard to the position of scene elements, and is merely updated in time for fresh intensity information.

Because the stabilization module requires unstabilized frame storage, offset results relate only to frame-to-frame differences. As was shown in section II, if the angular rate is constant, the offsets are constant on unstabilized imagery. Taking this one step further, with constant angular rate applied to the sensor, the correlator results will be zero. To track the position of the stabilized scene in the special function module memory relative to the sensor, this module must accumulate correlator offset results, thereby calculating the total offset since locking onto a scene. Figure 5.2 is a block diagram of this concept.

Whenever the total offset added to the current pixel and line addresses constitute a valid memory address, that address may be updated with current video. In other words, updating of stored scene information is allowed only when and where the current scene from the sensor overlaps the scene stored in memory. Whenever such updating is allowed, the current scene information is brightened a few gray levels to display the location of the stabilized scene on the unstabilized scene as shown in figure 5.3.

Should the offset tracking performed by the pipeline correlator ever break lock under the conditions described in section II, the correlator will reinitialize itself, but all reference between stabilized and unstabilized scenes will have been lost. In this case, the special function frame store merely maintains the stored scene in a true freeze frame mode, and a message indicating the loss of reference will be displayed to the operator on the control panel alphanumeric display.

PASSIVE STABILIZATION

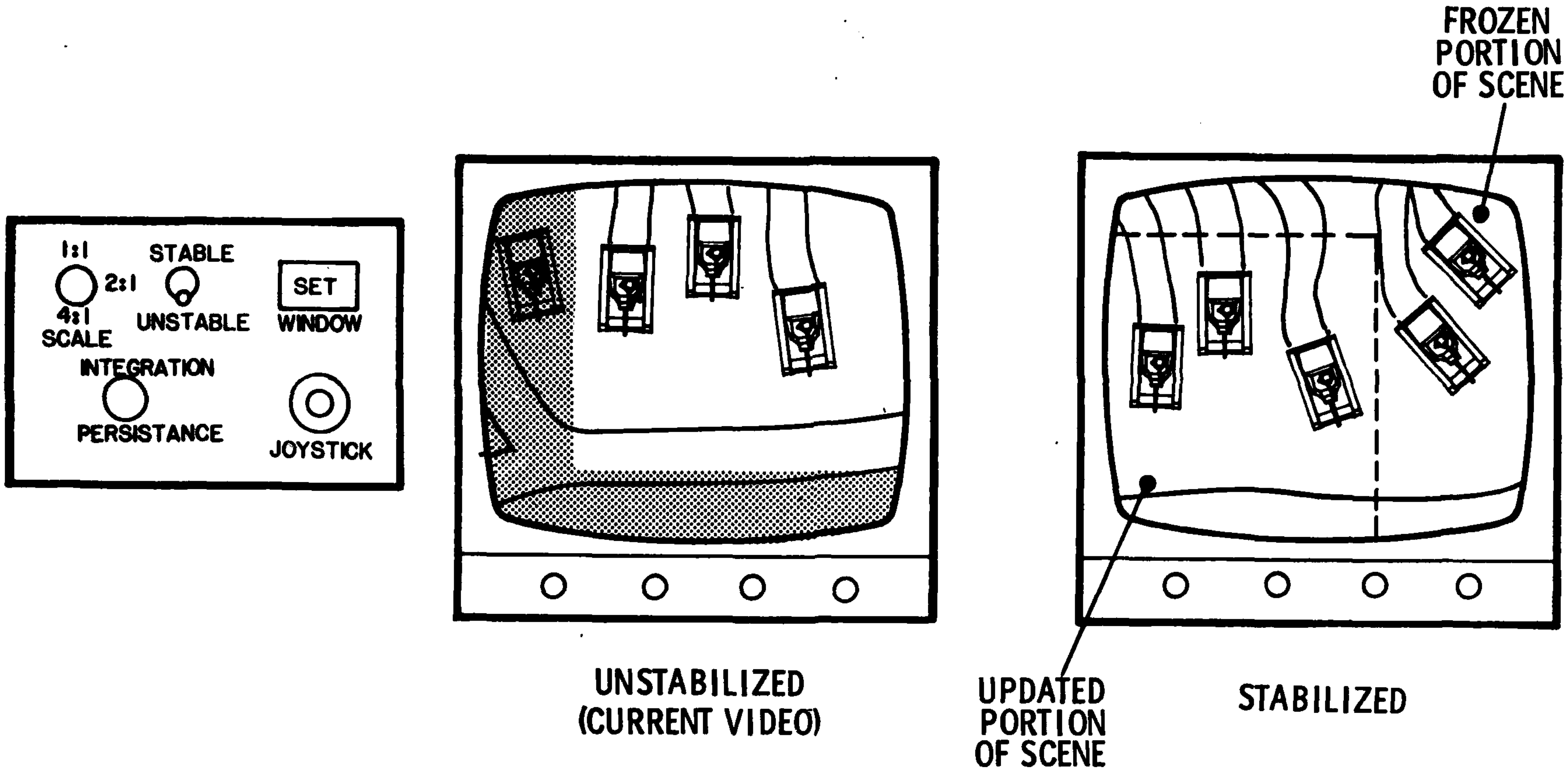


FIGURE 5.1

PASSIVE STABILIZATION

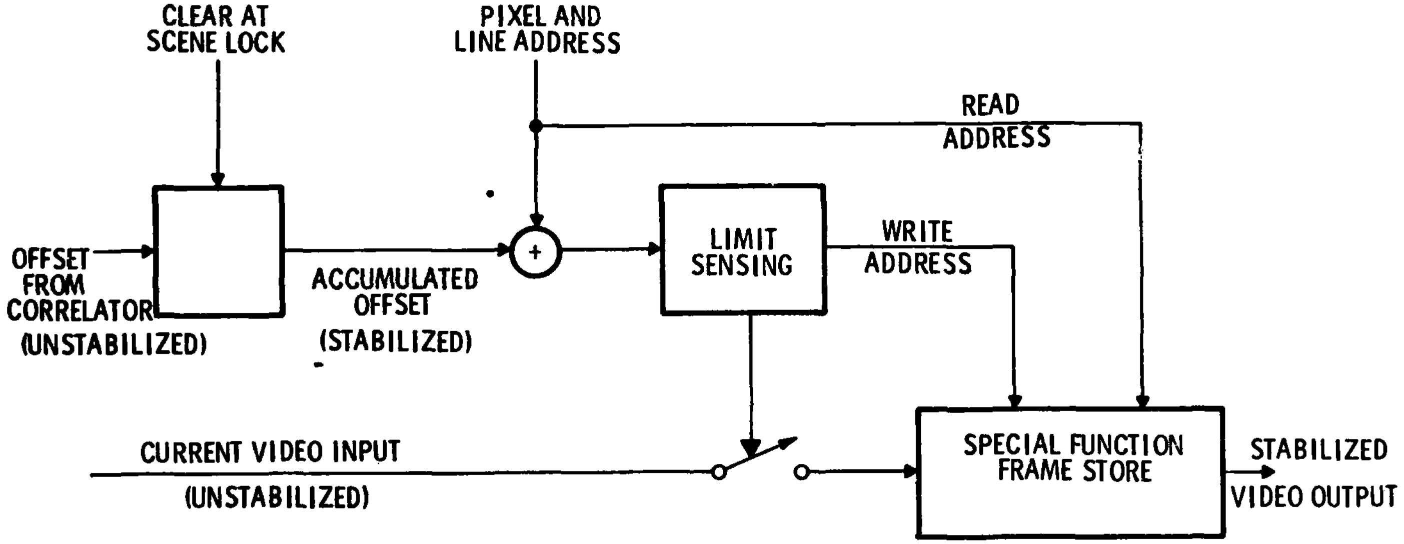

CLEAR AT SCENE LOCK
PIXEL AND LINE ADDRESS
READ ADDRESS
OFFSET FROM CORRELATOR (UNSTABILIZED)
ACCUMULATED OFFSET (STABILIZED)
+
LIMIT SENSING
WRITE ADDRESS
CURRENT VIDEO INPUT (UNSTABILIZED)
SPECIAL FUNCTION FRAME STORE
STABILIZED VIDEO OUTPUT

FIGURE 5.2

STABILIZATION WINDOW DISPLAY

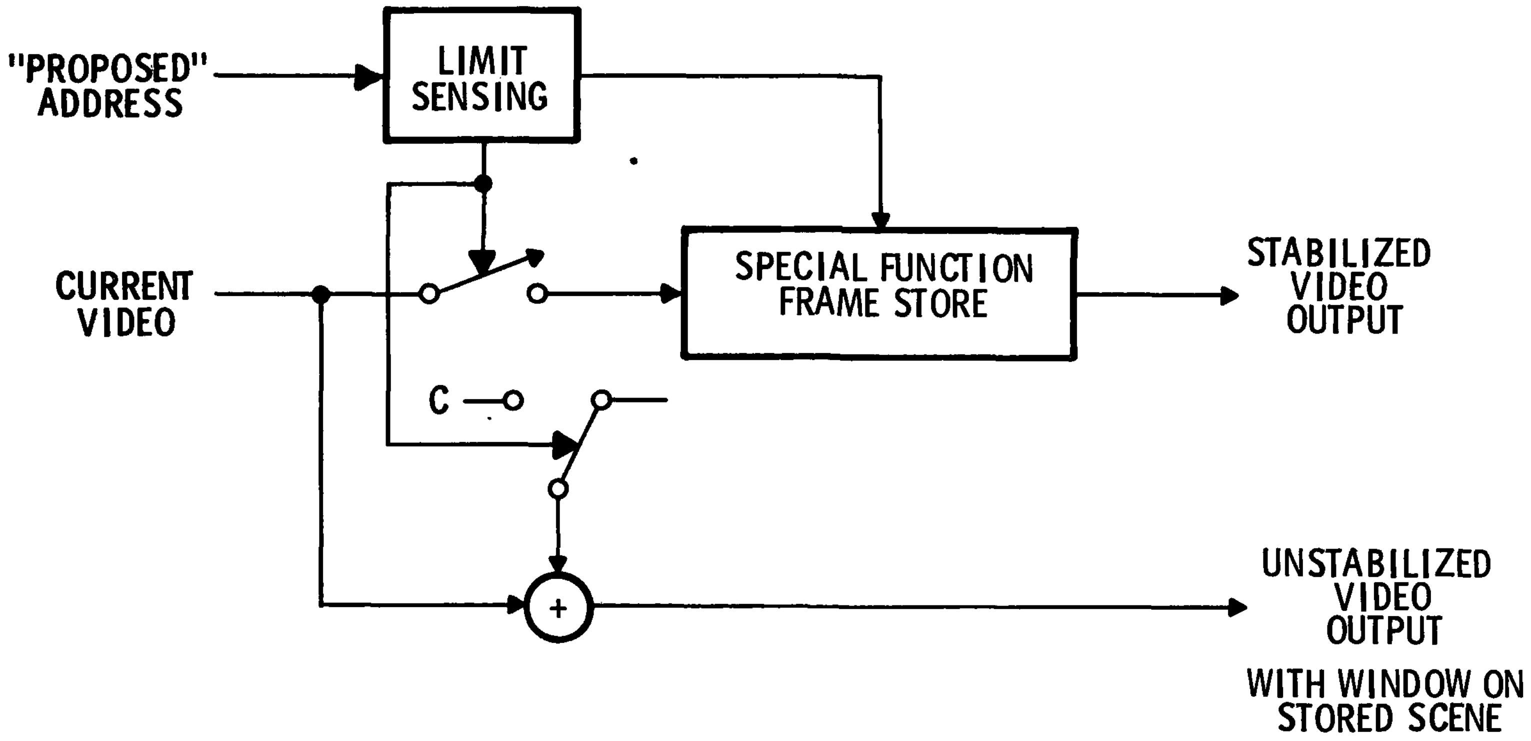

"PROPOSED" ADDRESS
LIMIT SENSING
CURRENT VIDEO
SPECIAL FUNCTION FRAME STORE
STABILIZED VIDEO OUTPUT
C
+
UNSTABILIZED VIDEO OUTPUT
WITH WINDOW ON STORED SCENE

FIGURE 5.3

Image scaling, by itself, is simply the writing of each pixel datum into a block of memory locations. An alternative method would be to re-read pixels and lines several times before proceeding to the next pixel or line. Although greater flexibility might be achieved by this approach, implementation is more difficult and it would conflict with the control required for dynamic sampling. As will be shown, dynamic sampling requires filling the memory with the scene portion of interest.

To scale, it is necessary to define what portion of the scene is of interest. Among the many methods, the easiest is to display a window on the unscaled video of appropriate size and allow the operator to move the window with a joystick or similar control. The scale factor determines the displayed window size. When the window is satisfactorily located, a control button enables the system to select that window to display an enlarged format. This is shown in figure 5.4.

Because memory chips can only store or recall one datum at a time, it is not practical to attempt writing a pixel datum to several memory locations in one chip. For this reason, the frame storage architecture is specifically designed as a 4 x 4 array of chips wherein each chip stores only every fourth pixel and every fourth line. Hence, a 4X scaling is accomplished by writing a pixel datum to the same address in all 16 memories simultaneously, as shown in figure 5.5. Similarly, 2X scaling is accomplished by writing to only four memories simultaneously. This architecture does limit scaling factors to 1, 2 or 4x.

A similar constraint is that the precision of stabilization is still limited to ±1/2 real pixel despite the scaling of pixels. If fractional pixel offsets were allowed, there could be up to four different addresses to the memory and would require independent address lines, or the writing would have to be multiplexed. There is insufficient time for multiplexed writing and independent address lines would be detrimental for size and complexity considerations. Greater precision in stabilization is, nevertheless, accomplished with dynamic sampling.

SCALING

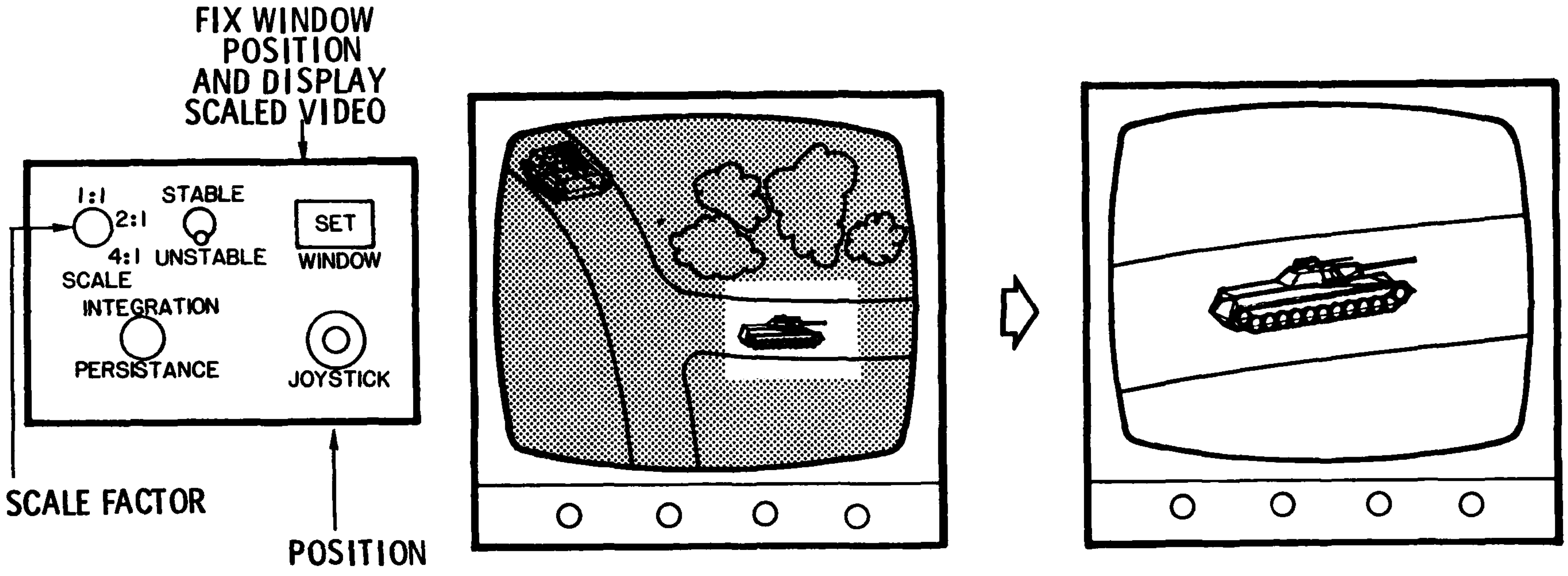


FIGURE 5.4

SPECIAL FUNCTION MODULE MEMORY ARCHITECTURE

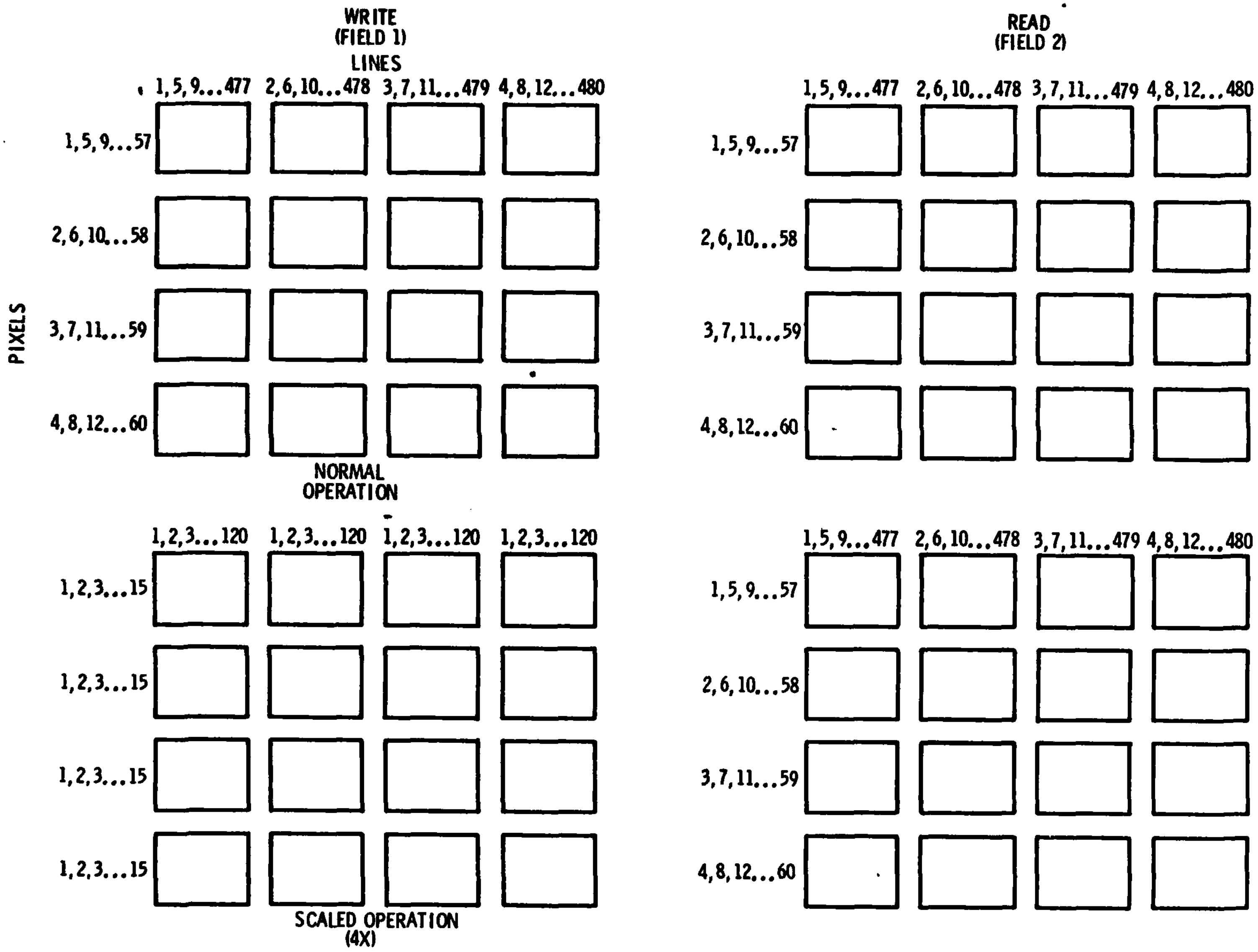


FIGURE 5.5

The reading of scaled imagery is performed in normal fashion just as though the scaled portion of the scene were the entire scene.

It is well known that the sampling of signal frequencies whose periods are shorter than two sample spacings will result in aliasing - the folding of high-frequency information to lower frequency output patterns. In images, such patterns are known as Moire patterns. The limiting frequency is known as the Nyquist frequency. *Because of aliasing, it is not possible to reproduce an original* signal whose frequency is greater than the Nyquist value from its samples. This does not mean that frequencies at or below the Nyquist limit can in fact be reproduced, as will be shown.

The sample spacing for solid-state detection is fixed by the physical size of the elements. It may then appear that the resolution is essentially fixed by the Nyquist limit. This is not the case. In fact, frequencies up to 3.5X the Nyquist limit may be resolved without aliasing merely by oversampling. That is, shifting the scene by fractions of a pixel and constructing an image from the various phases allows improved resolution. The construction of one image from the independent phase versions may be done in one of two ways:

- By considering each sample to be a full pixel in width but averaging overlapping fractions of a pixel according to phase
- By considering each pixel to be representative of a fraction of a pixel and simply interleaving the results according to phase

Figure 5.6 illustrates each method. Note that even at Nyquist frequency, under static sampling, there may be no information at a particular phase (ϕ = 1/2), or there may be a slight phase shift and loss of contrast (ϕ = 1/4 and 3/4). The two results show that there is improved contrast with the interleaving method, but no decision should be made with this single case.

Figure 5.7 compares the two methods for several cases, sinusoid and square waves at 2.67X the Nyquist frequency, a single 1/4-pixel wide bar and a three-bar target at 3X the Nyquist frequency. Interleaving generally shows better contrast,

DYNAMIC SAMPLING METHODS

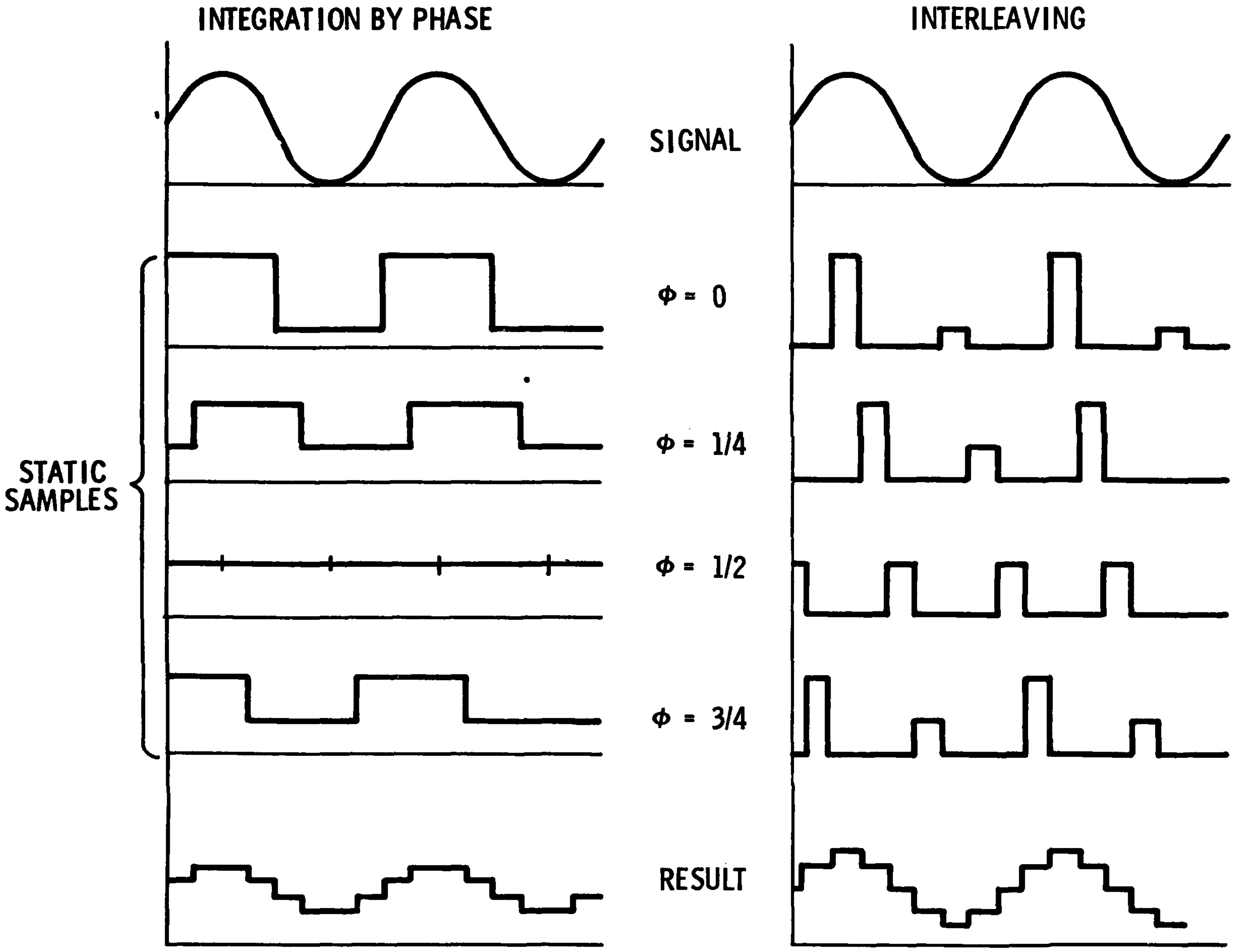


FIGURE 5.6

DYNAMIC SAMPLING

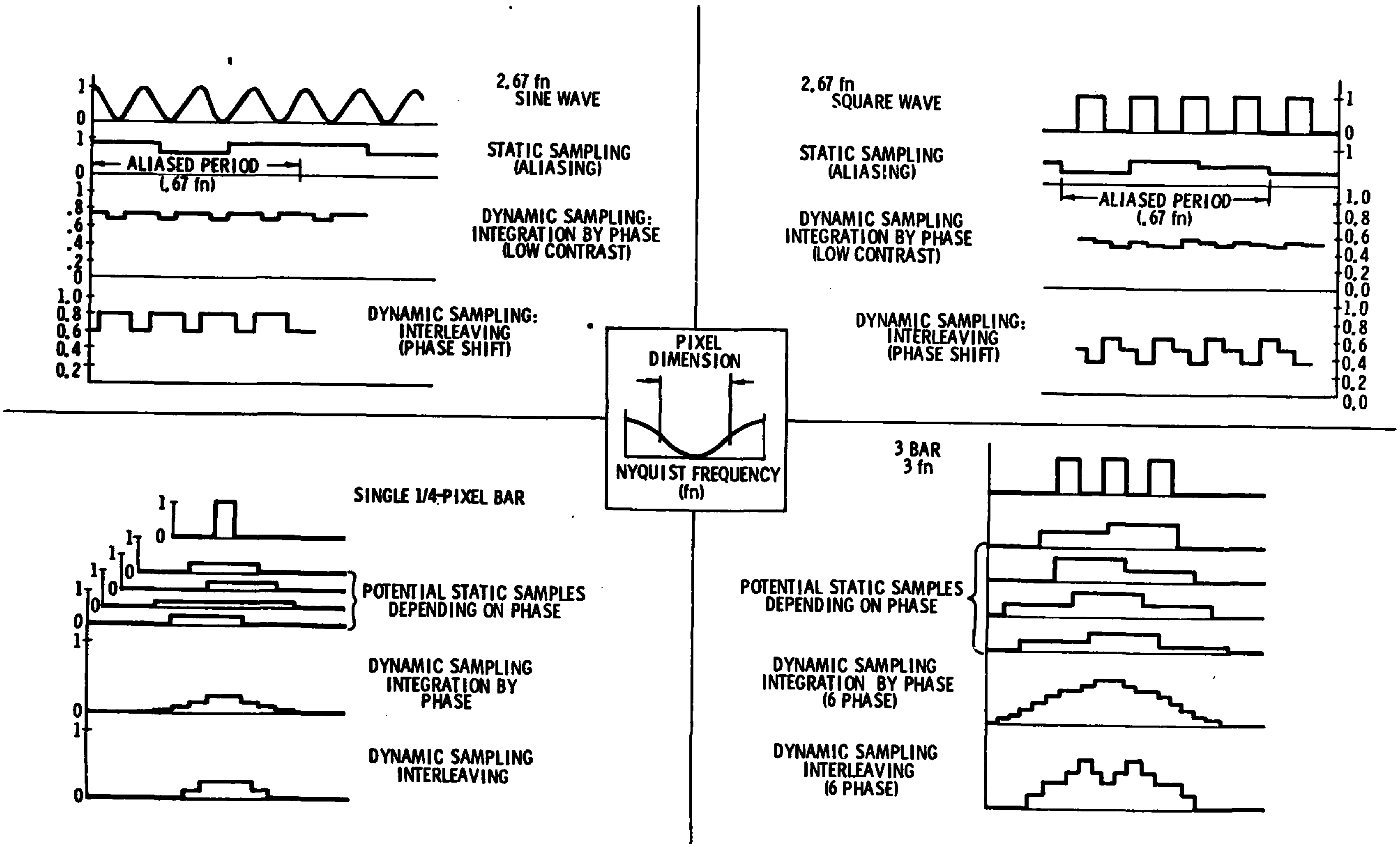


FIGURE 5.7

but often with a phase shift or contrast reversal. The last example of a three-bar target clearly shows the problem - spurious resolution. Signal power is shifted such that two instead of three bars appear. Integration by phase does show correct phase relationships, but the contrast is generally so poor as to be useless. In fact, the three-bar target illustrates that such loss of contrast can even mask the fact that there are three bars, leaving only a general hump. It is clear that dynamic sampling is not a perfect solution, but it is a definite improvement. Consider the first two periodic signals. Static samples clearly show aliasing and any sense of a high-frequency signal has been lost, while the dynamically-sampled version clearly resolves the pattern even if it is spurious resolution in the interleaving case. The method of interleaving appears to be a preferred approach.

Fortunately, interleaving is the most simply-implemented method of dynamic sampling. Each pixel and line address plus offset is merely scaled by some scaling factor when writing data into the frame store. Reading is performed in normal fashion. There are several points to be made regarding this method. First, there are no restrictions on scale factors since only one writing of data is attempted for each pixel. Second, stabilization-mode operation is required and it is inherently tuned to the scale factor in terms of precision. This is in contrast to simple scaling. Third, there may be gaps in the information stored if particular phases of offset do not occur.

A fourth point must also be considered. There must be some image motion for dynamic sampling to work, that is, if there is active stabilization it must be imperfect. The motion may easily be induced and controlled if there is active stabilization. Without active stabilization, however, the system will have to rely on the random motions of the sensor induced by the environment. The gaps in information stored may be covered somewhat by storing the first frame in a standard scaled fashion and thereafter updating those phases which occur. In this manner, there will be some reasonable data in all memory locations, although it does impose the scaling limitations previously mentioned.

Integration of imagery is the stωring of a weighted average of the current frame and the stored frame. The relative weights must add to unity while the ratio of weights is a measure of persistance. To simplify hardware implementation, only eight possible weighting ratios have been allowed: from 1:0 (no integration) to 1:127 (maximum integration). Since the video is in 8-bit digital format, a ratio of more than 1:127 would be equivalent to a freeze frame since curent video would contribute less than a least-significant bit.

Integration is defined by the equation:

$$S' = 2^{-n} C + (1 - 2^{-n}) S$$

where:

S' = new intensity

C = current pixel intensity

S = old stored intensity

N = persistance measure (from 0 to 7)

Figure 5.8 shows a block diagram of this function. Note that the information from any one frame is never dropped from the calculation, it merely decays. Nevertheless, the improvement in signal-to-noise ratio is equivalent to a straight average of 2^n frames. The improvement in signal to noise ratio is then $\sqrt{2^n}$ as shown in table V-2.

Without stabilized (either actively or passively) imagery, integration merely results in smear. With stabilization, integration can provide considerably improved performance in low signal-to-noise ratio conditions. Integration may be used with simple scaling or dynamic sampling as desired.

CONTROL PANEL

The control panel for the special function module is a plug-in addition to the basic control panel previously described. It includes switches for scaling, (1:1, 2:1, or 4:1) stabilization (unstabilized, stabilized, dynamically sampled), and an eight-position switch for integration persistance on a window control area. For window position control, a four-button array is used as a joystick. It is both less expensive and more rugged than a joystick and is easily operated with a single

FRAME INTEGRATION

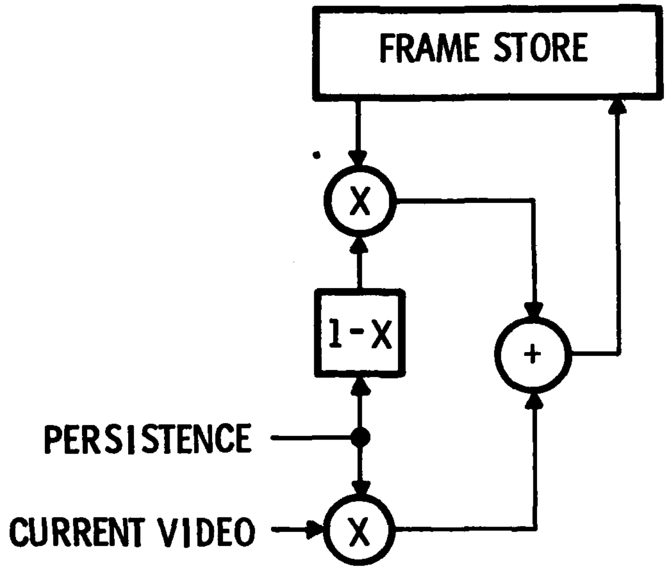


FIGURE 5.8

TABLE V-2

FRAME INTEGRATION

PERSISTANCE SETTING	INTEGRATION RATIO	SNR IMPROVEMENT
0	1:0	1.0
1	1:1	1.4
2	1:3	2.0
3	1:7	2.8
4	1:15	4.0
5	1:31	5.6
6	1:63	8.0
7	1:127	11.2

finger or thumb. An additional pushbutton control releases positional lock. Upon release, the positioning control becomes active and the window may be moved to the desired location. When the positioning control buttons are released, the window position is fixed.

These controls allow complete control of the special function module. The windowing position control may also be used for displaying the velocity or range of a particular target with the appropriate system configuration. In this mode, the window positioning control is active. If a particular window is locked in the special function module, it will be unaffected. The window merely defines which output of the MTI/range card will be latched and displayed on the alphanumeric display of the basic control panel. This mode is made operational by a control button on the keyboard of the basic control panel. If constants need to be input to derive absolute velocity in range, the control panel will so inform the operator.

SECTION VI
DESIGN CONSIDERATIONS

MODULARITY AND VERSATILITY

The basic premise of the common module program is the ability of the user to select the particular components which meet a set of needs, and to easily combine them into a system.

These modules attempt to preserve such versatility. Each block operation is a mother board configuration. Frame store extensions from 60 to 120 to 180 pixel arrays are handled merely by adding the necessary number of memory cards to the mother board. The whole frame store could be used as an independent stand-alone frame store if desired for some purpose. Similarly, each optional configuration described involves merely connecting the appropriate mother boards or inserting them in their slots. For example, if only a relative motion of range map capability were desired, the basic remote control panel module may or may not be needed, depending on the application. The basic stabilization will work perfectly well without this panel, but when it is needed, no system modification is required. Each mother board may be housed in one or separate card racks.

The special function module, providing passive stabilization, scaling, integration and dynamic sampling, is not required in general for stabilization applications. Such functions should only be needed in fairly hostile environments. Since it is expected that these functions will be used less than those provided with the stabilization module, it is provided as an optional plug-in addition. While it seems an unlikely situation, this module could be configured as a stand-alone module given some other stabilization signal with which to stabilize imagery and perform dynamic sampling or integration.

Another form of versatility has been achieved by including multiple functions or outputs wherever possible. In general, there has been little or no additional complexity involved.

By a modular approach to card layout on a mother board module, the system is designed for easy growth. Each card performs a specific function and is in a sense a submodule. The addition, deletion or replacement of functions will thus have minimal impact on the entire system.

The control panel has also been designed for versatility and growth. Being based on a single chip microprocessor control and having an alphanumeric readout, a wide variety of control functions, error messages and operational cues can be implemented with virtually no effect on system interface. Commands and controls between the panel and the operational hardware are transmitted and received via a three-wire bus in a serial format. Interface with 1553B bus architecture has also been considered and can be readily implemented.

As can be seen, the concept of modularity has been a prime concern in the design of the modules described.

SPEED, SIZE AND COST

In the design of a pipeline processor, speed of operation becomes a serious concern. Each operation must be completed within 1 pixel time. After being multiplexed to a serial format, a pixel time may be on the order of 193 ns when the format is a field of 180 pixels by 480 lines.

The operations to be performed are addressing, storing and retrieving of information from memory, arithmetic operations and various forms of encoding and decoding of control conditions. For current memories, access times of 150 ns are considered fast but not unusual in today's devices. For arithmetic operations, additions are easily handled in this time, multiplications as well as multiply/accumulate operations are fast enough when performed by special devices on the market, but standard divisions take too long, approximately 2.2 µ s, for the fastest devices. Logic arrays to handle control functions are sufficiently fast to be used.

Size and cost often suffer in the quest for speed. For example, the special multipliers are fairly large 40-pin devices and the multiply/accumulate packages are 48 to 60-pin devices, depending on the number of bits. If arithmetic division were performed, 12 dividers would have to be multiplexed resulting in a large size and cost. In general, optimal speed, size and cost tradeoffs must be carefully balanced.

The approach taken in this study was to begin by reducing size and component count as far as possible within the constraint of operating speed and with cost as a consideration; then, with this near minimal size, to perform card layouts, and finally, on those cards which permit expansion, to relax size constraints if cost savings would result. Throughout the design process, the availability and cost of militarized parts has also been a prime concern to minimize redesign for production.

The final product is an initial design which is as small and inexpensive as feasible, yet will operate at the fastest rate expected to be required.

Table VI-1 summarizes the considerations discussed in this section.

TABLE VI-I

DESIGN CONSIDERATIONS

- MODULARITY AND VERSATILITY
    - MOTHER BOARD PLUG-IN AND OPERATE EXPANSION
    - STAND ALONE OPERATION OF BLOCKS
    - MULTIPLE FUNCTIONS/OUTPUTS
    - DESIGN FOR GROWTH
- SPEED, SIZE AND COST
    - SPEED IS BASIC REQUIREMENT
    - OPTIMIZE FOR SIZE WITHIN REASONABLE COST LIMITS
    - CARD LAYOUTS - DIVISION OF FUNCTIONS
    - RELAX SIZE WHERE POSSIBLE TO MAINTAIN CARD COUNT AND REDUCE COST
    - AVAILABILITY AND COST OF MILITARIZED COMPONENTS HAS ALSO BEEN CONSIDERED

SECTION VII
BRASSBOARD DESIGN AND OPERATION

For the brassboard design, a single enclosure, individual card approach has been taken. However, card designs have accounted for eventual transition to the mother board concept.

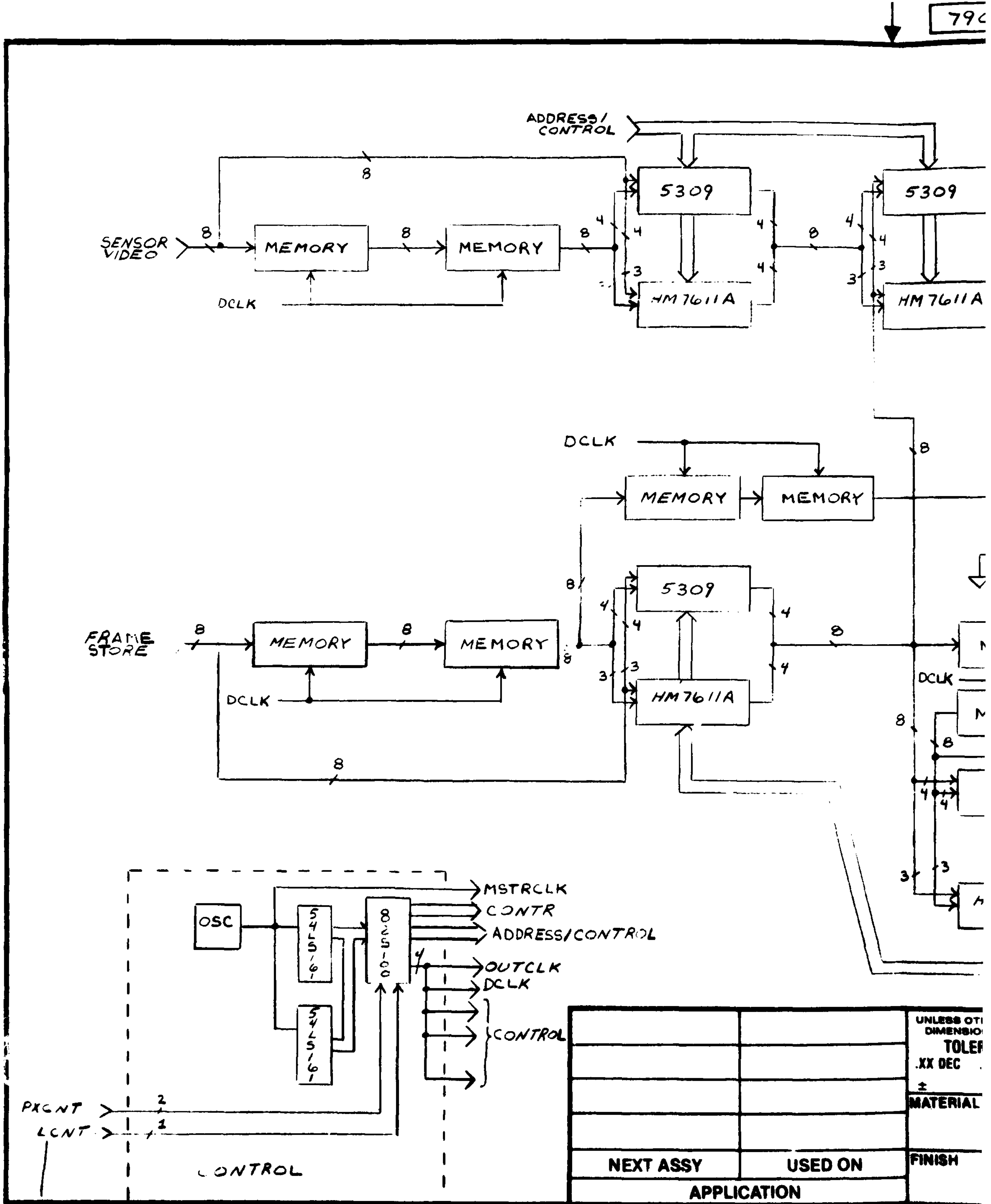
PIPELINE CORRELATOR CARDS

The pipeline correlation processors, shown in figures 7.1 and 7.2, have fairly straightforward operation. The required inputs consist of two streams of video, a pixel clock and a line clock. Data manipulation is handled in random access memory (RAM) and operations are generally handled in a look-up table fashion with programmable read only memory (PROM). These methods are not only the most efficient, but offer considerable flexibility.

Summations over a window illustrate considerable evolution from the block diagram given in section II. For the cross scan direction, partial results are accumulated for one line and output at the end of a line for final processing.

Summations or accumulations may be performed with two 4-bit adders and a latch to loop the sum back for the next result. Alternatively, a multiply/accumulator may be used. Even though the multiplication function may be wasted, these devices are more efficient in terms of space requirements. Naturally, they are also more expensive. If, however, the first method results in the circuit growing too large to be fit on one card, the cost savings may be lost as shown in figure 7.3.

The scan direction offset requires more complex processing, since running summations over windows covering several lines and being output at a pixel rate during a line are needed. The shift register architecture suggested in the block diagrams in section II is not a good hardware approach. Shift registers are neither available in arbitrary lengths nor flexible. RAM's are considerably better choices, even if they may provide for more memory than required. Thus, instead of

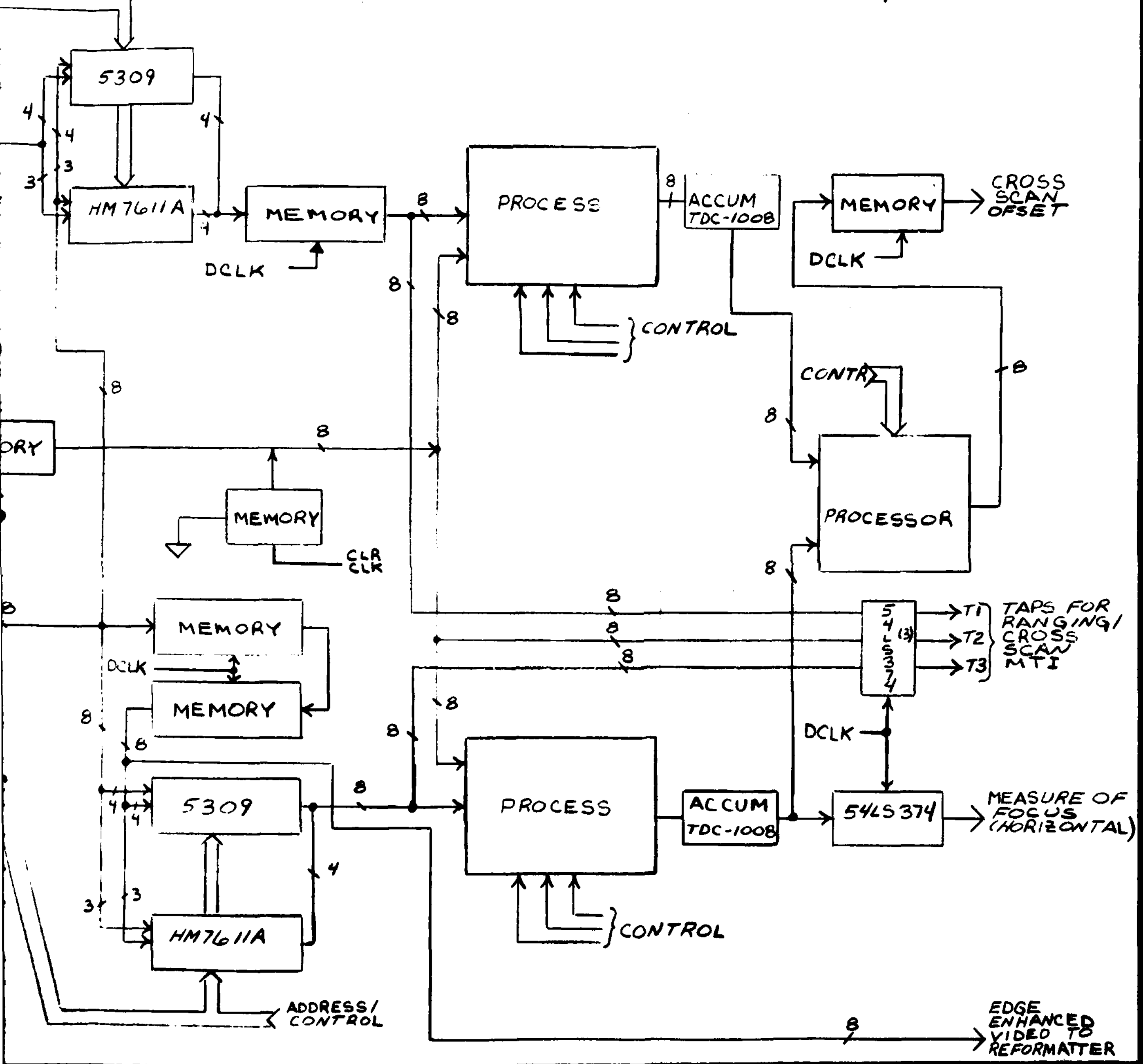


NEXT ASSY	USED ON
APPLICATION	

UNLESS OT
DIMENSIO
TOLER
.XX DEC
±
MATERIAL
FINISH

7901-2B

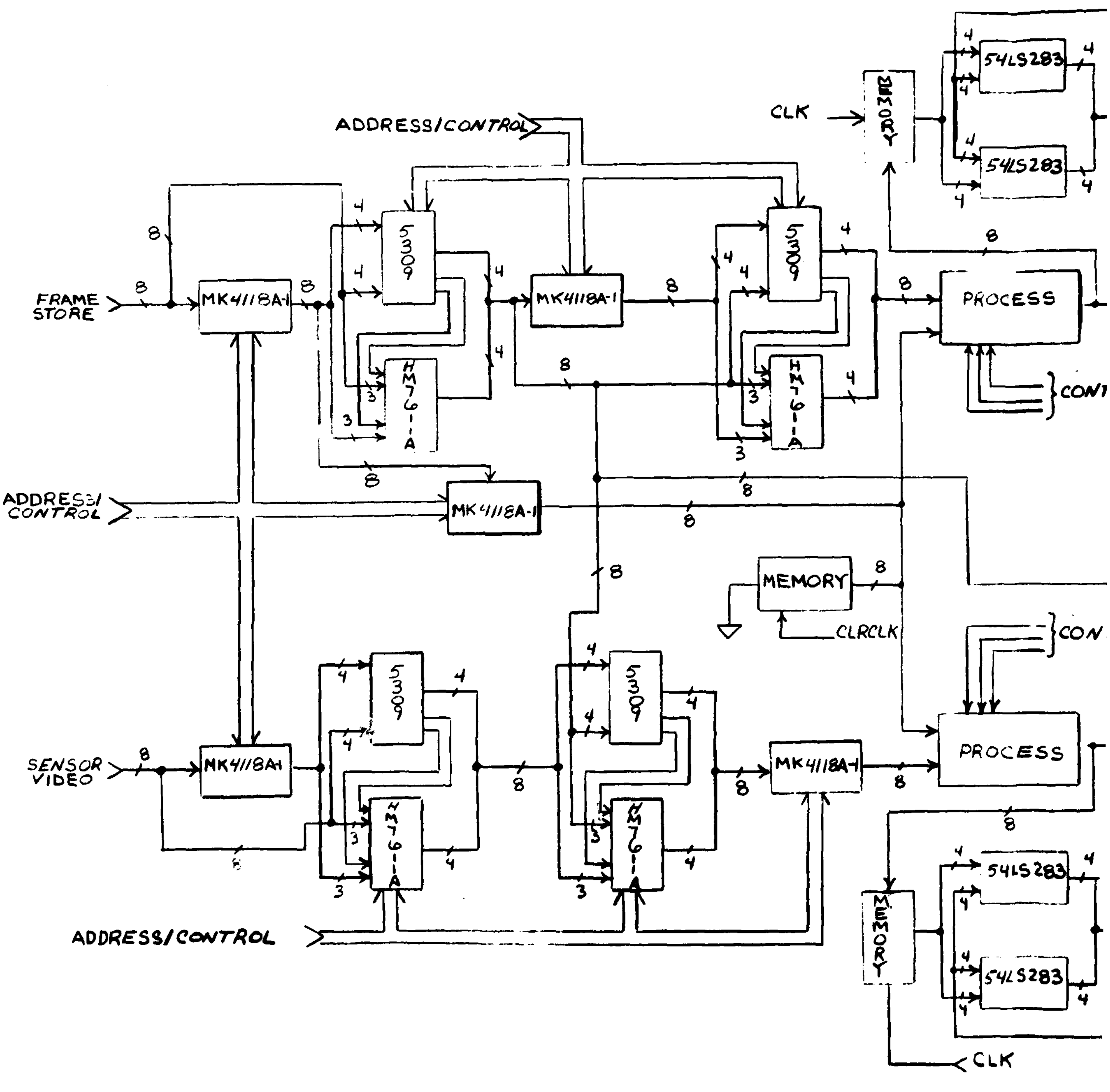
REVISIONS			
REV	DESCRIPTION	DATE	APPROVED



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: .XX DEC .XXX DEC ANGLE ± ± ±	CONTRACT NO. DAAK70-81-C-0255	CAI a Division of RECON/OPTICAL, Inc. BARRINGTON, ILLINOIS 60010
MATERIAL	DRAWN T. GOODRICH 3/29/82	PIPELINE CORRELATOR HORIZONTAL CROSS-SCAN OFFSET
	CHECKED	
	APPROVED	SIZE B / FSCM NO. 11871 / DWG NO. 7901-2B
ON / FINISH	APPROVED	
	RECORDED	SCALE / WT ACT CALC LBS / SHEET 1 OF 1

FIGURE 7.1

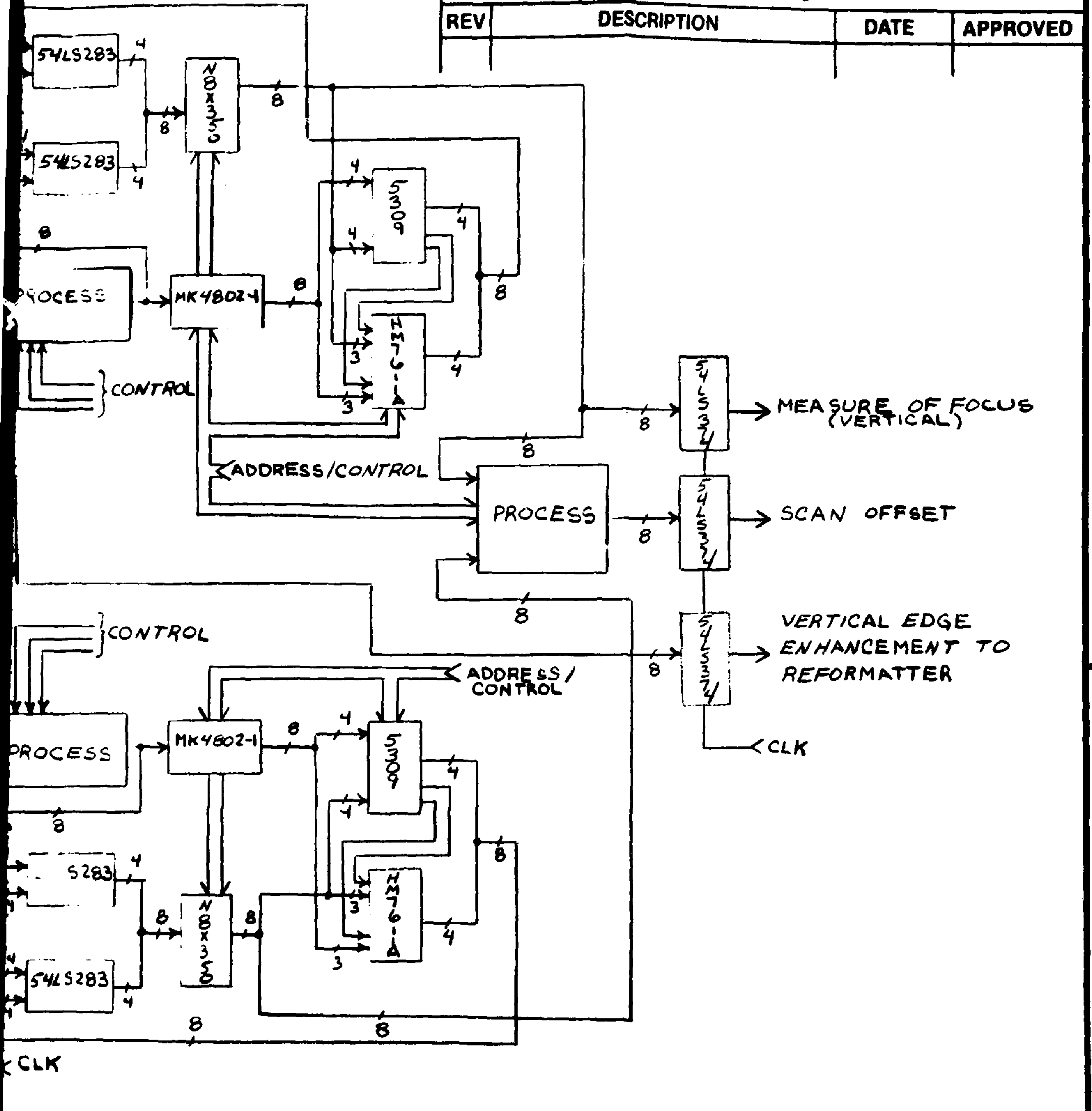
79



		UNLESS DIMENS
		TOL .XX DEC ±
		MATERI
NEXT ASSY	USED ON	FINISH
APPLICATION		

7901-2D

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES TOLERANCES ARE: XX DEC ± XXX DEC ± ANGLE ±	CONTRACT NO. DAAK70-81-C-0255	CAI a Division of RECON/OPTICAL, Inc. BARRINGTON ILLINOIS 60010	
MATERIAL	DRAWN T. GOODRICH 3/29/82	PIPELINE CORRELATOR VERTICAL SCAN OFFSET	
	CHECKED		
	APPROVED	SIZE B	FSCM NO. 11871 DWG NO. 7901-2D
FINISH 2	APPROVED		
	RECORDED	SCALE WT ACT CALC LBS	SHEET 1 OF 1

FIGURE 7.2

SUMMATION

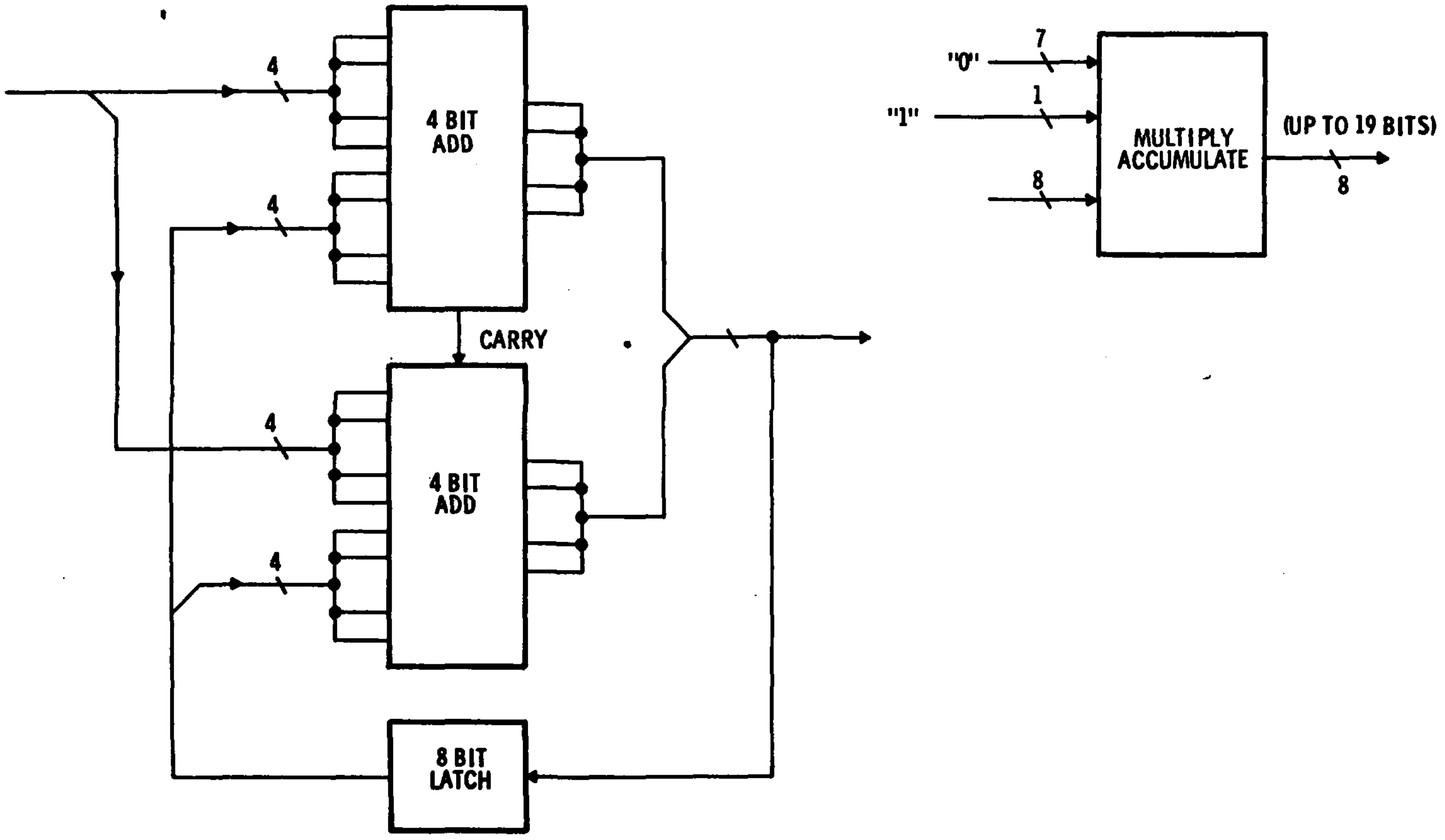
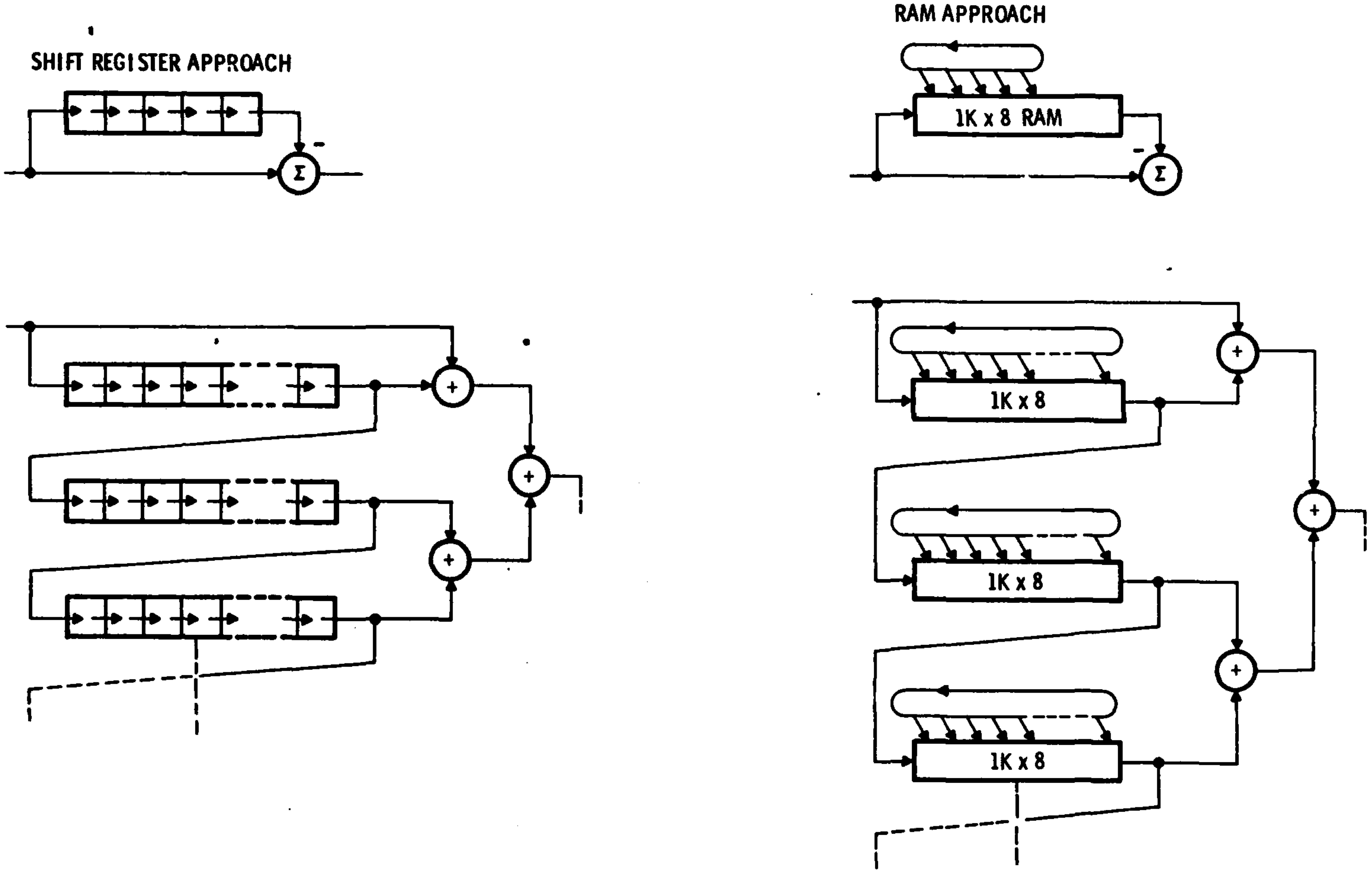


FIGURE 7.3

WINDOW SUMMATIONS



SHIFT REGISTER APPROACH

RAM APPROACH

FIGURE 7:4

RUNNING SUMMATION ACTUAL IMPLEMENTATION

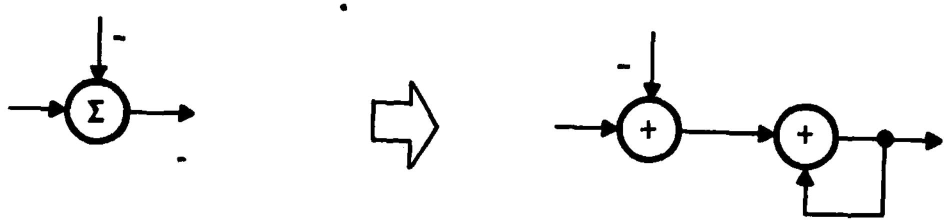


FIGURE 7.5

shifting data through shift registers, the data is stored in one location and the address pointer loops through locations as shown in figure 7.4. The top diagram showing a running sum of several pixels indicates a subtraction and addition to the accumulation. This is clearly not feasible. The proper approach is a subtraction followed by accumulation, as shown in figure 7.5.

Subtraction itself is not performed as easily as indicated thus far. A typical approach might be to perform a 2's complement on the subtrahend and then add by standard methods. Not only does the complement operation require an addition with the number 1, but sign bits necessitate the use of more components and the entire design becomes quite involved. A somewhat cleaner approach which handles positive and negative numbers in 2's complement form is shown in figure 7.6. Curiously, it is the minuend rather than the subtrahend which is inverted. The final result is then inverted (unless the result is to be the minuend for a second subtraction), to arrive at the correct answer in 2's complement notation. As illustrated, full resulution is achieved with a simple logical "OR" gate. Less 1-bit resolution consists of dropping an LSB and extending the sign bit into the MSB pins on the adders. Also shown in figure 7.6 is the simplest approach, a PROM lookup table.

Finally, the summing over block windows, shown in figure 7.4, still shows an adding tree to arrive at a final sum. This is cumbersome at best. A preferred approach is more like the first accumulation shown in figure 7.3, except that the latch becomes a RAM at least 1-line long to store summations by pixel. The adder tree and the accumulation approaches are compared in figure 7.7. The tree approach requires n-1 lines of memory, but a separate device for each line, while the accumulation techniqu require n+1 lines, but only two devices are needed.

FRAME STORE

The design of the frame store portion of the stabilization module is far less involved than for the pipeline correlator. This is shown in figures 7.8 and 7.9. Modularity demands that the memory plane be segmented into 60 pixel groups, while commonality dictates that this architecture be identical to that used for the

2'S COMPLEMENT SUBTRACTION

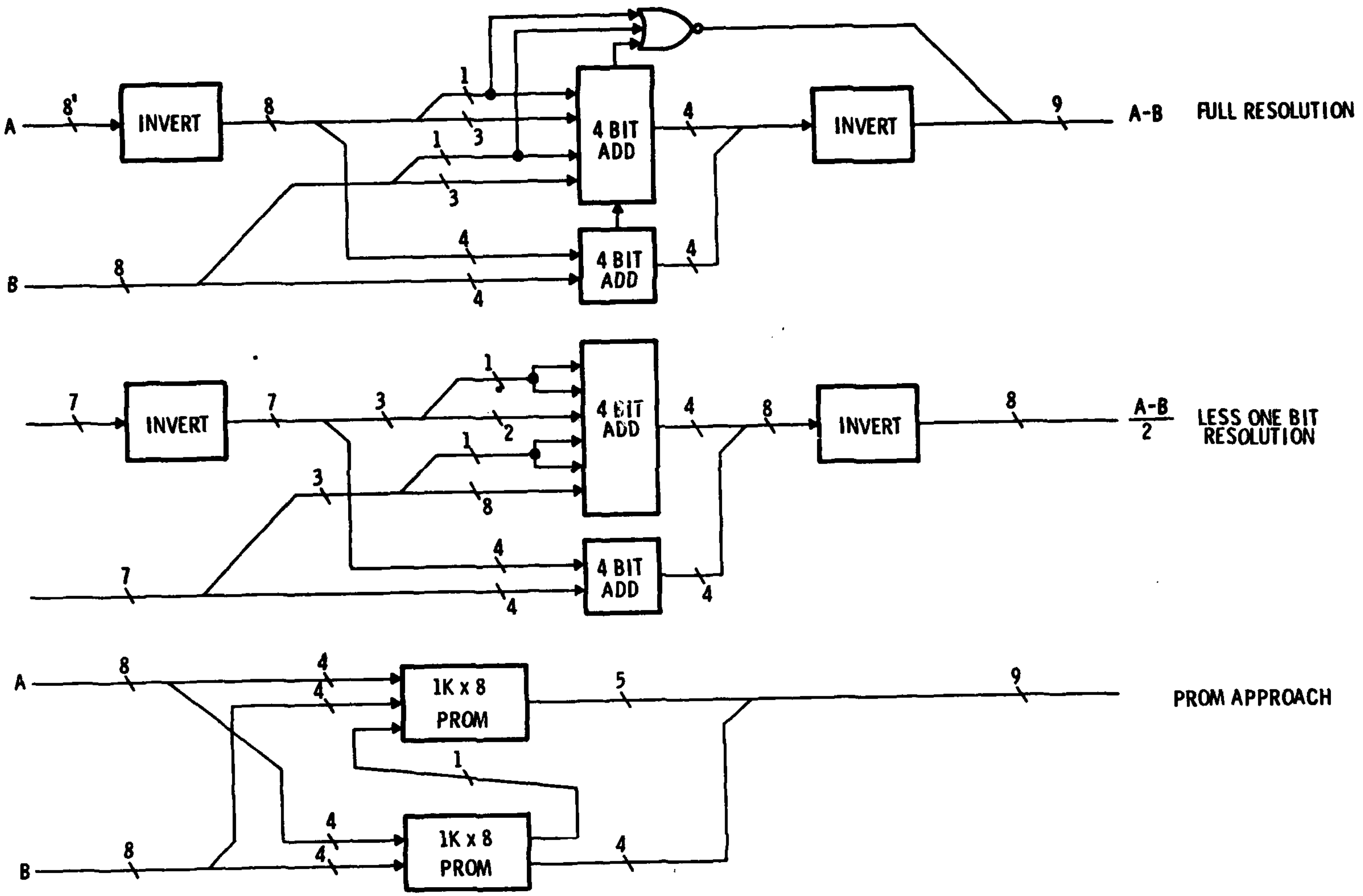
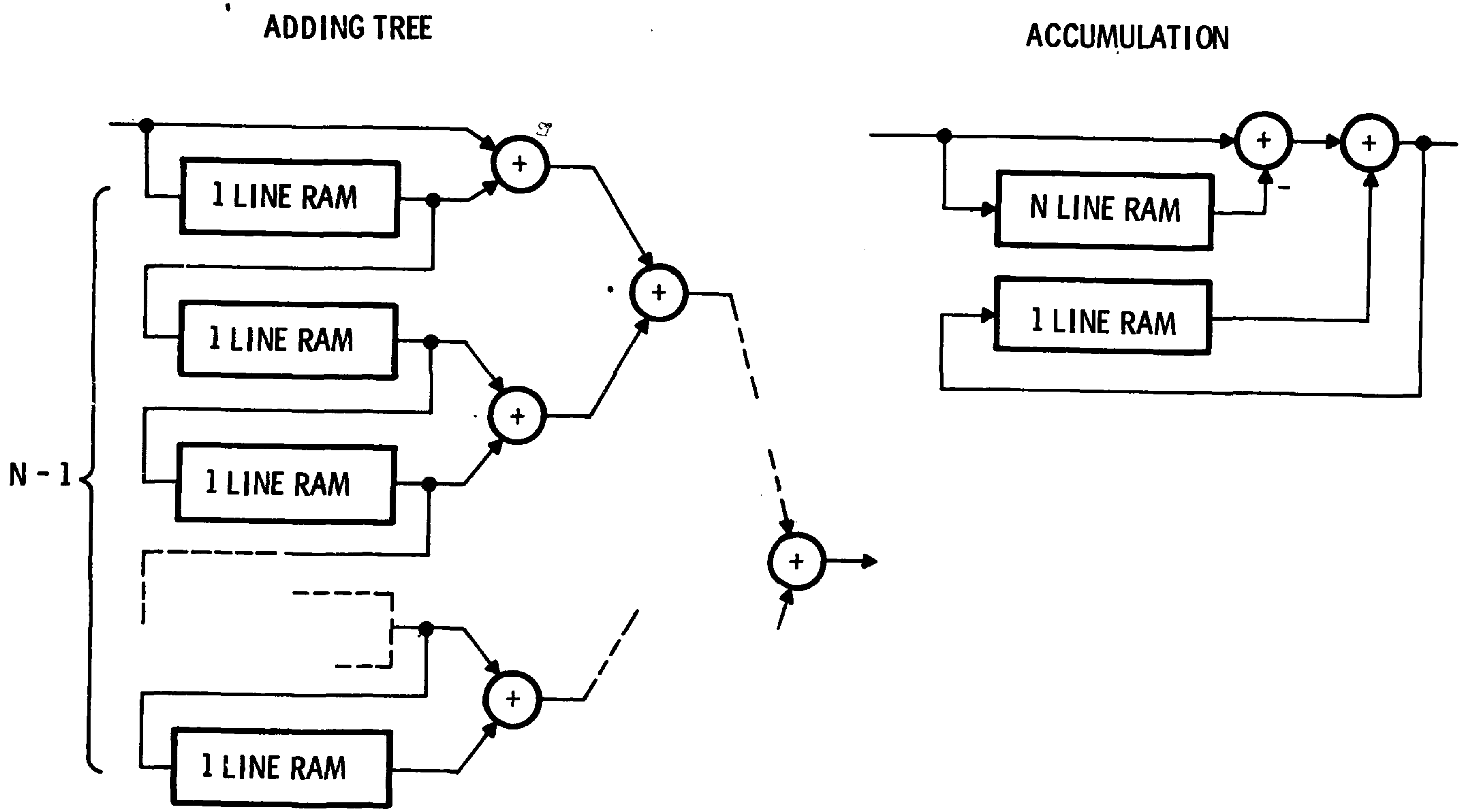


FIGURE 7.6

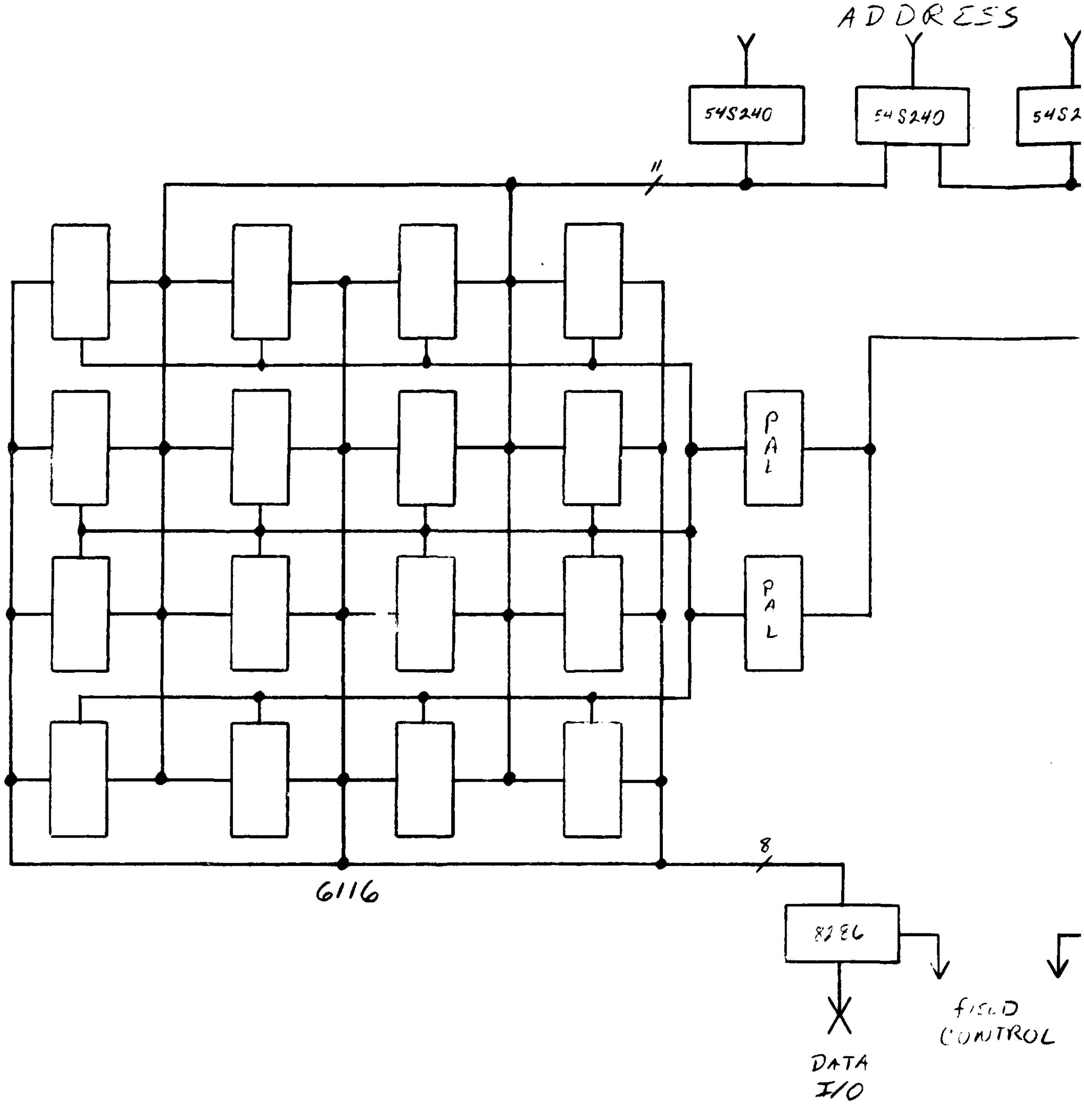
BLOCK WINDOW SUMMATION APPROACHES



ADDING TREE

ACCUMULATION

FIGURE 7.7

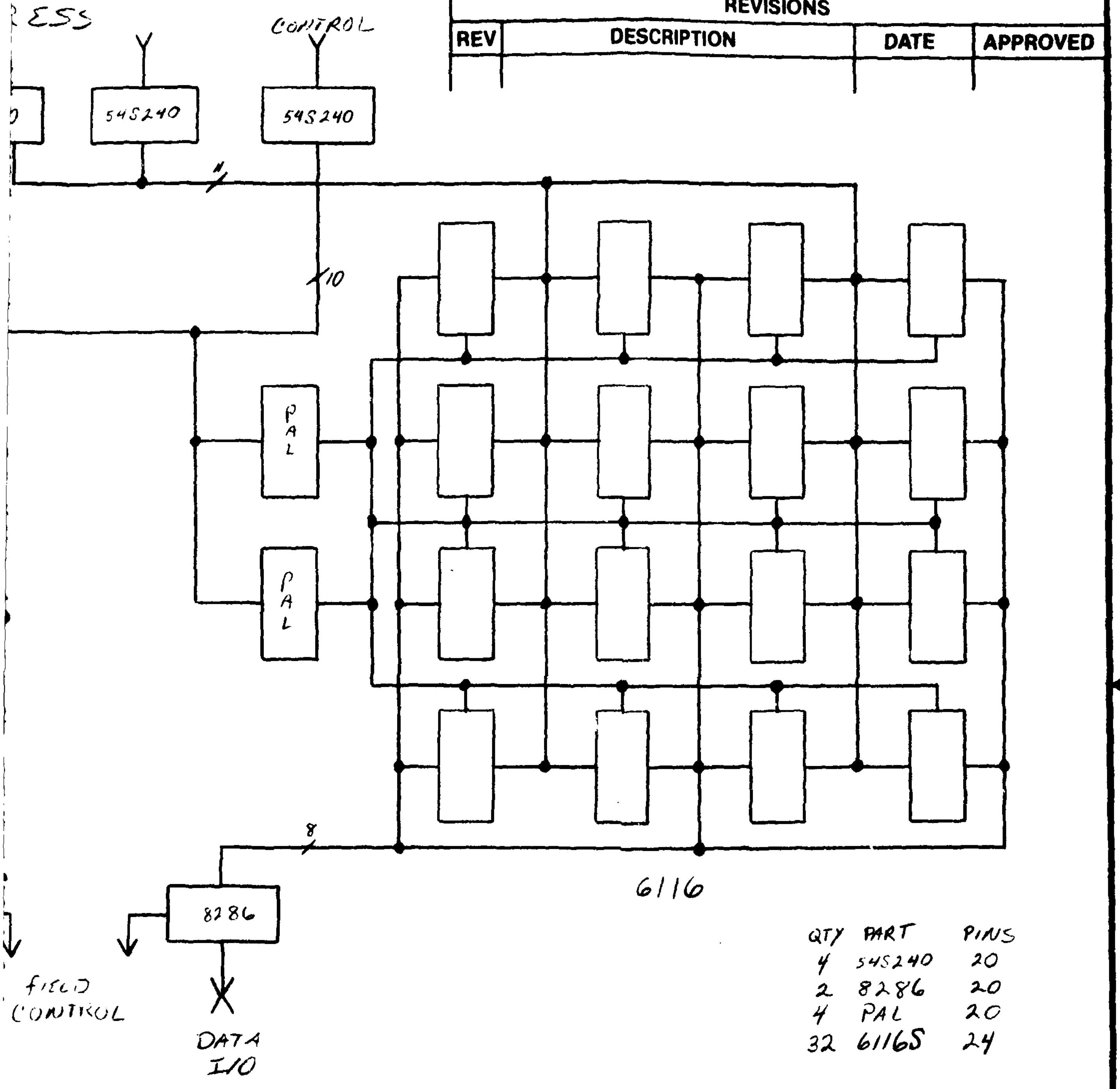


NEXT ASSY	USED ON
APPLICATION	

UNLESS
DIMEN
TO
.XX DEC
±
MATERI
FINISH

1

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED



QTY	PART	PINS
4	54S240	20
2	8286	20
4	PAL	20
32	6116S	24

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES ARE:
.XX DEC ± .XXX DEC ± ANGLE ±
MATERIAL
FINISH

CONTRACT NO. DAAK70-81-C-0255
DRAWN C Schiefer 29 Mar 82
CHECKED
APPROVED
APPROVED
RECORDED

CAI a Division of RECON/OPTICAL, Inc.
BARRINGTON ILLINOIS 60010

FRAME STORE MEMORY

SIZE B | FSCM NO. 11871 | DWG NO. 7901 - 3A
SCALE | WT ACT CALC LBS | SHEET

FIGURE 7.8

AD-A119 869 CAI BARRINGTON IL F/G 17/5

FLIR STABILIZATION STUDY.(U)

APR 82 D DEFOE, T GOODRICH, J MARTIN DAAK70-81-C-0255

UNCLASSIFIED CAI-7901-A002 NL

2 of 2

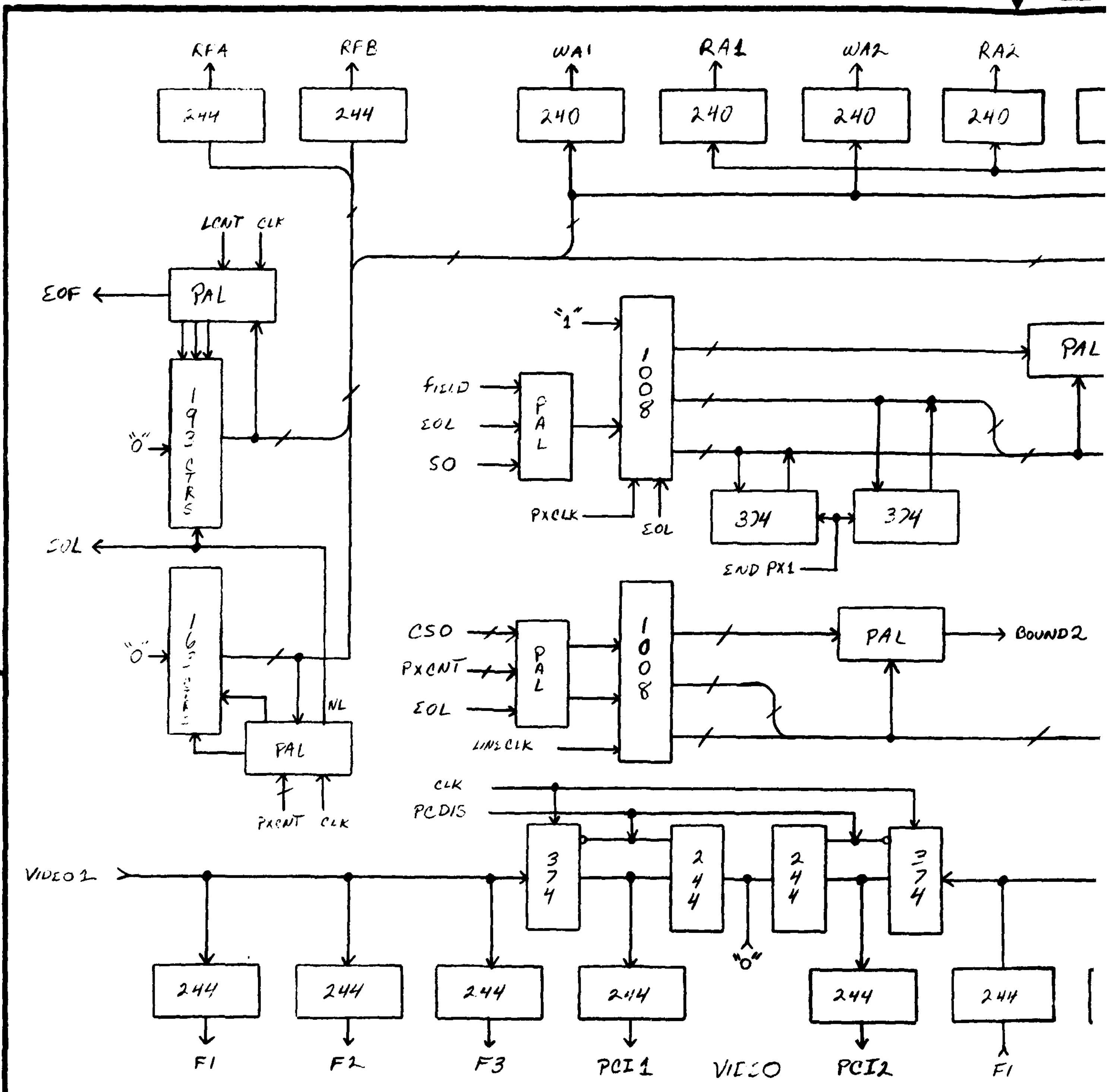
AD A119 869

END

DATE FILMED

11 82

DTIC

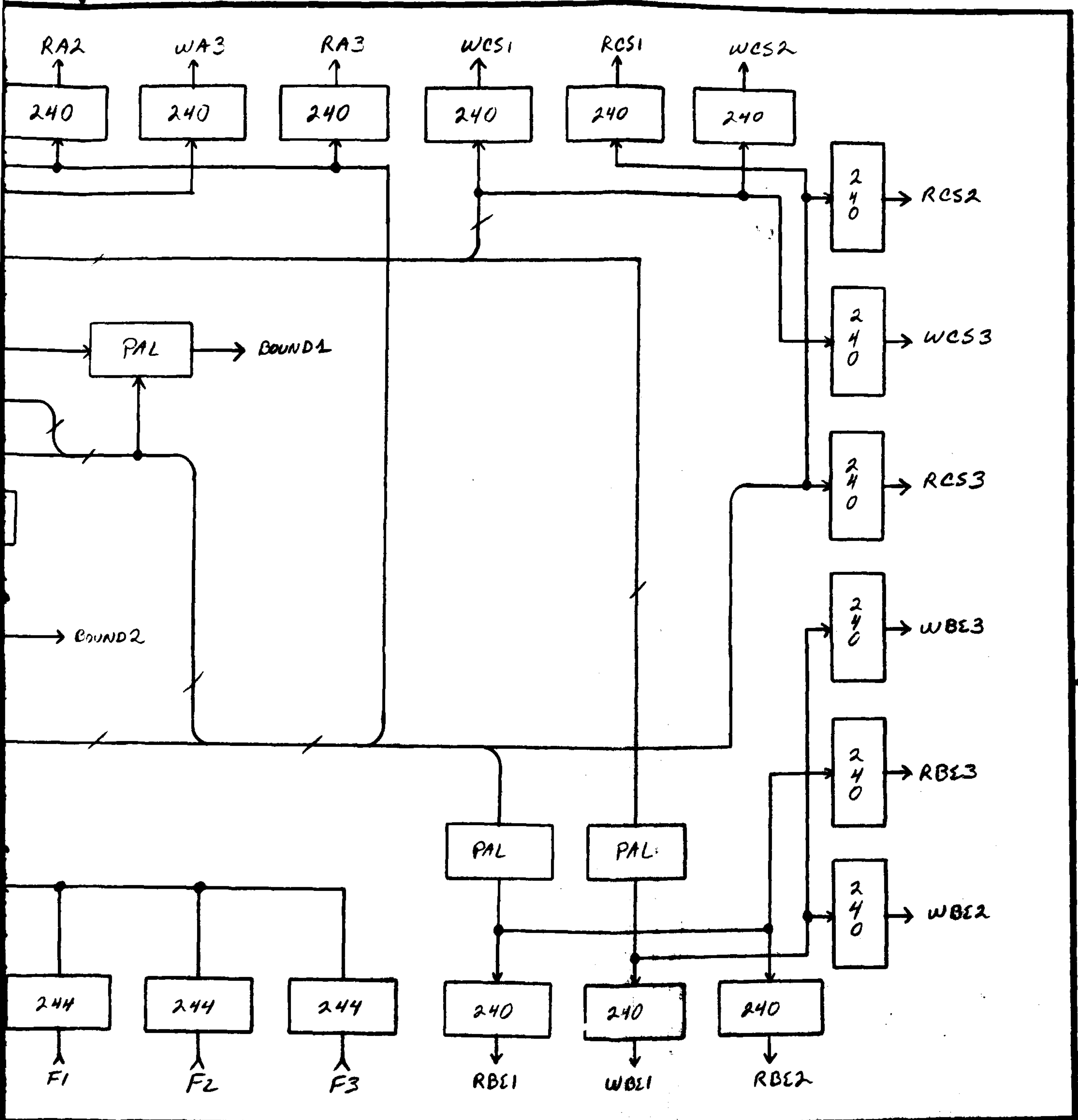


NEXT ASSY	USED ON
APPLICATION	

UNLESS
DIMEN
TO
.XX DE
±
MATER
FINISH

1

KAE 19-1253



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES ARE:
.XX DEC ± .XXX DEC ± ANGLE ±

MATERIAL

FINISH

ON

CONTRACT NO.
DAAK70-81-C-0255

DRAWN C Schlief 29 Mar 82

CHECKED

APPROVED

APPROVED

RECORDED

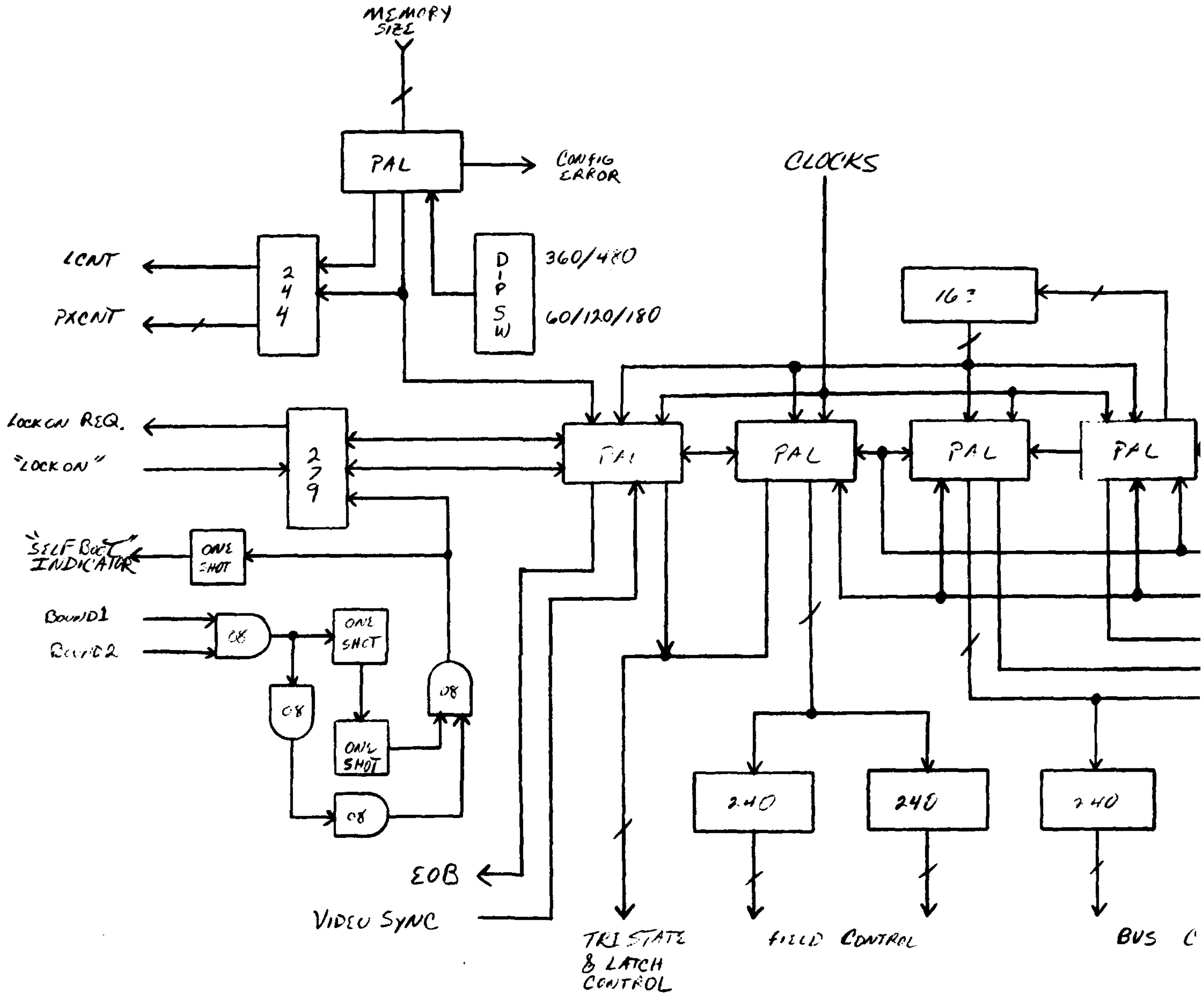
CAI a Division of RECON/OPTICAL, Inc.
BARRINGTON, ILLINOIS 60010

FRAME STORE ADDRESS CONTROL

SIZE B
FSCM NO. 11671
DWG NO. 7901-48

SCALE
WT ACT CALC LBS
SHEET

FIGURE 7.3a



NEXT ASSY	USED ON
APPLICATION	

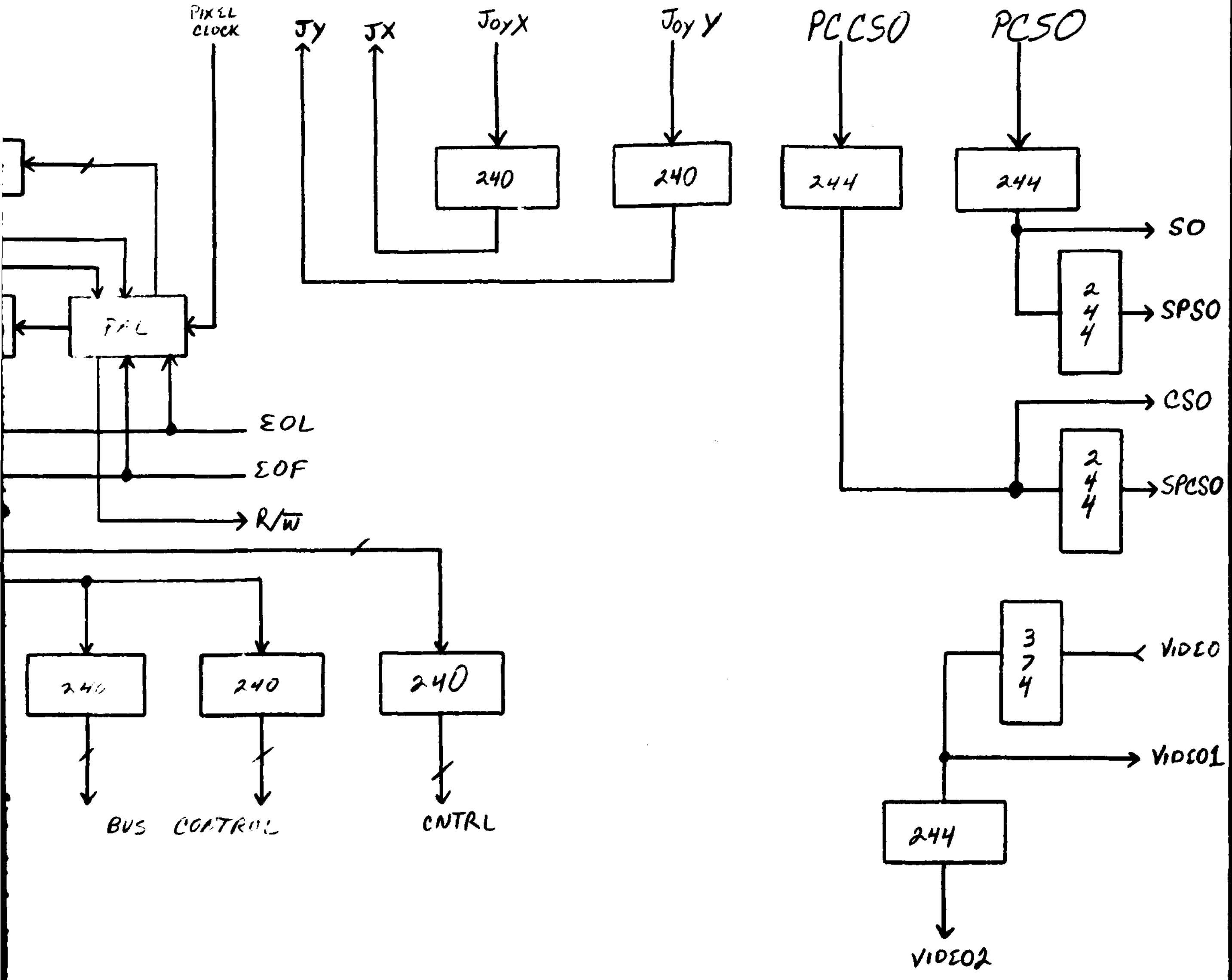
UNLESS
DIMENS
TOL
.XX DEC
±
MATERI
FINISH

1

K&E 19-1253

REVISIONS

REV	DESCRIPTION	DATE	APPROVED



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES

TOLERANCES ARE:

.XX DEC	.XXX DEC	ANGLE
±	±	±

MATERIAL

FINISH 2

ON

CONTRACT NO.
DAAK70-81-C-0255

DRAWN		29 Jul 82
CHECKED		
APPROVED		
APPROVED		
RECORDED		

CAI a Division of RECON/OPTICAL, Inc.
BARRINGTON ILLINOIS 60010

FRAME STORE MASTER CONTROL

SIZE	FSCM NO.	DWG NO.
B	11871	7901-4A

SCALE	WT ACT CALC	LBS	SHEET

FIGURE 7.38

special function module. It has already been shown that a 4 x 4 array of 60-pixel by 480-line segments is desirable for the latter, and there is no reason to differ in the basic frame store. Three fields of storage are required for proper operation: one for the writing of current data, one to hold the previous field until it can be correlated with the next field, and one to read for correlation with the current field. It is not possible to combine the first and third memory fields into one because offset conditions could be such that memory locations would be written with current data before they are required to be read for correlation. Figure 7.10 illustrates the sequence of operation. Note that alternate fields are written into memory in reverse order. There are several benefits to this method. First, addressing control is simplified since the counters need only count in opposite direction rather than be reloaded with zero. Second, no special control is needed for bootstrapping. Bootstrapping merely consists of holding the field control bit for one field. Third, scene locations occupy nearly the same addresses in each field. This is important to simplify the window locating and marking functions.

In summary, while the relationship between the actual hardware design and block diagram form discussed in section 2 may not be immediately obvious, they are functionally equivalent. The hardware design is achieved with the fewest number of available components, while the block diagram was "designed" to explain the operation.

It has been mentioned that this memory operates in an unstabilized mode so that current video is written in normal order, but to track offsets, the stored image is retrieved with offsets added to the addresses. There are several implications associated with this procedure. The first, already discussed, is that reading and writing may overlap so that three memory planes are require. Second, addresses will have to be checked for validity after offsets have been added. And third, when an image is undergoing either skew or rotation, the scan direction offset may be sawtooth waveform. Because of this possiblity, the scan direction offset of the first window in each line must be latched and reloaded at the end of a line. This allows the system to track skew along a line without losing reference from the end of one line to the beginnng of the next as shown in figure 7.11.

BASIC FRAME STORE – MEMORY OPERATION

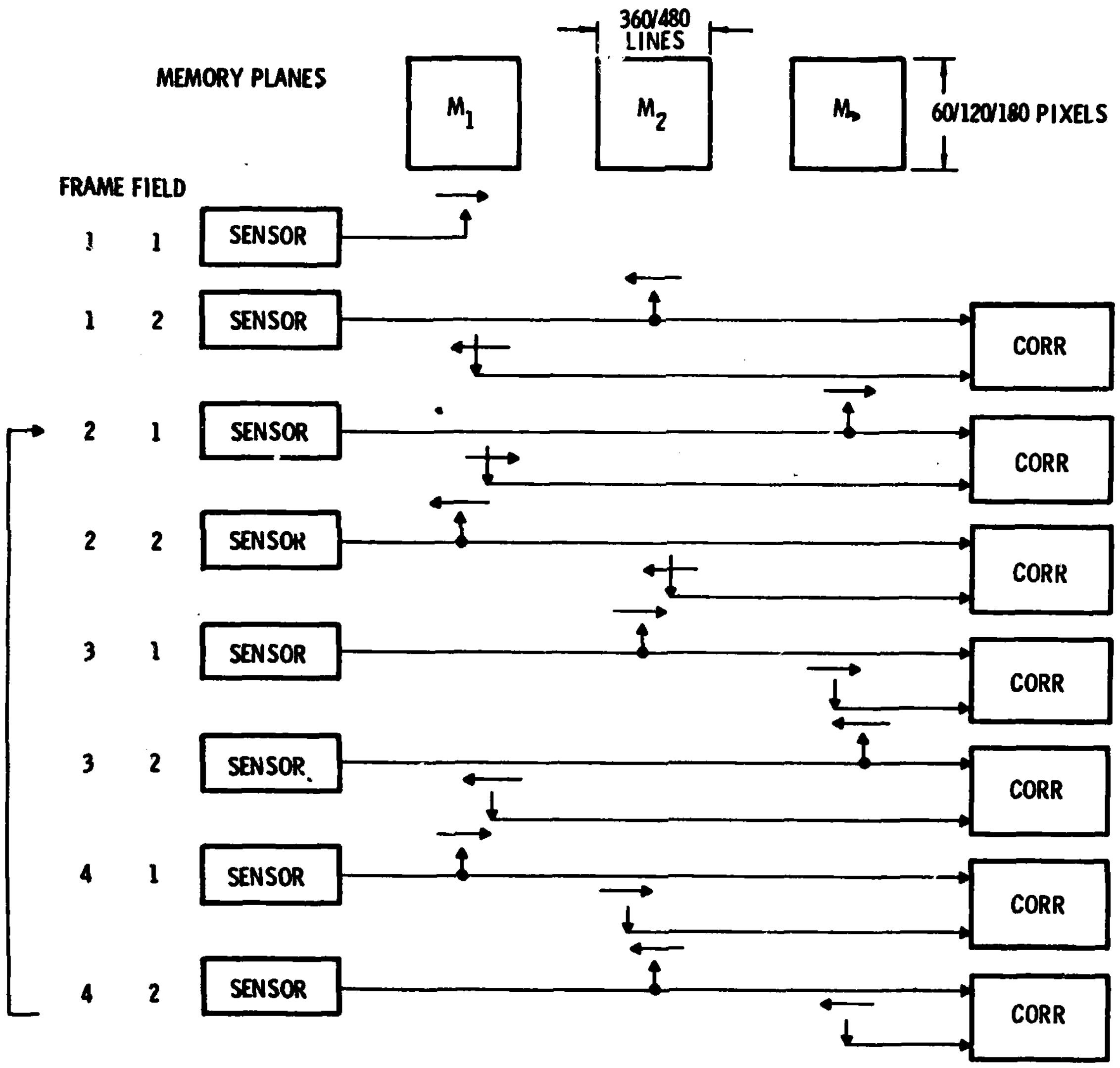
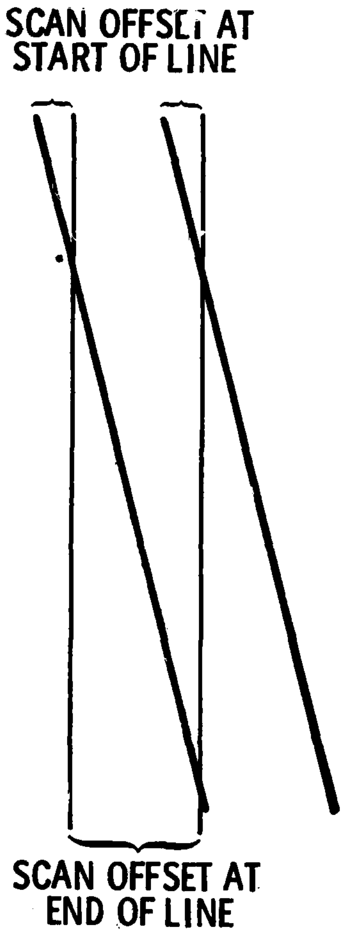


FIGURE 7.10

SKEW OR ROTATION



NOTE: SCAN OFFSET WILL NOT TRACK BETWEEN END OF ONE LINE AND START OF NEXT;

OFFSET MUST BE LATCHED AND RELOADED START OF LINE

FIGURE 7.11

The most obvious method to provide address with offset is to add or subtract a count from the base address. As previously pointed out, subtraction is not as easy a procedure as it might seem. Therefore, a bias offset would seem appropriate to avoid adding negative numbers. However, this will complicate address decoding. Furthermore, this method is somewhat inflexible in allowing a change in offset of only one location per update. An improved method is achieved by decoupling the nonoffset addresses and the offset addresses. Being decoupled, care must be taken to assure synchronization between the two address generators.

The alternate method, shown in figure 7.12, uses a multiply/accumulator. At one input, a constant "1" is presented to the device while the other input is an integer which is determined by the offset data. For no offset change, a 1 is presented and the accumulator increments one address position as would be expected. To effect a +1 offset change, a 2 is presented and the address skips a location. Similarly, a -1 offset change causes a 0 to be presented and the address holds for a count. This device allows very sophisticated and flexible control functions. For example, address resetting at the end of a line may be easily implemented by presenting a -60 to one input and a 1, 2 or 3 to the other, depending on the sensor line length. Alternatively, a slight skip at the end of a line would allow line and pixel address control in a single device.

The rest of the design and operation of the frame store is relatively straightforward, involving chip selection, address decoding and read/write control.

MTI/RANGING CARD

The MTI/range function consists of calculating offsets for small square windows and providing direct output, binary output of offsets greater than a threshold or a scaled output. Since the scan direction offset is already calculated over the desired window by the pipeline correlator, these results can be used directly. For the cross scan direction, however, the window of results from the pipeline correlator is a single line. Therefore, the partial results must be tapped and summed over a square window before final processing to derive an offset.

For motion outputs, the average motion should be calculated and subtracted from each result, so that the results do not include sensor motion. Further, it may

ADDRESS OFFSET

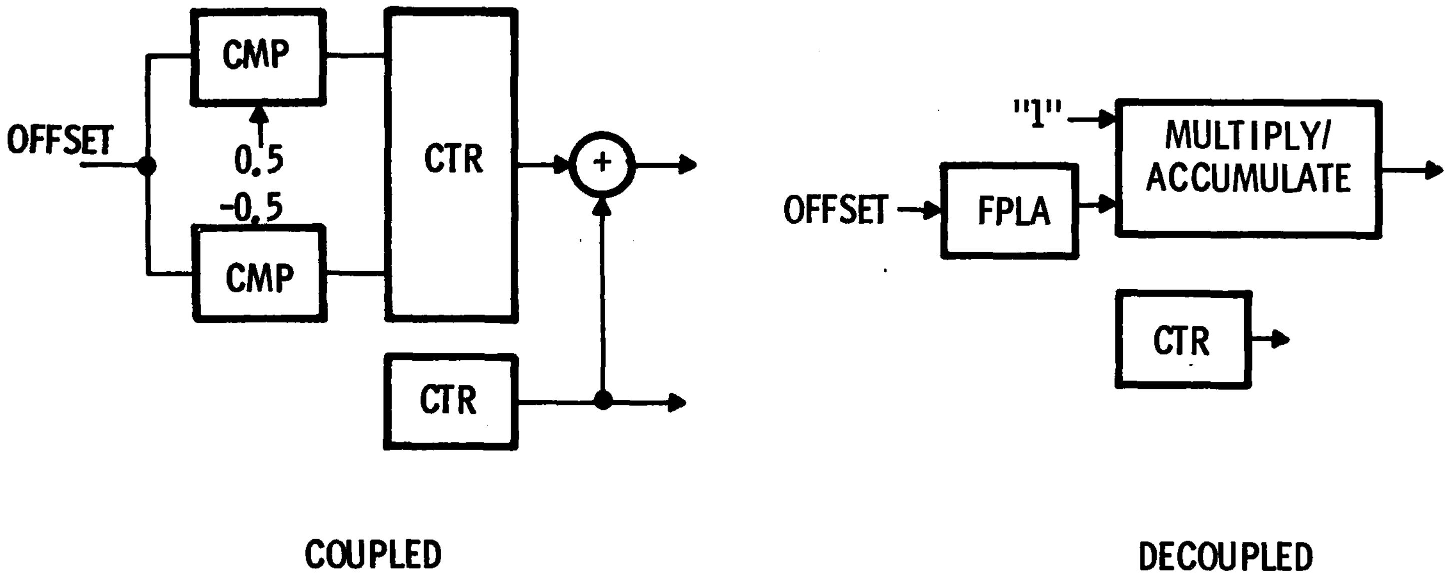


FIGURE 7.12

be desirable to calculate the magnitude of motion from the two-axis results. For ranging, on the other hand, normalization is not desired nor are the axis results combined. Range is determined solely by the cross scan offsets, while the scan offsets are used to synchronize the scan mirrors of the two sensors.

For range or motion, there are three options: thresholding for "within range" or MTI outputs, unscaled outputs for relative maps or scaled output for absolute results. Figure 7.13 illustrates the card design in block diagram. Except for the motion magnitude, which is performed as a look-up table in PROM, the design considerations and operation are virtually identical with the basic pipeline correlation function.

SPECIAL FUNCTION MODULE

The special function module is very similar to the correlation frame store. The memory architecture is identical except that only two planes are required, one for each field since there is no possiblity of contention between reading and writing. While one field is being stored in one plane, the other field is being read from the second plane. In this memory, when the image is to be passively stabilized, the current video is written to addresses with offset, while reading is performed in normal order without offset. This is the reverse of the procedure for the correlation frame store.

It is in the control where the differences between this special function module and the correlation frame store lie. The handling of offsets is the same as described for the latter - that is, using multiply/accumulators, but here the power of this method is used more fully. For example, when the image is being dynamically sampled, the address counting, the offsets and the window position must be scaled by the scaling ratio. This is easily implemented by including scale ratio as an input to the FPLA, so that the multiplicand reflects this consideration. There has been some rearrangement of how input terms to the multiply/accumulator are handled, but the general operation is the same for both frame stores.

MTI/RANGE CARD

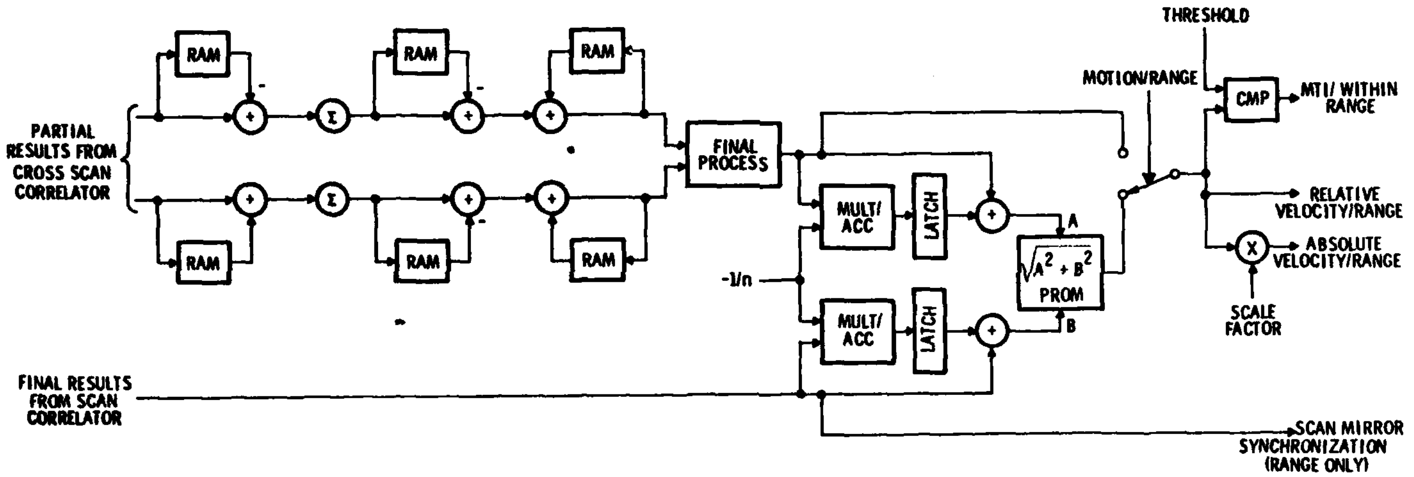


FIGURE 7.13

As previously discussed, it is necessary to accumulate offset results to maintain reference between the unstabilized correlator frame store and the stabilized special function frame store. The accumulation is quite similar to the window partial results accumulations used on the correlation cards. A RAM is used to store pixel results along a line, or line results along a frame, and fresh offset data is summed to the sum of prior data. When initializing a fresh scene lock-on, the memory is effectively cleared by adding in zeroes for the first frame.

The controller design is shown in figure 7.14.

Another function which is found in this module is scene integration. There are several possible ways to implement this function. This integration can be written in two forms:

$$S' = X \times C + (1 - X) S$$

or: $$S' - S + X (C - S)$$

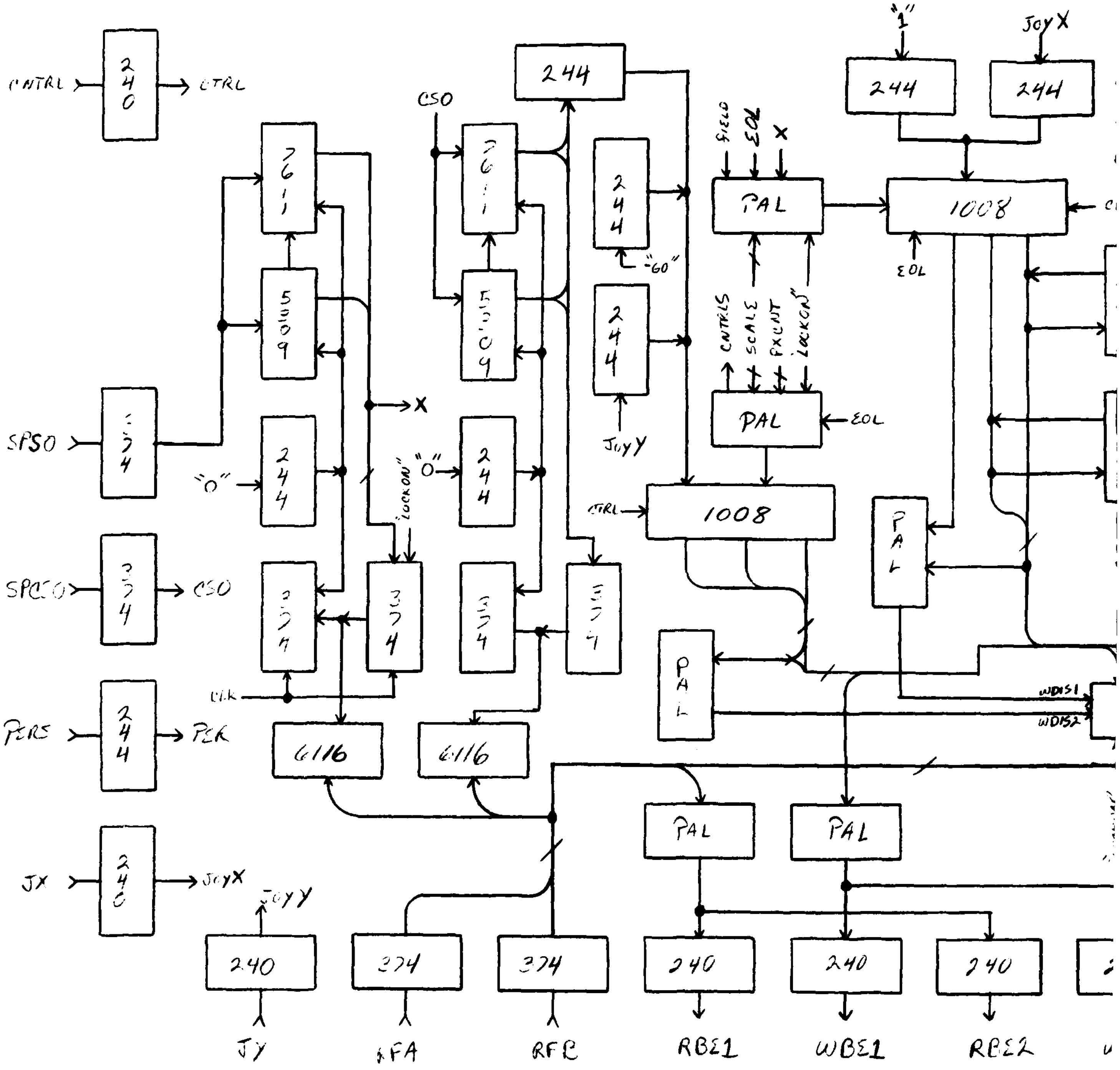
where:

C = current video

S = stored video

S' = result of integration to be stored

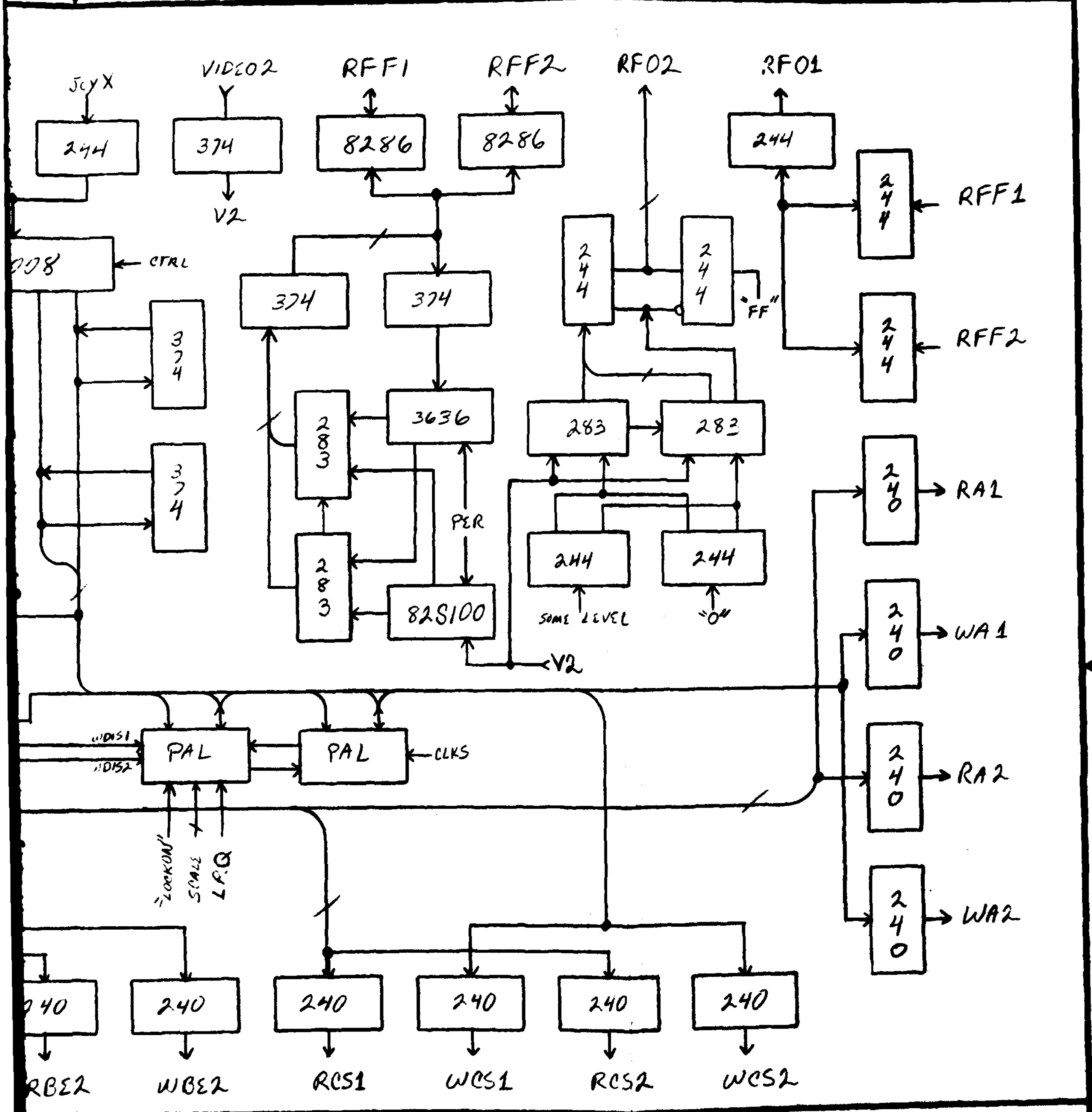
Figure 7.15 illustrates the implementations which were considered. Method 1 uses several components, including two high-speed multipliers, which use disproportionate amount of real estate for the task. Method 2 is somewhat less expensive in cost and real estate, but still uses a high-speed multiplier and the subtraction function, as has been previously stated, is not a simple operation. Both of the first two methods allow virtually any value of persistance. The third method limits persistance values to eight possible values. Prior discussion has shown that eight integration ratios are quite sufficient. Now, a simple look-up table suffices to perform the (1-X)S operation, while the X x C operation is easily coded in FPLA. The third approach results in the least expensive, fastest and smallest configuration, a rare occurrence.



NEXT ASSY	USED ON
APPLICATION	

UNLESS
DIMEN
TOL
.XX DEC
±
MATERI
FINISH

1



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES

TOLERANCES ARE:

.XX DEC .XXX DEC ANGLE

± ± ±

MATERIAL

FINISH 2

N

CONTRACT NO.

DAAK70-81-C-0255

DRAWN C Schaefer 29 Mar 82

CHECKED

APPROVED

APPROVED

RECORDED

CAI a Division of RECON/OPTICAL, Inc.

BARRINGTON, ILLINOIS 60010

SPECIAL ADDRESS CONTROL

SIZE B

FSCM NO. 11871

DWG NO. 7901-5A

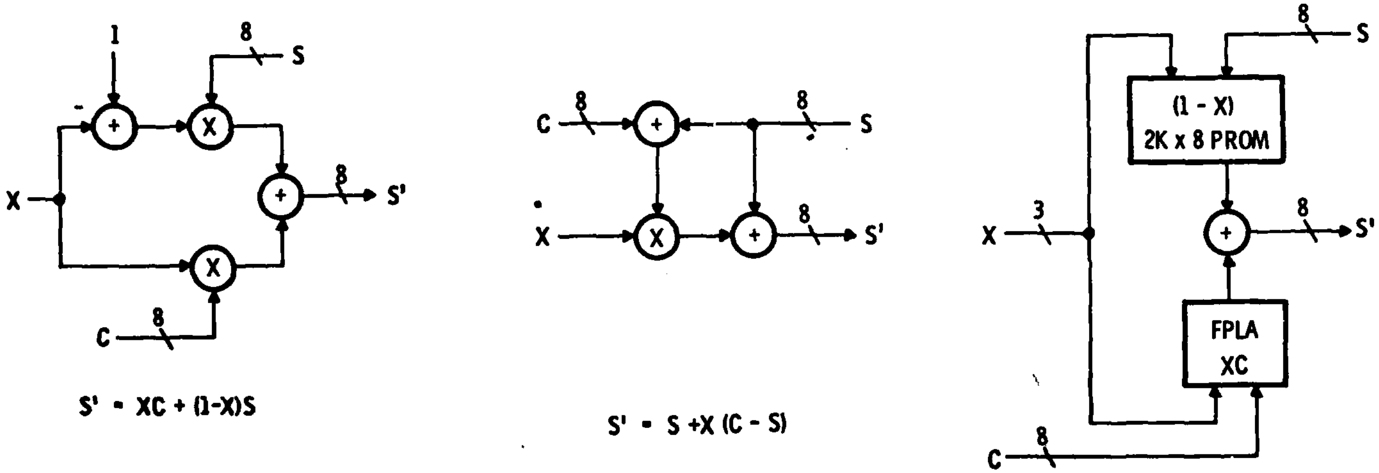
SCALE

WT ACT CALC

LBS

SHEET

SCENE INTEGRATION



$S' = XC + (1-X)S$

METHOD 1

$S' = S + X(C - S)$

METHOD 2

$S' = XC + (1 - X)S$

METHOD 3

FIGURE 7.15

CONTROL PANEL

For the proposed brassboard design, a single control panel will be used (shown separately as figures 7.16 and 7.17). The control panel allows for entering pixel size, focal length and baseline or range data via a pushbutton keyboard. Information such as threshold, absolute velocity or range, and error conditions are displayed on an alphanumeric display. The window position is controlled by a set of four pushbutton switches arranged as an effective joystick. The switches are used instead of a joystick for their lower cost and more rugged construction. They also eliminate the danger of having a protruding lever on the control panel. Other controls include scale, (1x, 2x, 4x), stability and integration persistance.

The heart of the control panel is designed with an 8251 single-chip microprocessor. The microprocessor provides on-chip ROM, RAM, a serial port and parallel ports. The version of this microprocessor that will be used for the brassboard design is the 8031, which includes all the above features except on-chip ROM, which is supported by a 2716 EPROM. This redues the cost for the initial design. Communications to and from the main electronic box are accomplished by an EIA Standard RS-232C interface, which requires only three wires. This feature reduces the cable size requirements. The MIL-STD-1553B interface could be substituted, but the additional cost due to the increased parts count and software design time is not considered cost-effective for a brassboard system. The communication interface in the main electronics box is also an 8251 single-chip microprocessor that supports ancillary functions. These functions include non-critical mathematical computation, system configuation and system initialization. Also, errors can be sensed and displayed on the control panel, as shown in figure 7.18.

The Intel 8279 keyboard/display interface is used for controlling the keyboard and display. This eliminates the need for the microprocessor to "poll" the keyboard (look for a key pressed) and multiplex the displays. It also allows greater flexibility in the software design.

MTI/RANGE CONTROL

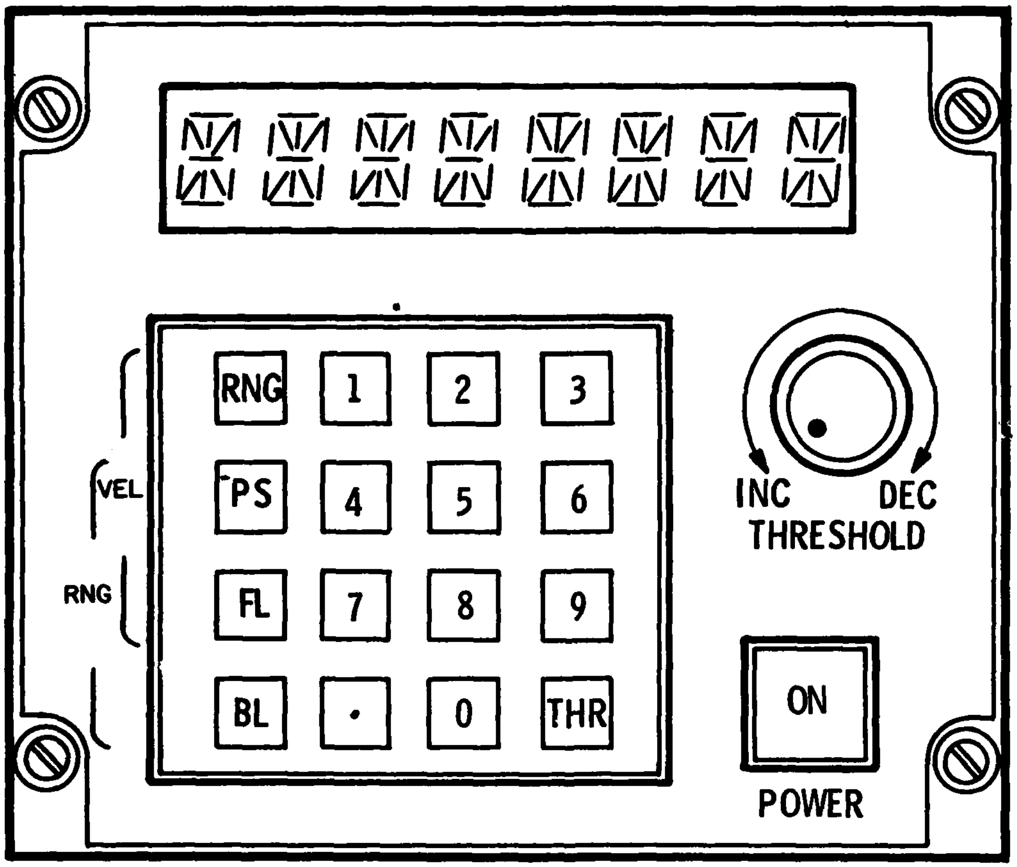


FIGURE 7.16

STABILIZATION CONTROL

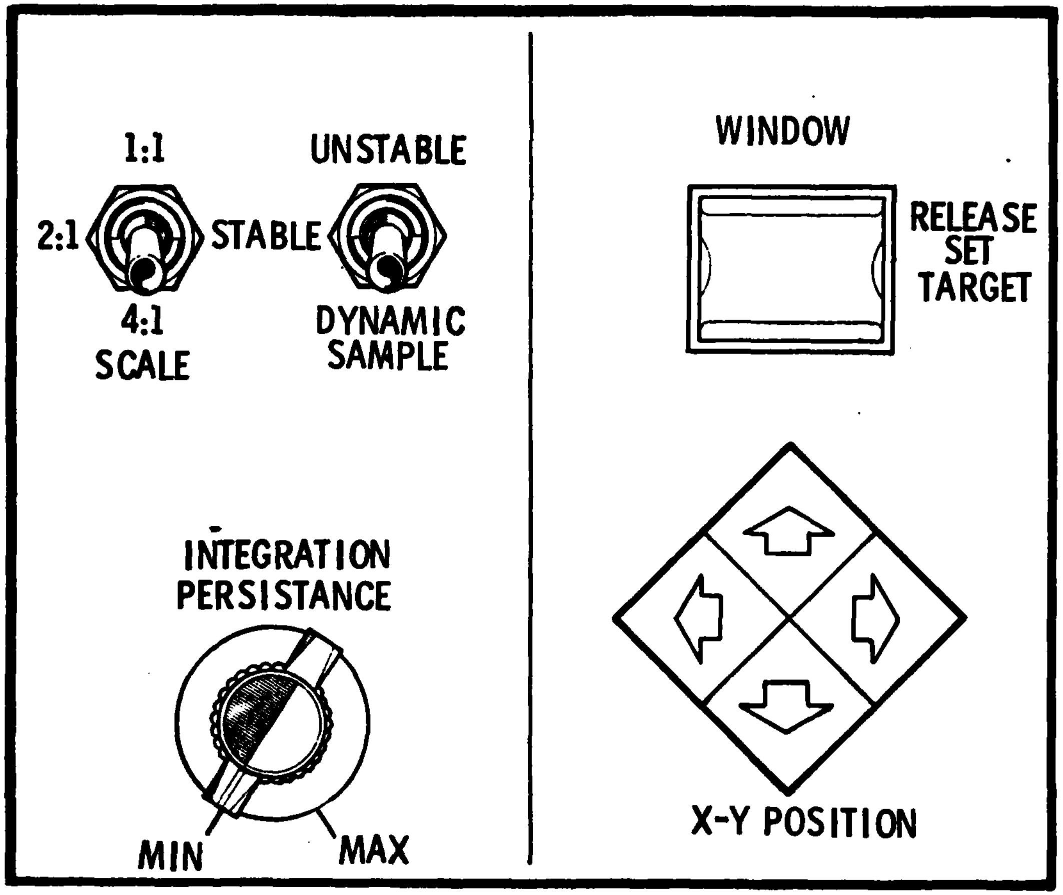
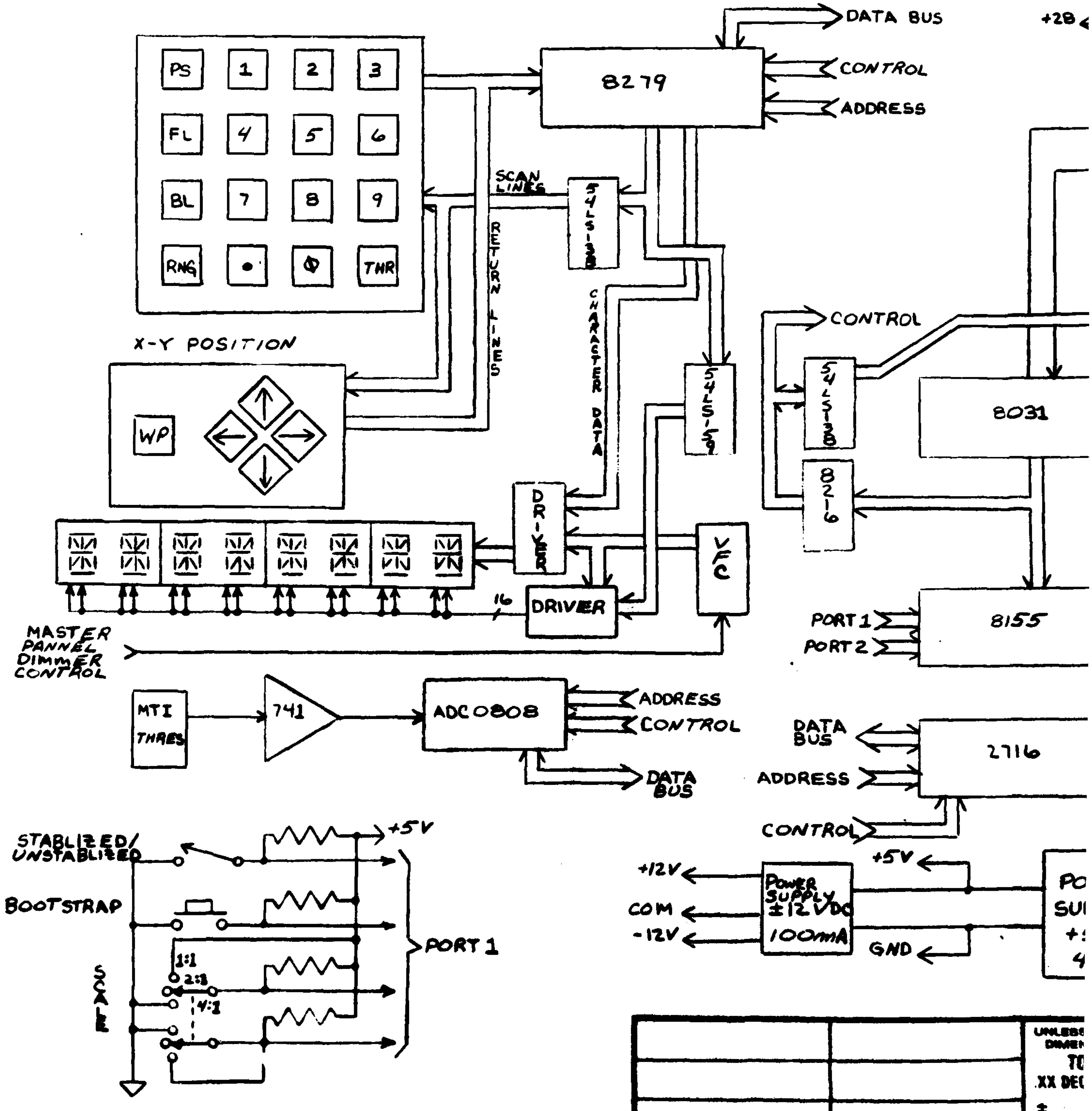


FIGURE 7.17

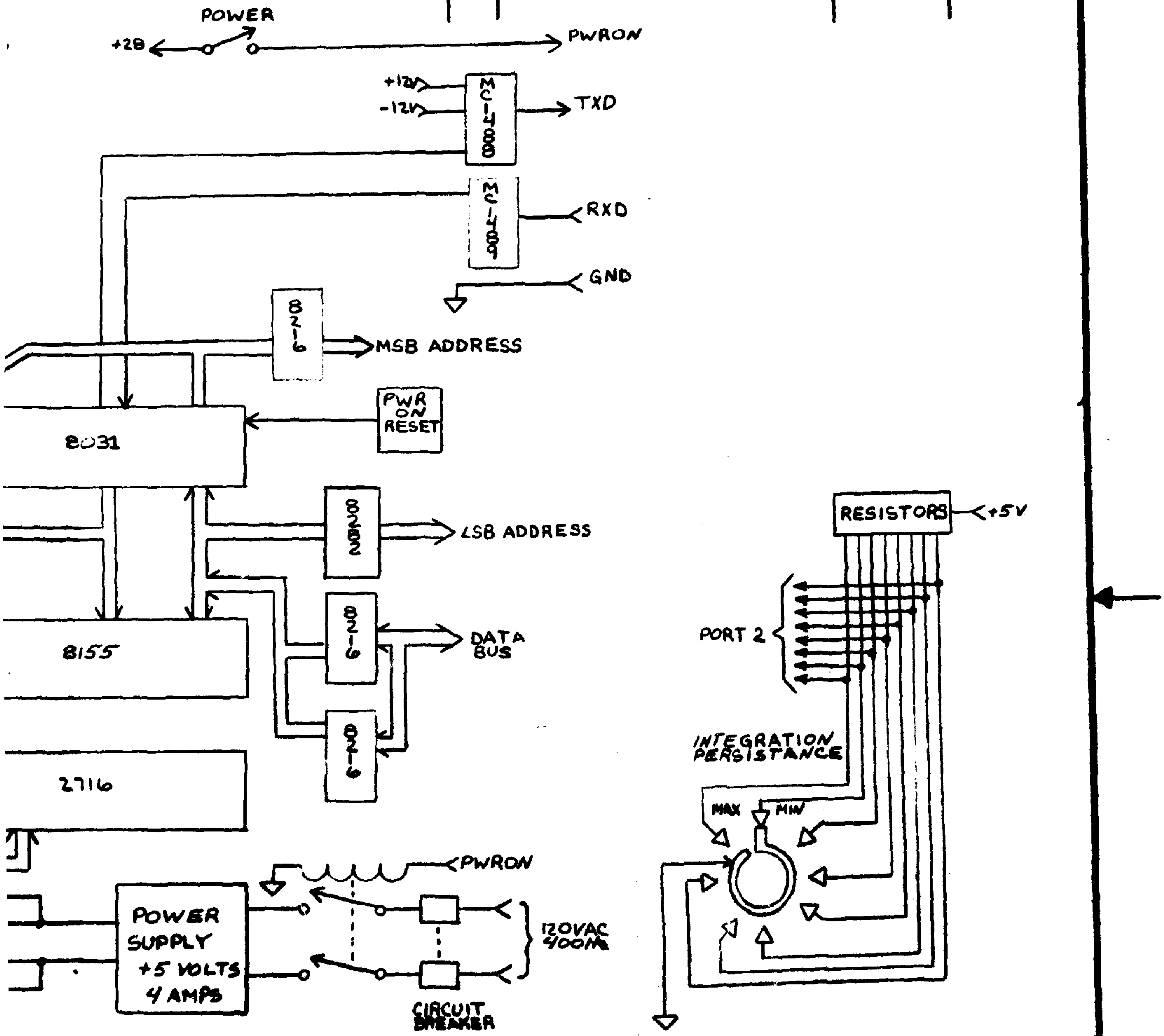


NEXT ASSY	USED ON
APPLICATION	

UNLESS
DIMEN
TO
XX DEC
±
MATER
FINISH

7901-1B

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED



UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES
TOLERANCES ARE:
XX DEC ± .XXX DEC ± ANGLE ±

MATERIAL

FINISH

ON

CONTRACT NO. DAAK70-81-C-0255

DRAWN T. GOODRICH	3/25/82
CHECKED	
APPROVED	
APPROVED	
RECORDED	

CAI a Division of RECON/OPTICAL, Inc.
BARRINGTON, ILLINOIS 60010

FLIR STABLIZATION MODULE
CONTROL PANNEL

SIZE	FSCM NO.	DWG NO.
B	11871	7901-1B

SCALE WT ACT CALC LBS SHEET 1 of 1

TABLE VII-I
SIZE, WEIGHT, POWER

	Approximate Dimensions	Volume in^3	Weight	Power Watts	Qty
Control panel	9 x 6 x 5	270	5	5	1
Control board	8 x 5 x 1.5	60	1	15	1
Pipeline correlator board	9 x 7 x 1.5	94.5	1.4	18.6	2
MTI/range board	9 x 7 x 1.5	94.5	1.4	18.5	1
Master control board	9 x 7 x 1.5	94.5	1.4	10	1
Address control board	9 x 7 x 1.5	94.5	1.4	34.4	1
Special address control board	9 x 7 x 1.5	94.5	1.4	28.9	1
Memory board	9 x 7 x 1.5	94.5	1.4	23.8	5
Power supply	12 x 9 x 3.5	378	7	-	1
Totals			28.4	268	

SECTION VIII
PRODUCTION DESIGN CONSIDERATIONS

A modular design approach will be used for production models. Four mother boards will be designed to accomdate the various options in a reconfigurable manner.

When fully configured, the system supports the pipeline correlator, frame store, MTI/range and communication interface and spacial functions. Internal DIP switches allow for selecting line width and pixel count. Each electronics enclosure can have its own power supply or a common supply can be used.

Two control panels are used for the full system. This design allows for separate control panels for special function and MTI/range control. These panels are designed to meet MIL-STD-25212. The production units will use the 8251 single-chip microprocessor to save board space and reduce total package count. A switch located inside the main electronics box allows the system power to be controlled at the control panel.

Each option is configured with a frame store (memory) for 60 pixels. Additional frame storage is available in 60-pixel increments, with a maximum number of 180 pixels.

Option 1 is a basic stabilization system which uses only the main electronics enclosure with the pipeline correction, memory and control boards installed. Option 2 adds the MTI function to the basic system. With this option, the MTI/range control panel is included along with the MTI/range board in the main electronics enclosure. Option 3 provides the stabilization and ranging functions. This option uses the main electronics enclosure with the pipeline corelator, MTI/range and master control boards, however the frame storage is not required. The MTI/range control panel is also used. Option 4 features stabilization and special functions. Both control panels are used for this option. Option 5 contains all the features (MTI/range, stabilization, special functions) and is considered the full system configuration.

The only differences between the brassboard system and production design are the operation of functions onto dedicated mother boards, the use of printed circuit boards, and the use of only military-qualified versions of components.

SECTION IX
CONCLUSIONS AND RECOMMENDATIONS

This study and preliminary design has shown that correlation processing can be used for FLIR sensor stabilization with better performance than gyro transducers, while providing many other useful and powerful functions. In particular, precision correlation has been shown, by a computer simulation program, to measure image displacements with errors under 1/20 pixel. This accuracy, combined with the high-speed pipeline implementation, allows new types of information to be displayed as images: motion maps, range maps obtained without active sources, and dynamically-sampled imagery with resolution up to 3.5 times the Nyquist limit.

The preliminary design effort has developed a modular approach which allows users to select the particular functions required for their application and to add or delete functions as desired without modification to other portions of the system. Considerable attention has also been paid to size, power and cost in the preliminary design, as well as future transition to a fully militarized production system. The result is a versatile, affordable menu of options.

It is recommended that a brassboard system be fabricated to demonstrate the capabilities described. The system will be used in conjunction with the common module digital scan converter. Video input to the system will be taken from the digitized, multiplexed output and results will be displayed via the reformatter. It is recommended that the brassboard system be configured for 120-element array sensors. The system will demonstrate passive stabilization, motion mapping, scaling, frame integration and dynamic sampling. With two boresighted synchronized FLIR sensors, the system can also demonstrate passive ranging and display of a range map.

APPENDIX A
PRECISION CORRELATION

INTRODUCTION

Correlation can be used for two purposes:

- To determine degree of match by determining the height of a correlation peak
- To measure the relative alignment between two signals by measuring the position of the correlation peak

It is the latter purpose which CAI has addressed with precision correlation. There are several applications for precision correlation. These include stabilization, tracking, MTI, passive ranging, navigation and terrain following.

CORRELATION THEORY

The applications of precision correlation depend upon the ability to determine the relative shift between two images. Succinctly, it is required to measure how far one image must be displaced to match the second image. The first task is then to define match in mathematical terms.

Since brightness is the only information with which to work, match must be defined as occurring when each sample in one image plane has the same amplitude as its corresponding sample in the second image plane. Thus,
$P_1\ (i,j)\ =\ P_2\ (i + \Delta\ i, j + \Delta j)$ for all i and j where Δi and Δj
define the shift required to make the match using some coordinate system. In reality, noise, lens mismatch, specular reflections and the like prevent the difference from ever being precisely zero over all i and j; hence, the need to find the best match or fit. One common measure of fit, given numerous samples, is to take the least squares approach in which parameters are varied until the sum of the squares of the differences shows a minimum. In this case, the parameters to be varied are Δi and Δj. There are, of course, other measures of fit such as minimizing the sum of the absolute values of the differences. In this vein, it can

be generalized to minimize $\left\{\left[P_1\ (i,j)\ -\ P_2\ (i\ +\ \Delta i,\ j\ +\ \Delta j)\right]^2\right\}^Q$ where, for example, Q = 1 and 1/2, respectively for the two cases mentioned. It can be shown that the least squares approach is mathematically equivalent to the statement that the mean of statistical sample is the best estimator of that sample, while the least absolute value is equivalent to the statement that the median is the best estimator. Much early work in correlation matching was devoted to determining which statement is the more valid for imagery, with the conclusion that the least squares approach yields a less noisy correlation curve.

Deciding a priori that average brightness and gain differences between the two images are to be ignored in determining match, the sample sets are normalized by subtracting the mean and dividing by the standard deviation. For the least squares approach, a match is defined by locating the minimum of:

$$\Sigma\left[\frac{P_1\ (i,j)-\overline{P_1}}{\sigma_{P_1}}^{\ 2}\ -\ \frac{P_2\ (i+\Delta i,\ j+\Delta j)-\overline{P}_2}{\sigma_{P_2}}\right]^2$$

Expanding:

$$\text{Min}\ (\Delta i,\Delta j):\quad \Sigma\left[\frac{P_2\ (i,j)-\overline{P_1}}{\sigma_{P_1}}\right]^2\ +\ \Sigma\left[\frac{P_2\ (i+\Delta i,\ j+\Delta j)-\overline{P_2}}{\sigma_{P_2}}\right]^2$$

$$-2\Sigma\left[\frac{P_1\ (i,j)-\overline{P_1}}{\sigma_{P_1}}\right]\left[\frac{P_2\ (i+\Delta i,\ j+\Delta j)-\overline{P_2}}{\sigma_{P_2}}\right]$$

Simplifying:

$$\text{Min}\ (\Delta i,\Delta j):\quad \Sigma\ P_1'^{2}\ +\ \Sigma\ P_2'^{2}\ \ -2\ \ \Sigma\ P_1'\ P_2'$$

The first two summations both equal 1 by definition of mean and standard deviation. What remains is finding the minimum of 2 $\left[1-\Sigma\ P_1'\ P_2'\right]$. This is equivalent to finding the maximum of $\Sigma\ P_1'\ P_2'$ which is recognized as the peak correlation coefficient. Thus, correlation is nothing more than a statement that match is defined by minimizing the least squares difference between two signals as a function of relative shift, with the implication that the mean of the sample-by-sample differences is the best estimator of the degree of match.

By plotting the correlation product as a function of image shift, a correlation curve is generated. Because the images are, in this case, focused on discrete element sensors, the images are sampled in a discrete regular pattern. With the images sampled discretely, the correlation curve is only defined at discrete intervals. Nevertheless, these correlation samples do in fact represent a continuous curve. Furthermore, this curve shifts precisely with the relative shift of the two images.

PRECISION CORRELATION

Clearly, the image shift can be determined in a gross sense merely by locating the peak correlation sample. However, to measure a shift smaller than the sample spacing (i.e., pixel spacing), the curve must be interpolated. In essence, this means assuming a mathematical shape for the curve and calculating the necessary constants based upon the samples obtained by autocorrelation (the correlation of one image with itself) which is known to be centered at the origin. Once the constants are evaluated, the fractional pixel shift can be determined by applying these values to the samples of the correlation curve obtained by cross correlation of the two different images. This procedure assumes that the curve maintains its shape as it shifts with the relative image shift. In practice, the mathematics can be arranged so that the constants never need to be evaluated explicity, but the cross correlation sample values and autocorrelation sample values are combined in one equation to directly yield the fractional pixel displacement.

The actual shape assumed for the curve can be gaussian, exponential, polynominal, spline, trigonometric or any one of a wide variety with various levels of generalization. The performance of any algorithm making an assumption of curve shape must be assessed both in terms of how well the assumed shape fits actual shapes generated with real imagery, ease of implementation and processing speed. A mismatch between the assumed shape and actual shapes of correlation curves obviously produces errors in the calculated interpolation. Furthermore, care must be taken in deciding which samples of the curve to use since the curve is quickly dominated by noise effects immediately around the peak, yet the peak itself on the autocorrelation curve has correlated noise included which will not be found in cross correlation. Finally, filtering of the scene can help or hurt the correlation accuracy by shaping the correlation curve to some extent.

Even if real correlation curves could be well described in practice by some mathematical shape, two error sources remain. The first is noise, which is always present to degrade results. The second is fine scene structure. Spatial frequencies about and greater than Nyquist frequency are strongly phase dependent in their effects on sample amplitudes. This phase dependence can produce fluctuations in the correlation curve which are not detectable solely by autocorrelation samples. Filtering can be a solution this problem only so long as the filter does not greatly reduce the slightly lower spatial frequencies which help to sharpen the correlation curve and allow the accurate measurement of small displacements.

CAI has expended considerable effort in determining the best assumed shape consistent with ease of implementation and speed of operation. The findings have shown that without filtering, RMS correlation error of 1/50th of a pixel can be achieved. Further improvement may•be obtained with noise and/or scene filtering.

EXPERIMENTAL RESULTS

The accuracy of the algorithm is dependent on the signal-to-noise ratio inherent in the signal and on the number of signal samples used in the calculation. Figures A.1 and A.2 show computer and analytic results (which have been verified on a physical demonstration unit) of rms error versus signal-to-noise ratio and number of samples. Further accuracy is obtained by filtering the correlation results with a low-pass filter as shown in figure A.3.

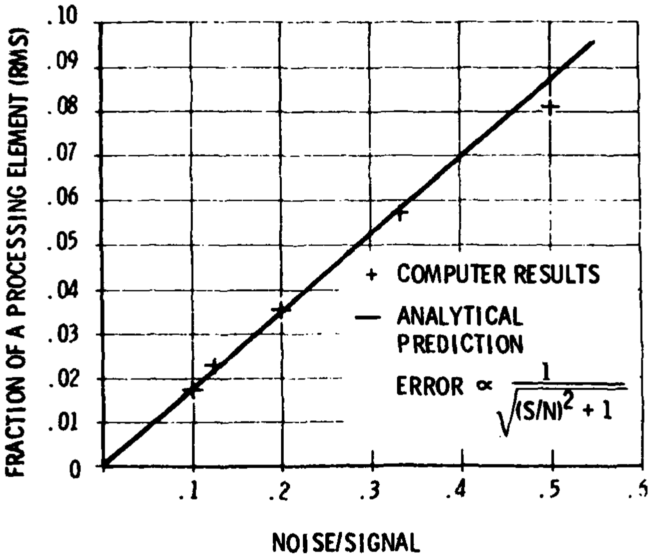


FIGURE A.1 CORRELATOR ACCURACY

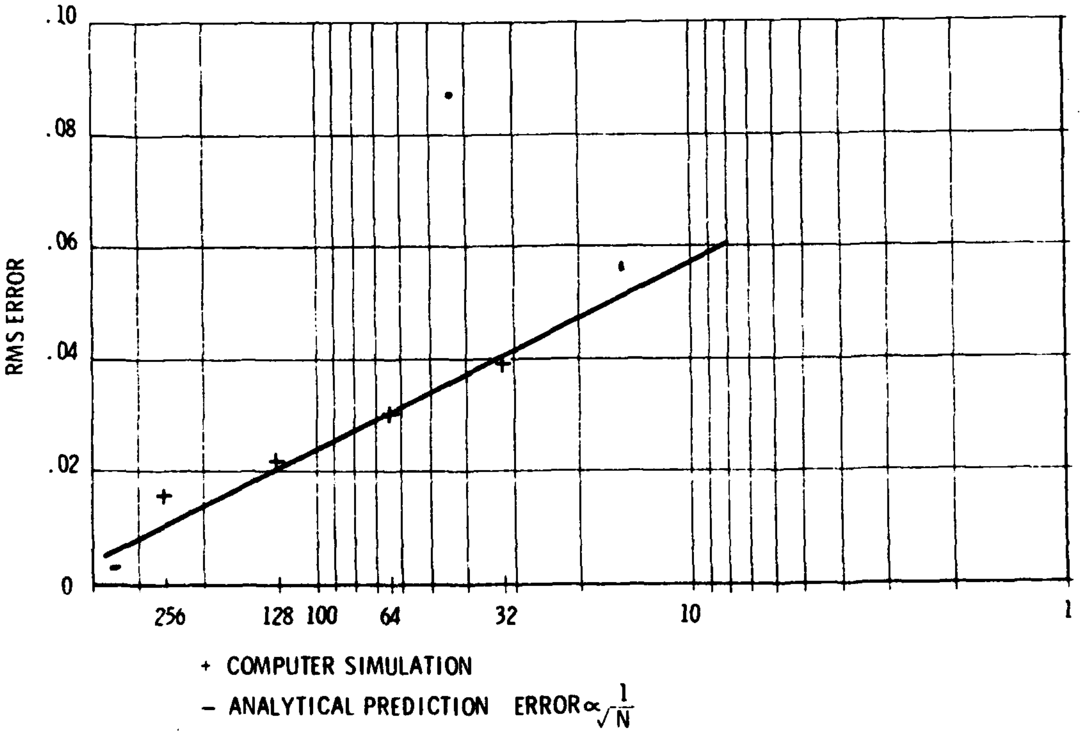


FIGURE A.2 ERROR VS NUMBER OF ELEMENTS

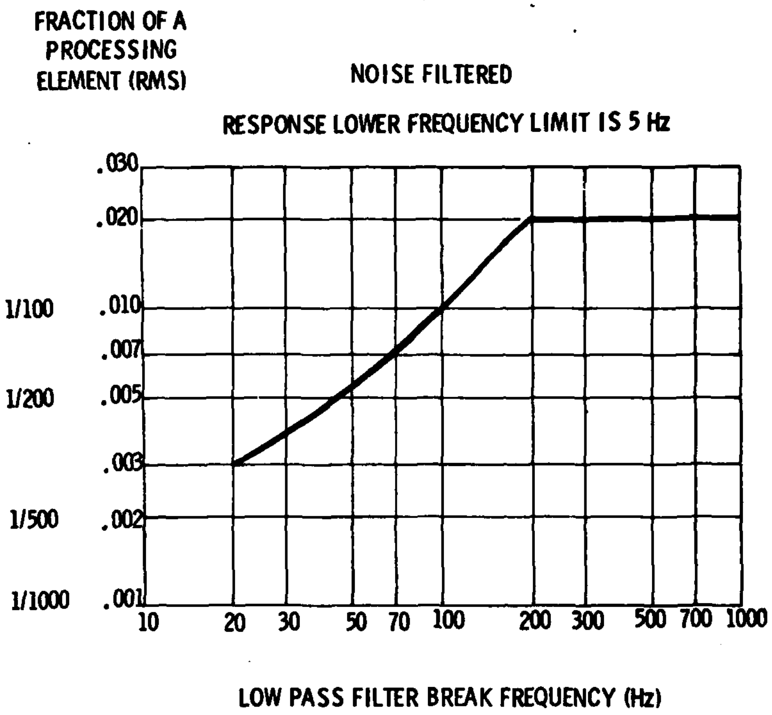


FIGURE A.3 CORRELATOR DEMONSTRATOR ACCURACY (NOISE FILTERED)

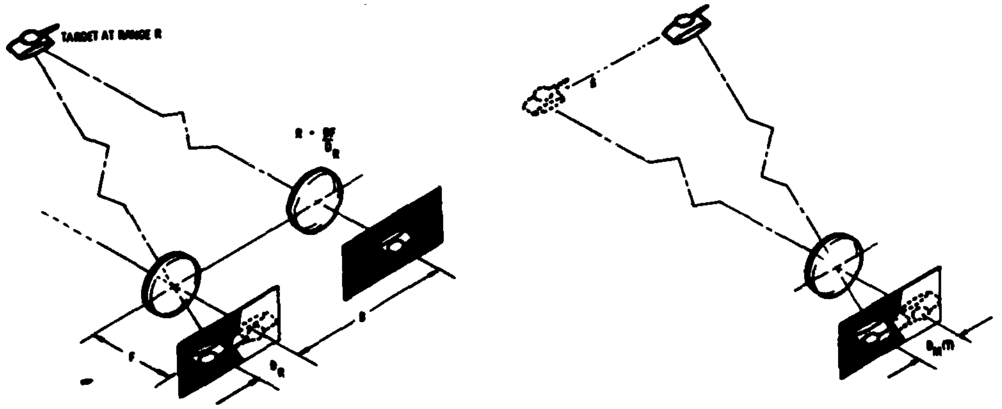


FIGURE A.4 RANGING AND TRACKING GEOMETRY

Generally, a basic correlation accuracy of 1/50th sample spacing is used for calculating expected performance. In many instances, however, as will be shown, the application may lend itself to filtering to improve the basic results.

USES AND LIMITATIONS

The most obvious immediate applications for precision image registration are tracking and its related function, stabilization. The high precision afforded is critical for high-resolution imaging, which is important in tracking for fire conrol and useful in general tracking. A less obvious application is passive ranging. Ranging is identical to tracking, except that the images displaced in time used for tracking are replaced by images displaced in space to obtain stereo images as shown in figure A.4.

The formula for range, R, given two parallel optical systems separated by a distance, B, is (noting similar triangles):

$$R = \frac{BF}{D}$$

where:

F = focal length of he optics

D = relative displacement between the object images

Assuming B and F are constant, the error is determining range is limited by the error in measuring D:

$$dR = -\frac{BF}{D^2}\, dD = -\frac{R^2}{BF}\, dD$$

Since the sample spacing on the correlation curve is equivalent to the sample spacing of the signal, that is a resolution element or pixel spacing at the image plane:

$$dR = \frac{R^2}{BF} P\varepsilon$$

where:

P = spacing between pixels

ε= algorithm accuracy

In contrast to active rangefinding with fairly constant range error, this method has a range-squared dependence. This is a disadvantage in the sense that to accommodate greater ranges, the physical package must expand for this passive technique; while an active technique requires only increased power to overcome weakening due to beam spread. Currently, power seems to be a less expensive commodity. Not only is increased package size undersirable in itself, but maintaining rigid alignment becomes impossible and other complexities must be incorporated to compensate. On the other hand, there is no range gating so no ambiguity of ranges can occur. It is also generally true that the nearer the object, the higher the accuracy desired, or at least the more critical is that accuracy. Stereoscopic ranging does not waste data bits producing highly-accurate results for far objects to achieve high accuracy on near objects. There is also a practical near-range limit due to excessive image shift. Even if the taget image has not moved off the sensor format at very close ranges, the processing cannot afford to generate a correlation curve several hundreds of samples long to locate a match.

Applications of passive rangefinding include: fire control, altimeters, TERCOM, terrain avoidance, target detection and possibly target recognition. For all but the first two applications, it is desirable to map correlation results over the image format. However, using standard techniques for calculating a correlation curve, locating a peak sample and interpolating the position of the curve peak, only a few results per frame could be computed. CAI has found that its precision correlation algorithm lands itself to pipeline implementation, allowing the kind of high-speed mapping desired.

APPENDIX B
PIPELINE PRECISION CORRELATION

OPERATION

Pipeline precision correlation (PPC) is a specialized form of CAI's precision correlation algorithm wherein each arithmetic operation is separately implemented in hardware. Two serial streams of video data are input and processed to produce two serial streams of partial results on a pixel-by-pixel basis. These partial results are summed over a desired correlation "window." The sums are then processed in a final stage to produce, on a pixel shift-by-pixel shift basis if desired, the calculated misregistration for each window. If suitably designed, the dimensions of the window can be dynamically adjusted.

Conceptually, incoming video information is simply transformed into misregistration results. The signal may be interlaced or noninterlaced and the results will be similar in format. The noninterlaced format is nonetheless preferred because the ability to correlate on small objects is improved.

"VIDEO LIKE" VERSUS BLOCK RESULTS

As shown in figure B.1, results are calculated as running results for all overlapping windows mXn. Results could also be calculated for adjacent windows only, but this would be only minimally easier to accomplish. Running "video like" results is to an extent, but not entirely, oversampling. Not only is aliasing reduced by oversampling, but the larger number of results allow smoothing, which improves the overall accuracy.

A particularly attractive implementation occurs if either the desired window width equals the scan line or the window height is a single scan line. In either case, the need for an array of shift registers or their equivalent is unnecessary.

RELATIVE VERSUS ABSOLUTE RESULTS

With time sequential applications such as tracking, absolute motion can be determined only insofar as the motion of the sensor is known.* It is rarely necessary that absolute motion of some object be determined in dynamic conditions. In ranging, however, absolute values are often desired even for dynamic conditions. The implication of absolute range with a steroscopic sensor is that the baseline separation and angular alignment be precisely known. Typically, ranging appliations requiring accuracies of 1/50th sample spacing are equivalent to determining parallax to a few microradians. Therefore, the relative angular alignment must be held or measured to such tolerances.

In terms of mechanical stability, this is a difficult proposition at best. An alternative to exceedingly rigid mechanical tolerances is incorporation of some form of autocalibration in the optomechancial system. This may be some means of allowing one sensor to "see" the other via most of the optical path within the physical limits of the system, or for each to "see" a reticle or pair of reticles, as is used in highly-precise visual rangefinders shown in figure B.2. The reticle pattern is separated from the scene both spectrally and in field position, so that each may be correlated simultaneously to obtain range regardless of any optical wedge within the system. Because the reticles can be an optimum pattern, well illuminated with good contrast and of large extent, the correlation precision achieved is expected to well exceed the typical 1/50th sample spacing accuracy of non-ideal objects.

Because of the increased physical and processing complexity required to ensure accurate absolute ranging, it is appropriate to examine which applications require it. Certainly fire control appliations do require absolute range to solve the gun directing equation. A TERCOM application conversely can use relative data, since it is only the terrain contour which is of interest. Care must be taken when matching to normalize not only for offsets in absolute range due to uncertainty of altitude, but for offsets in reciprocal range due to misalignment of optical channels.

* With a stationary background, the sensor motion is easily calculated as the reverse of the apparent background motion.

PIPELINE PRECISION CORRELATION FOR m X n WINDOW

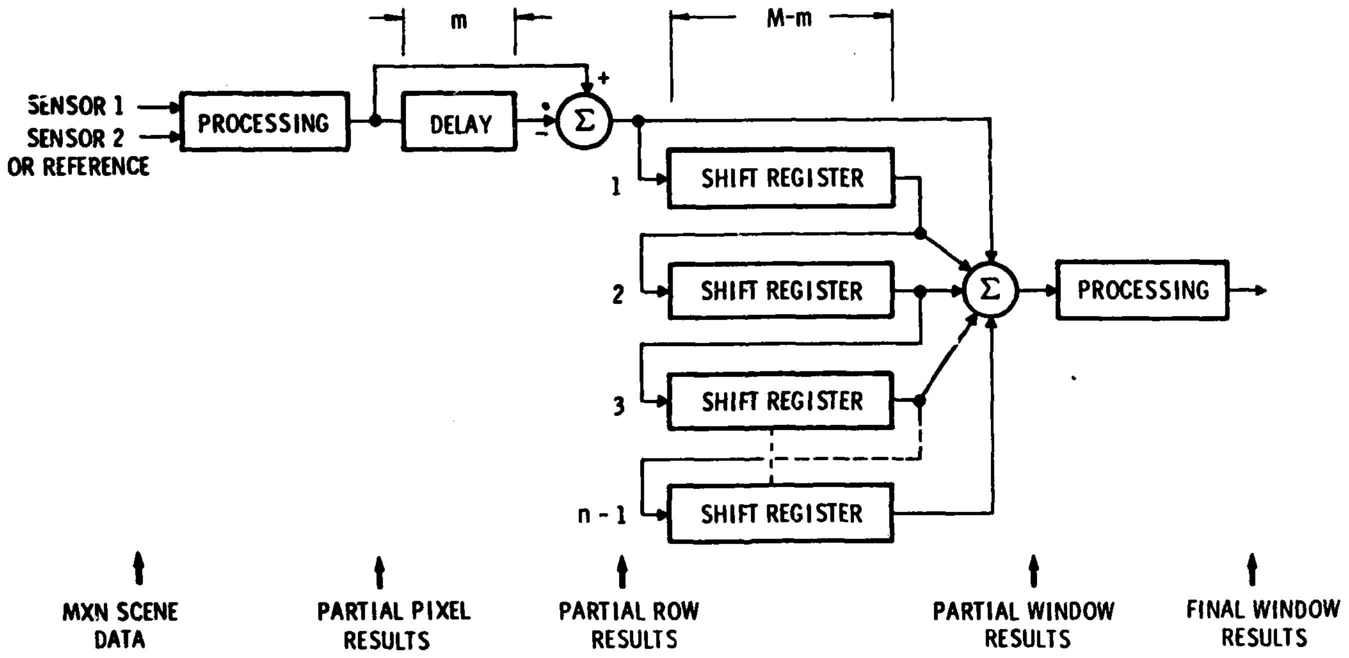
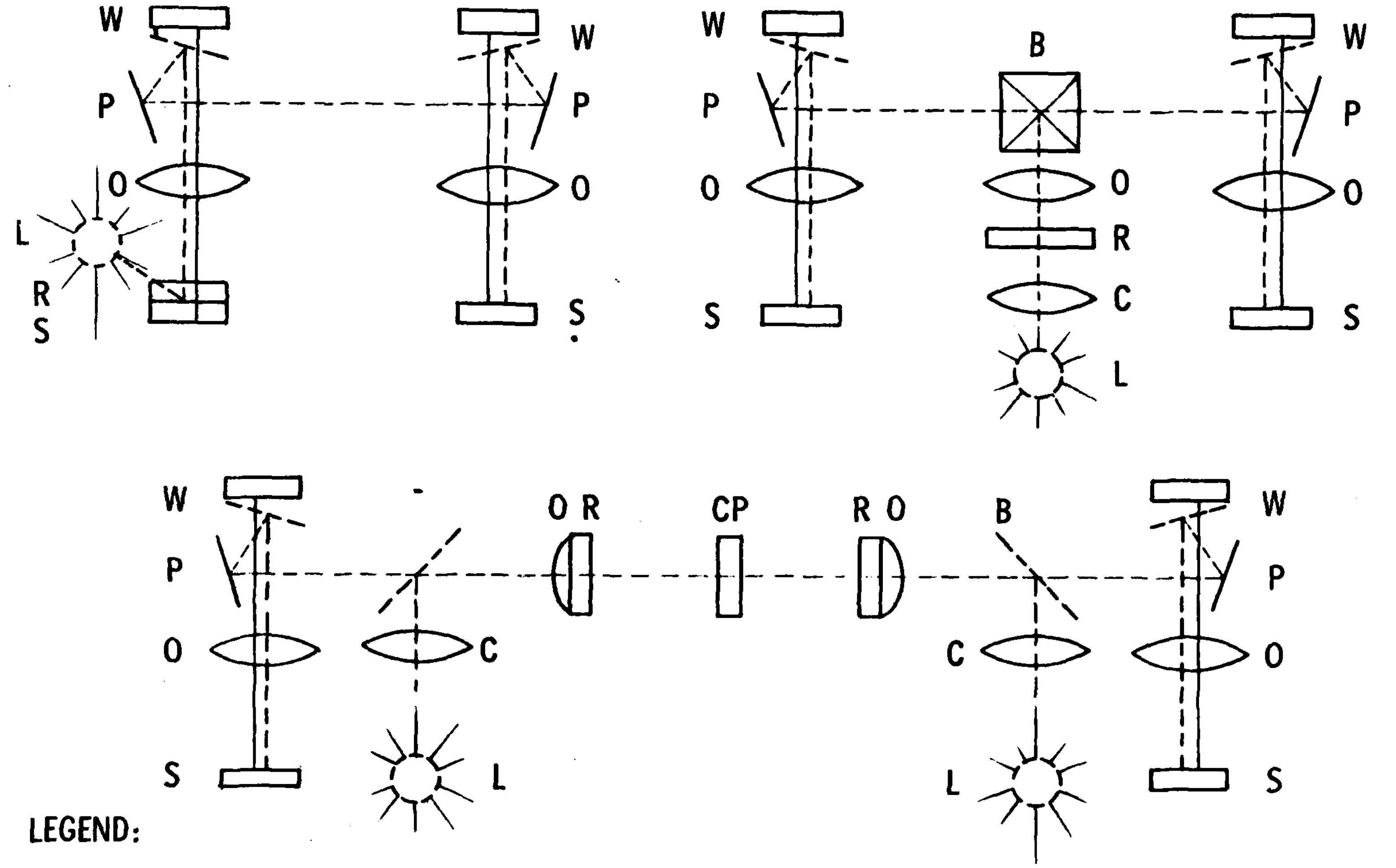


FIGURE B.1

218–23A

AUTOCALIBRATION TECHNIQUES



LEGEND:

B - BEAMSPLITTER
C - CONDENSER
L - LAMP

O - OBJECTIVE
P - PENTAREFLECTOR
(WITH SPECTRAL FILTER FACET)

R - RETICLE
S - SENSOR
W - WINDOW

CP - CORRECTOR PLATE

FIGURE B.2

For terrain avoidance, it may be initially assumed that absolute ranging is needed. If it is required, there are several options. First, the necessary autocalibration could be included in the system design. Second, a separate, active rangefinder could provide the absolute range to a point in the scene which would allow scaling the entire relative map. In this way, use of an active sensor can be kept to a minimum with an occasional random flash. Third, the absolute range could be roughly calculated by measuring the angular rate of some off-axis portion of the scene. The formula for range parallel to the line of flight (R_{LOF}) is given by:

$$R_{LOF} = \frac{V\ (T - T')}{\frac{P_T}{P_{T'}} - 1}$$

where:

V = platform velocity

P_T = image location of an object at time T

T = current time

T' = time at which the image location of an object was previously noted

There are several potential error sources in the latter option of range calculation, but over a period of time and over several image points, the accuracy should be acceptable. In fact, if a human is the sole user of the relative range map, it is possible that observation of the angular rates on display would be sufficient to estimate absolute range to objects and take appropriate action. These three options are not mutually exclusive and may be used in combination to obtain greater accuracy than any one alone. With the cues given by angular rate, it is not clear that the explicit calculation of absolute range is definitely required for terrian avoidance.

LIMITATIONS

Because PPC implements only the correlation curve interpolation, the correlation curve itself is never calculated. This fact has several implications. First, there is no checking for degree of match. It must then be assumed that the two input video signals represent a matched pair. Similarly, the peak sample is not located explicitly. It is assumed that the peak sample is within one or two sample spacings of some nominal value. PPC calculates only the interpolated peak position. The results are valid strictly if the actual misregistration is within ±1 sample spacing of nominal. In fact, the algorithm will perform nearly as well if the misregistration is within ±2 sample spacings of nominal, but larger misregistrations up to about 3 or 4 sample spacings will be interpreted as 1 or 0. Larger misregistration will produce results which vary about the nominal. However, if the misregistration can be expected to vary smoothly as in a tracking, a down-looking or a range-tracking application, as opposed to discontinuously as in a forward-looking application, any degree of misregistration can be accommodated by delaying one of the video signals relative to the other. Assuming that the appropriate relative delay is initially set, whenever a misregistration equal to 1 sample spacing is calculated, a delay is introduced so that the calculated result is zero. Thus, so long as the misregistration varies somewhat smoothly, continued high-accurate correlation is possible. Similarly, if the alignment between two ranging sensors is not well maintained, delays can be incororated to account for expected variations.

To reiterate, the limitations of PPC described so far are that the two signals must be known a priori to match and that expected misregistrations are either limited or smoothly varying. If a sufficiently large and complex system were allowable, even the latter limitation could be avoided. Generally, a fixed rectangular window is chosen, but the only real requirement is for the window to have flat horizontal edges and that the horizontal width be constant although the vertical edges may curve and the window may even be a composite of several windows. One final limitation is that normalization is not performed so the two input signals should be well matched in dc level and gain.

CALCULATION RATE

With a pipeline implementation, the rates at which correlation results may be produced is limited by the slowest single operation. For this algorithm, the slowest operation sets a limit to 8 million correlations per second with commercial chips and chip sets. This corresponds to maximum input video bandwidths of 4 MHz. Alternatively, CAI has developed an algorithm to perform normally slow arithmetic functions at rates up to 100 million operations per second. This is another pipeline implementation and, a such, is most useful when a steady steam of high-rate calculations are required. With such special electronics, PPC can be performed with input and output bandwidths of 50 MHz.

APPENDIX C
COMPUTER SIMULATION PROGRAM

INTRODUCTION

A considerable portion of the effort expended under this program went to the development and execution of a computer program to simulate pipeline precision correlation. This appendix describes the program, presents listings of the program and results, and analyzes the results.

It is important to note that simulation programs are inherently suspect because it is impossible to model all possible aspects of a system without actually building the system. The programmer must make some sort of evaluation of what factors are important, what details of those factors are important and how to most effectively model them, considering both accuracy and program size and speed.

This program, ir addition to the general problem of simulation, involves two other difficulties. First, processing architecture of a computer is not well suited to simulate a pipeline processing architecture. Second, the simulation involves performing calculations on scenes. Just defining adequate descripters of scenes is the subject of a vast amount of research. Actually, generating a reasonable facsimile of a scene in general is, then, a premature undertaking. There is, however, no alternative but to make the attempt and decide subjectively if the results are "scene like". The results of any process dependent on such artificial scenes can only be considered a general indication, since there is no way to prove that the scenes could be real scenes, let alone real scenes likely to be encountered.

The sole justification of results from this simulation program is that they basically agree with results obtained from a hardware demonstration unit which implemented a more standard "one target window" correlator, rather than a pipeline correlator.

This simulation program was also a design process to determine the values which several variable parameters, such as window size and digitization level (number of bits), shold take in the preliminary hardware design. The factors against which precision pipeline correlation were tested included actual image shift and signal-to-noise ratio. Other factors such as lens MTF, high-frequency vibration and other image degradations were not tested. These factors can only be adequately modeled with transform manipulations and were not considered cost effective for the purposes of this study.

The program was initially written with very broad calculations in the sense that many combinations of design parameters were calculated. At each stage, choices were made that eliminated several combinations. As the design choices were narrowed, the testing against image shift and noise was broadened. This approach does not guarantee that final choices for design parameters were optimal for all cases, but, as will be seen, the choices were either so clear-cut for limited cases that there is little likelihood of a bad choice or, towards the end with broader tests, so close as to make little difference.

DESCRIPTION

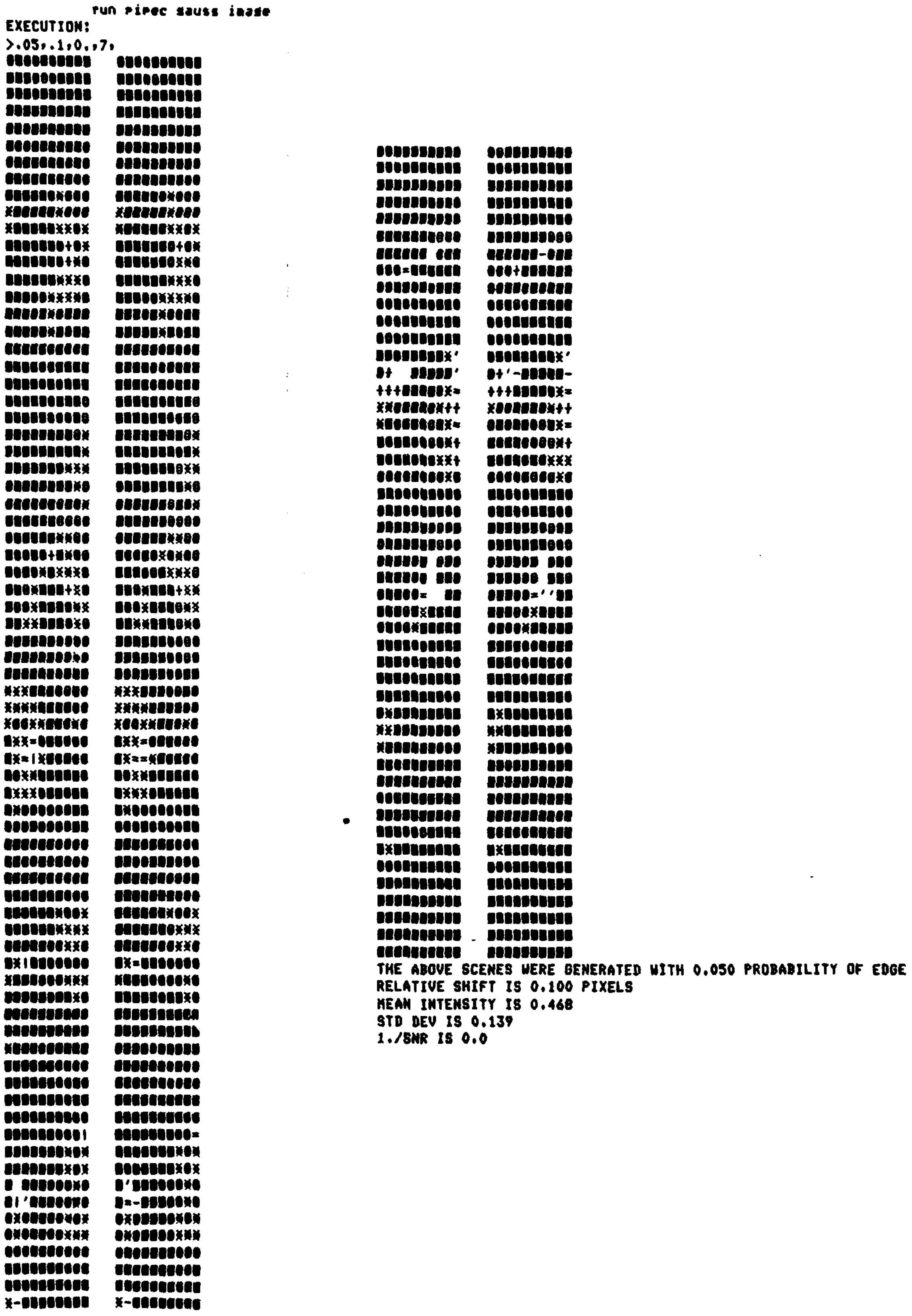
The program consists of two sections. The first section generates imagery and the second operates on the imagery to simulate pipeline correlation.

CAI has in its library a program to generate large images which includes the capability to superimpose targets on background with horizons, define ground and sky clutter statistics, set independent motions for,target, ground and sky, apply MTF's and other filters and so forth. Because of its versatility, it is a large program and not well suited to the requirements of this study. Pipeline correlation by its nature requires continuous streams of video data. Hence, rather than generate scenery on a frame-by-frame basis, the approach taken was to generate scenery "on the fly". This approach requires a different set of image descriptors than he generation of a full frame. The primary descriptor is probability of edge, that is, the likelihood that a pixel intensity will be unlike any of its previously-calculated neighbors. Neglecting initiation and edge effects, each pixel intensity being generated will have had four of its neighbors previously generated. The generation of scenes proceeds in the following manner. First , a uniformly-distributed random number is used to decide if the current pixel will be an edge. If so, the average of its four previously generated neighbors is taken and the new pixel is uniformly allowed to be any value outside a tolerance of this average, but between 0 and 1. Other distributions such as an agaussian (an upside-down gaussian curve) led to intractable mathematics. If, on the other hand, the current pixel is not to be an edge and therefore like one or more of its neighbors, the next step is to decide what combination of neighbors it is to be like. Again, a uniform random variable detrmines how many neighbors it will be similar to: one, two, three or four. If four, no further decisions are required. If one or three, yet another uniform random number determines which of the four is used or excluded, respectively. Finally, if two, a uniform random number determines which pair. Once the combination of pixels to be used has been determined, the average of their intensities is calculated. Using this average as the mean and a linear function of the number of pixels as the standard deviation, a random number of a gaussian distribution is calculated. The random number is used as the intensity of the current pixel. This procedure is reasonably efficient in calculation. Its purpose is to allow small gray shade differences, as well as edges, to propagate randomly in any direction.

As one scene is being generated, a shifted version of it must also be generated. The second version is produced by calculating a weighted (depending on shift) average of two neighboring pixels in the fist version. High-frequency effects are thought to be adequately considered by this method. It may appear that the averaging method avoids high-frequency effects. If it is examined in reverse, however, as though the first were derived from the second versions, it would seem that high-frequency effects are continuously generated; the version "one" pixels rarely, if ever, being an average of version "two" pixels.

Typical images generated by this procedure as shown in fight C.1 and C.2. The latter shows several copies of the original at varying shifts.

While the images are being generated, their statistical standard deviations are calculated. These values are used to calculate a noise term added to each pixel and yielding the desired signal-to-noise ratio.



```
          run pipec gauss image
EXECUTION:
>.05,.1,0.,7,
```

```
THE ABOVE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.100 PIXELS
MEAN INTENSITY IS 0.468
STD DEV IS 0.139
1./SNR IS 0.0
```

FIGURE C.1 SAMPLE IMAGE

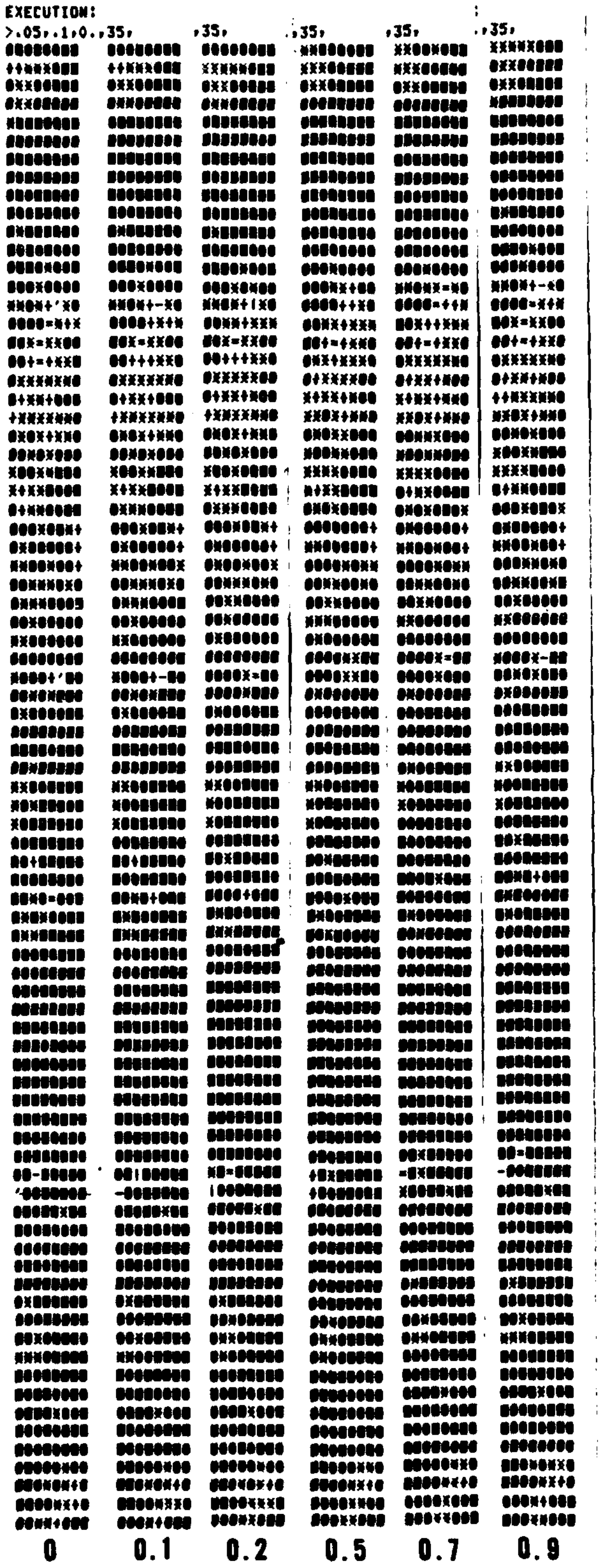


FIGURE C.2 SAMPLE IMAGE WITH VARIOUS SHIFTS

The second section of the program is the actual simulation of pipeline correlation. This is a fairly straightforward mathematical procedure. To simulate a pipeline architecture, numerous partial results must be stored in files. To compare design parameters, the most efficient calculation method is to execute the program a single time and only calculate alternative partial or final results at the appropriate point in the operation. For example, all results use the same scene data, so the scene is generated once. The first differentiation occurs immediately after the scene generation stage where loops are set up to test different terms in the precision correlation algorihm. Within this loop is a loop of different digitization levels of the video to be tested. There is then a loop to test different window sizes. Finally, after the results for all these combinations have been calculated, is a loop to test for different lengths of a running average to smooth the results somewhat and improve the accuracy of the final output. As can be seen from the many combinations possible, files storing partial results can be quite large. All of these loops are executed for each new line of video generated. Over many lines, the statistics of final results are calculated and these statistics are printed for each combination at the end of program execution. During the study, as decisions were made, possible combinations were eliminated. Both the program and files became more tractible, allowing more extensive analysis of the effects of noise and image shift. At each decision point, the program was modified to more efficiently calculate the remaining options.

As with any development, during modification to optimize the program, new and more efficient calculation procedures were conceived and implemented so the modifications were not always limited to deleting existing code. Also, errors were found which for the most part had minimal effect.' Those results which were, or may have been affected by the errors not detected until later are marked as suspect. In general, it was not worthwhile backtracking and rerunning such executions as the errors did not affect the conclusions.

Listings of the major versions of this program will be found at the end of this appendix. The first listing includes a few utitlity subroutines such as a gaussian distribution random number generator and an image printing routine.

RESULTS

Figures C.3 through C.5 are graphs of some of the later results whe most of the design parameters had been decided. Figure C.3 shows RMS error in fractions of a pixel, versus window size. Not surprisingly, larger window sizes provided better performance, although the ratio of improvement is not as predictable as may have been expected. The crossovers and general nonlineraity of improvement is probably due to the high-frequency effects mentioned previously, where large windows may include some scene information which causes loss of accuracy not seen by a smaller window. Figure C.4 shows the effects of digitization level on rms error. Clearly, at 4 bits, there is a severe breakpoint. Binary 1-bit correlation is even worse, as will be illustrated below. Because 4 bits is a breakpoint, and thus some what dangerous to use, the design effort used 8 bits of digitization. Some fairly strong performance is shown in figure C.5, where error versus signal-to-noise ratio is graphed. Note that even at signal-to-noise ratio of 2.5:1 the performance is still adequate for stabilization.

There were essentially five stages of analaysis in the simulation study. However, before presenting and analyzing the results, some clarifications should be made. Initially, three different sets of terms were used in the precision correlation algorithm to attempt to determine the optimum set. However, the sets chosen exhibited a preference for shift in one direction. Therefore, results for positive and negative shifts were calculated for each set of terms. The two cases are referred to as "Type 1" and "Type 2" on the results printouts. Because a relatively wide diffrence existed, during the first two stages of program results, the third and fourth stages calculated results both ways, and determined which shift direction was the more likely, based on partial results. In the fifth stage, a set of terms that had been worked our to be independent of shift direction were used and shown to exhibit quite good performance.

The results printouts present four statistics for each combination of parameters. The statistics are, in order: the average error, the rms error, the maximum error during the trials, and the standard deviation of error. For the first two stages, however, the average is not divided by the number of trials and thus is the sum of errors, and the second statistic is neither divided by the number of trials nor is the square root taken and, hence, this statistic is the sum of the

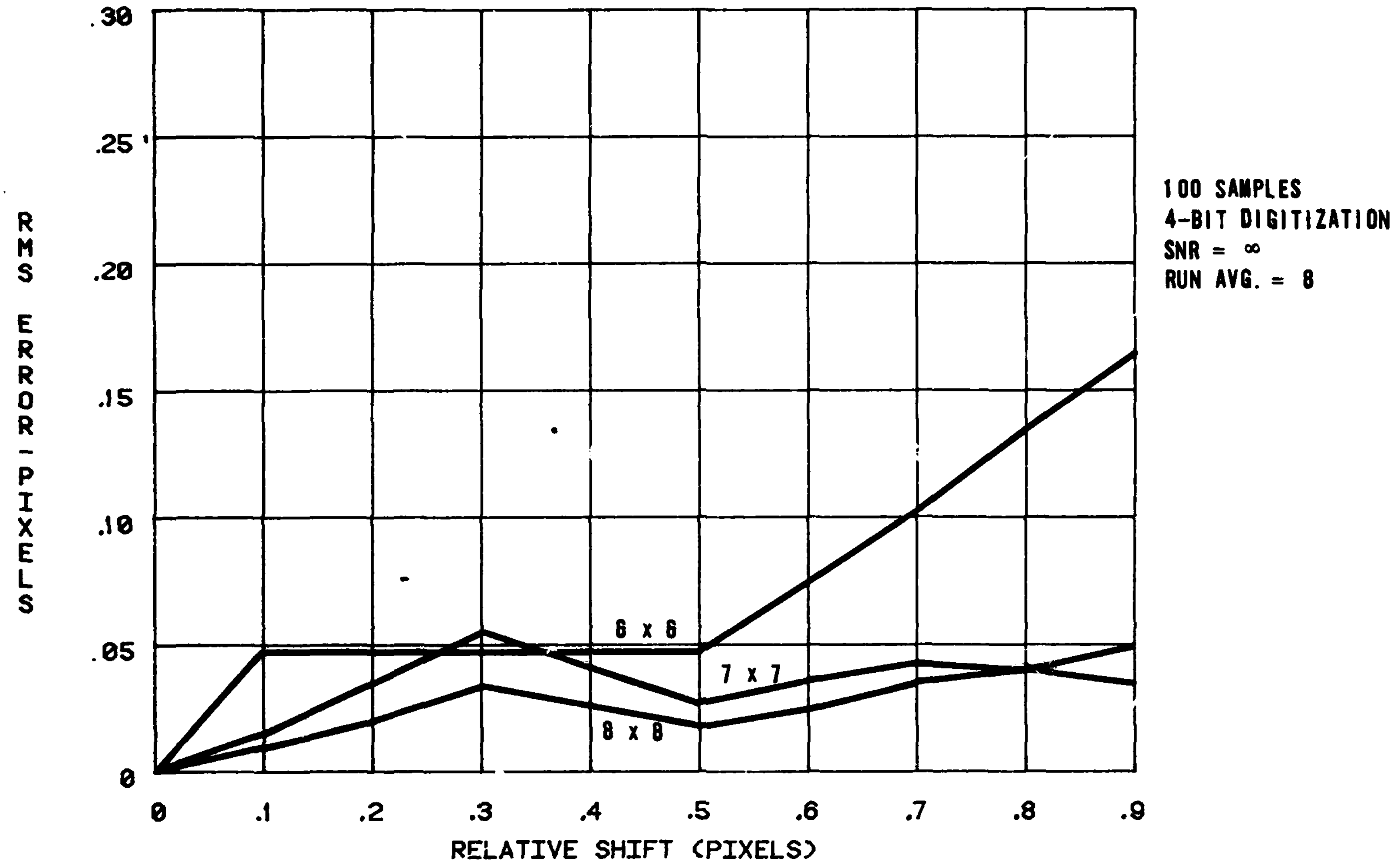


FIGURE C.3 RMS ERROR VS SHIFT AND WINDOW SIZE

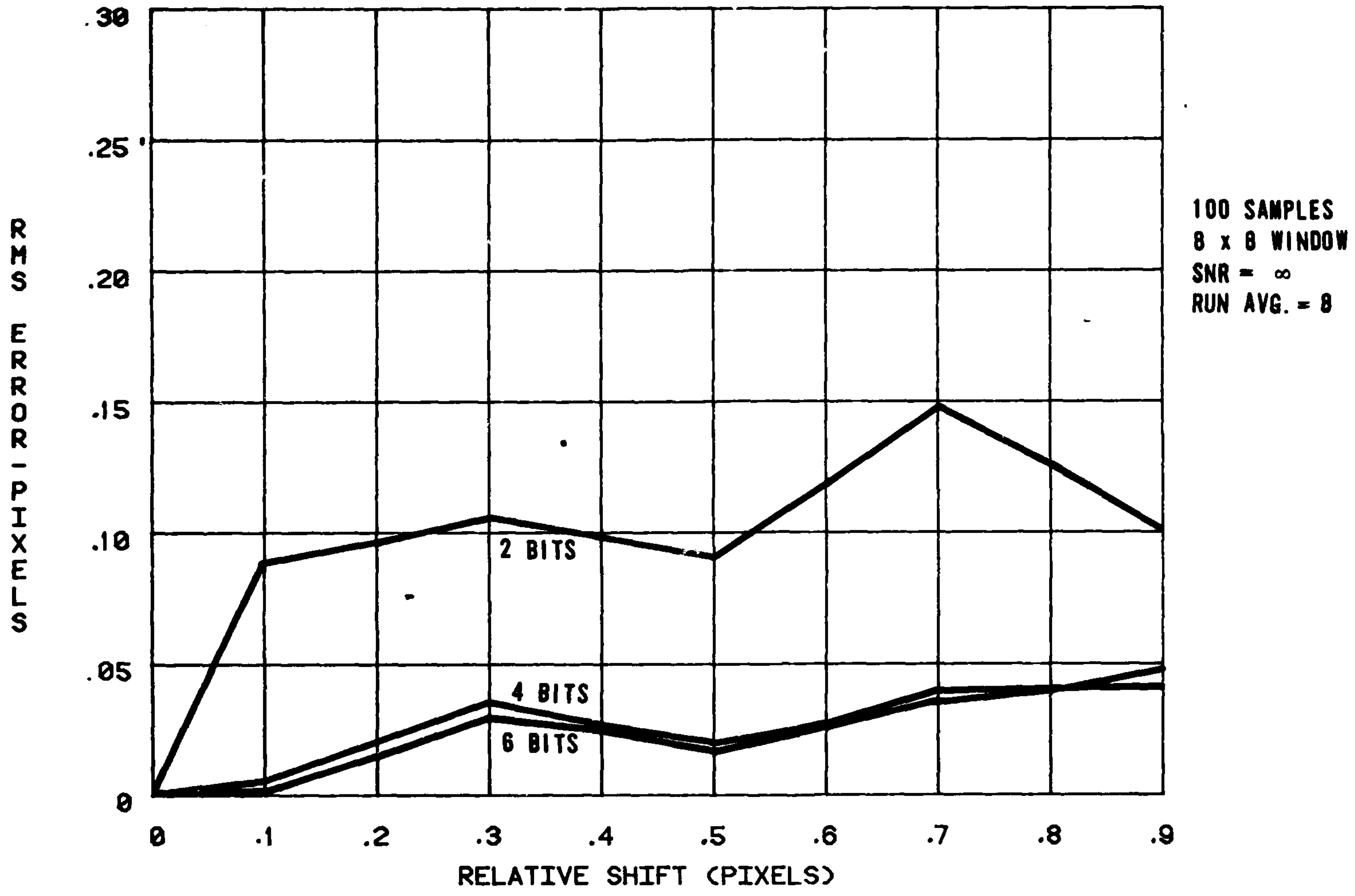


FIGURE C.4 RMS ERROR VS SHIFT AND DIGITIZATION LEVEL.

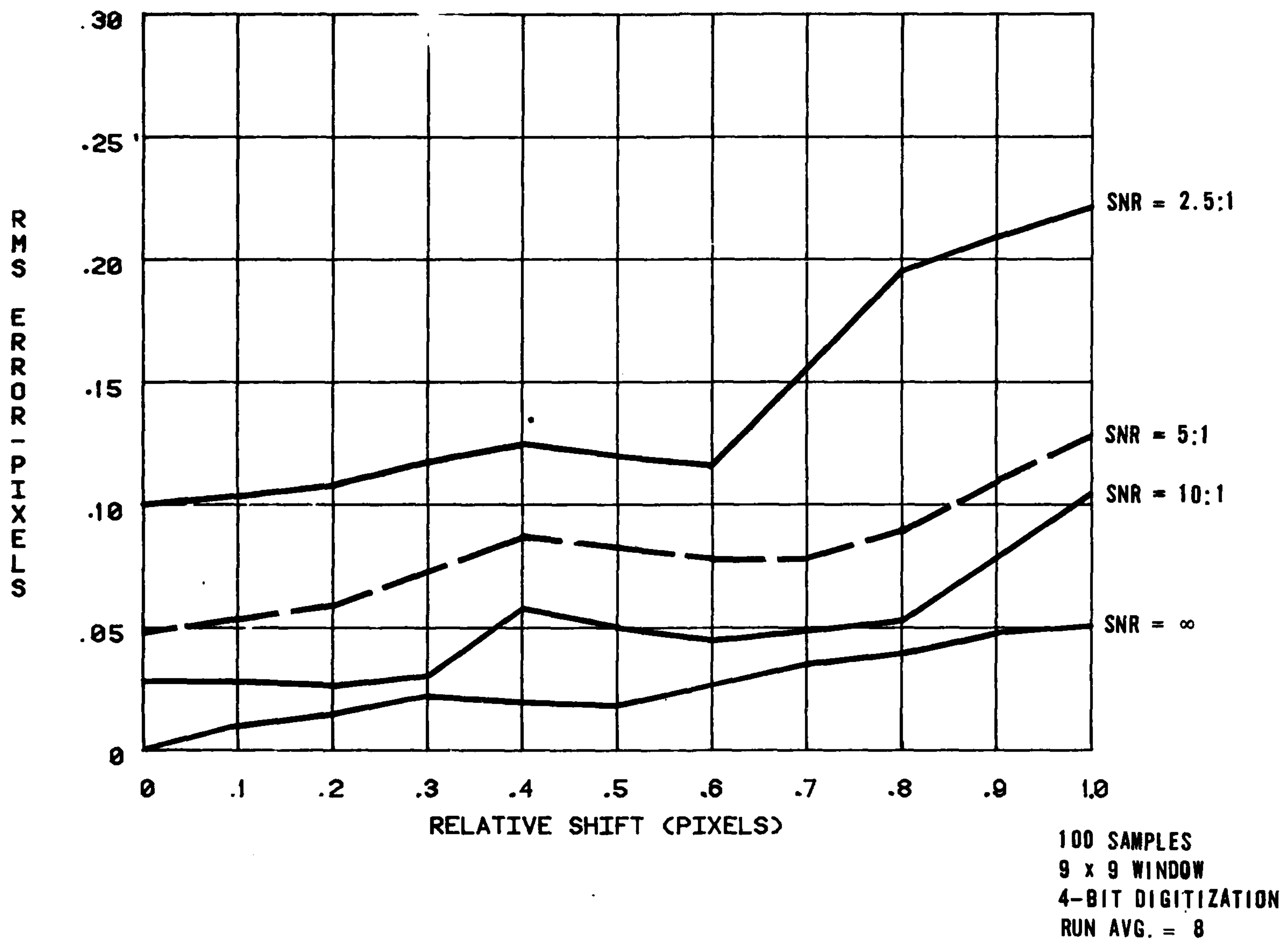


FIGURE C.5 RMS ERROR VS SHIFT AND NOISE

squares of errors. The purpose of leaving the statistics as sums is to allow their combined statistics to be more easily calculated.

Obviously bad results, those indicating shifts greater than 5 pixels, were artificially set to five. The integer at the end of each line is the number of trials considered valid. A minor program error actually added 7 or 8 to this number, as indicated.

At the first stage of analysis (figure C.6), the second and third sets of algorithm terms were eliminated as showing poorer performance relative to the first set. In terms of digitization, the low counts of valid trials for 1-bit digitization, leads to the conclusion the binary correlation is a poor choice. It is also noted that windows larger than 8 x 8 provide only minimal improvements in performance. Finally, it is clear from these first stage results that smoothing by taking a running average of results offers reasonable improvements in performance. All of this analysis is based upon images with 0.1 pixels of relative shift between them and no noise imposed on the video signal.

The second stage maintains the zero noise condition. Only the first set of algorithm terms is tested with a more limited set of window sizes and digitization levels, and the smoothing running average length is fixed at eight. Conversely, the number of shift positions is expanded to include relative shifts of 0.1, 0.2, 0.3, 0.7 and 0.9. Note that the first and last statistics, sum of errors and standard deviation, for the 6 x 6 windows in the first three blocks are suspect because of an error detected in a later stage. The conclusion fom this stage of analysis is that the preference of the set of terms for one shift direction is intolerable and other terms will have to be defined which are shift-direction independent. Further, 2-bit digitization is unacceptable with windows less than 8 x 8, and 4-bit digitization is only marginally acceptable.

At the third stage, the technique of calculating both cases of shift direction and choosing the most likely is tested with the same parameters as for the second stage. Note that the first two statistics are now proper mean error and rms error. Results are considerably improved. Although even a 6 x 6 window with 2-bit digitization produces acceptable results for stabilization, noise has still not been considered. Therefore, larger windows and greater resolution is still preferred.

STAGE I.

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 ! -0.942  3.304 -0.882  0.175 ! -5.953  1.961 -0.528  0.123 ! -6.484  1.229 -0.400  0.088 ! -6.894  0.961 -0.240  0.069 ! 106
 7X 7 !  1.020  1.894  0.742  0.132 ! -3.938  0.938  0.339  0.086 ! -6.521  0.795 -0.207  0.061 ! -6.294  0.565 -0.154  0.043 ! 107
 8X 8 !  3.683  3.427  1.127  0.175 ! -1.344  1.439  0.528  0.115 ! -4.304  0.720  0.201  0.071 ! -2.629  0.249 -0.104  0.041 ! 108
 9X 9 ! -1.449  3.195 -1.613  0.171 ! -6.492  2.057 -0.927  0.124 ! -7.710  1.473 -0.535  0.092 ! -6.466  0.860 -0.288  0.066 ! 108
10X10 !  1.083  0.351  0.322  0.056 ! -3.970  0.341  0.143  0.043 ! -4.594  0.280 -0.097  0.028 ! -3.811  0.166 -0.069  0.017 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 !  0.133  0.069  0.255  0.025 ! -4.918  0.272  0.077  0.021 ! -6.075  0.379 -0.071  0.019 ! -5.461  0.301 -0.062  0.015 ! 108
 7X 7 !  0.012  0.001  0.019  0.003 ! -5.044  0.253 -0.058  0.013 ! -6.120  0.371 -0.070  0.015 ! -5.474  0.297 -0.060  0.013 ! 108
 8X 8 ! -0.931  0.538 -0.690  0.070 ! -5.985  0.630 -0.421  0.053 ! -7.050  0.629 -0.250  0.040 ! -6.588  0.496 -0.152  0.029 ! 108
 9X 9 ! -0.008  0.001 -0.017  0.003 ! -5.058  0.254 -0.060  0.013 ! -6.274  0.390 -0.069  0.015 ! -5.783  0.331 -0.061  0.014 ! 108
10X10 ! -0.025  0.001  0.009  0.002 ! -5.077  0.256 -0.056  0.013 ! -6.290  0.392 -0.067  0.015 ! -5.801  0.333 -0.061  0.014 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 ! -1.013  7.176  1.232  0.258 ! -6.009  4.466  0.679  0.196 ! -7.181  2.583  0.382  0.140 ! -7.517  1.164 -0.208  0.077 ! 107
 7X 7 !  4.292 17.569  1.679  0.401 ! -0.674  9.185  1.130  0.292 ! -3.218  5.147  0.765  0.216 ! -1.967  2.362  0.448  0.147 ! 106
 8X 8 ! -0.424 11.318 -2.087  0.324 ! -5.466  6.202 -1.113  0.234 ! -8.210  3.461 -0.624  0.162 ! -6.701  1.659 -0.349  0.107 ! 108
 9X 9 !  3.837  7.078  1.516  0.254 ! -1.223  3.542  0.741  0.181 ! -1.958  1.726  0.364  0.125 ! -0.943  0.733 -0.169  0.082 ! 108
10X10 ! -0.384  7.060 -1.528  0.256 ! -5.444  3.991 -0.910  0.186 ! -5.857  2.312 -0.530  0.136 ! -5.593  1.367 -0.331  0.100 ! 106
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 !  0.095  0.006  0.058  0.008 ! -4.958  0.246 -0.059  0.013 ! -6.060  0.365 -0.068  0.015 ! -5.481  0.298 -0.062  0.014 ! 108
 7X 7 !  0.051  0.010  0.064  0.009 ! -5.006  0.253 -0.068  0.014 ! -6.018  0.361 -0.071  0.015 ! -5.390  0.289 -0.060  0.013 ! 108
 8X 8 ! -0.047  0.011 -0.045  0.010 ! -5.100  0.264 -0.074  0.015 ! -6.123  0.376 -0.074  0.016 ! -5.656  0.319 -0.068  0.015 ! 108
 9X 9 ! -0.050  0.012  0.076  0.010 ! -5.100  0.263 -0.077  0.014 ! -6.279  0.393 -0.077  0.016 ! -5.785  0.333 -0.066  0.015 ! 108
10X10 ! -0.057  0.155  0.302  0.038 ! -5.108  0.317 -0.155  0.026 ! -6.304  0.420 -0.123  0.022 ! -5.810  0.347 -0.089  0.018 ! 107
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 ! -5.041 13.872 -1.530  0.355 !-10.063  8.828 -1.013  0.270 !-10.983  5.198 -0.851  0.194 !-12.594  2.912 -0.495  0.116 ! 107
 7X 7 ! -2.179  4.328  1.365  0.199 ! -7.154  2.476  0.677  0.136 ! -9.232  1.788 -0.315  0.096 ! -8.092  1.146 -0.217  0.071 ! 102
 8X 8 !  3.266  6.503  1.542  0.244 ! -1.782  2.962  0.829  0.165 ! -3.156  1.377  0.392  0.109 ! -2.445  0.493 -0.161  0.064 ! 107
 9X 9 !  0.878 11.177  1.809  0.322 ! -4.197  5.397  0.817  0.220 ! -4.471  2.725  0.717  0.153 ! -3.972  1.375 -0.330  0.107 ! 107
10X10 ! -2.273  6.450 -1.318  0.243 ! -7.312  3.669 -0.759  0.171 ! -7.813  2.066 -0.424  0.118 ! -7.382  1.242 -0.279  0.083 ! 106
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO 10 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                            2                                  4                                  8
WINDOW
 6X 6 ! -0.390  0.066 -0.194  0.024 ! -5.441  0.320 -0.151  0.021 ! -6.501  0.429 -0.110  0.019 ! -5.943  0.354 -0.082  0.016 ! 108
 7X 7 ! -0.677  0.416 -0.628  0.062 ! -5.734  0.503 -0.369  0.043 ! -6.711  0.527 -0.226  0.032 ! -6.225  0.423 -0.139  0.024 ! 108
 8X 8 ! -0.035  0.103  0.288  0.031 ! -5.088  0.308 -0.101  0.025 ! -6.192  0.404 -0.088  0.021 ! -5.720  0.336 -0.073  0.018 ! 107
 9X 9 !  0.084  0.033  0.150  0.017 ! -4.966  0.262 -0.079  0.018 ! -6.169  0.386 -0.079  0.018 ! -5.690  0.325 -0.066  0.015 ! 108
10X10 ! -0.211  0.030 -0.112  0.017 ! -5.263  0.289 -0.108  0.017 ! -6.470  0.422 -0.092  0.018 ! -5.975  0.357 -0.074  0.016 ! 107
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -1.009  3.615 -0.905  0.183 ! -6.009  2.128 -0.538  0.129 ! -6.482  1.298 -0.397  0.092 ! -6.731  0.952 -0.241  0.070 ! 106
 7X 7 !  0.952  1.845  0.740  0.130 ! -4.009  0.924  0.341  0.085 ! -6.225  0.742 -0.205  0.060 ! -5.654  0.485 -0.149  0.042 ! 107
 8X 8 !  3.602  3.339  1.095  0.173 ! -1.422  1.484  0.519  0.116 ! -4.086  0.734  0.220  0.073 ! -2.267  0.261  0.095  0.044 ! 108
 9X 9 ! -1.397  3.058 -1.567  0.168 ! -6.441  1.977 -0.897  0.121 ! -7.468  1.394 -0.518  0.090 ! -6.233  0.810 -0.279  0.065 ! 108
10X10 !  1.019  0.367  0.337  0.058 ! -4.033  0.360  0.155  0.044 ! -4.405  0.269 -0.097  0.029 ! -3.711  0.160 -0.067  0.017 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 !  1.884  3.415  1.832  0.177 ! -3.163  1.825  0.900  0.127 ! -4.121  1.031  0.434  0.090 ! -3.428  0.526  0.195  0.062 ! 108
 7X 7 !  0.032  0.020  0.099  0.014 ! -5.020  0.263 -0.082  0.016 ! -5.873  0.351 -0.088  0.017 ! -4.620  0.217 -0.062  0.013 ! 108
 8X 8 !  0.918  0.676  0.717  0.079 ! -4.138  0.596  0.455  0.064 ! -4.822  0.452  0.193  0.047 ! -4.300  0.287 -0.084  0.033 ! 108
 9X 9 ! -0.043  0.014  0.085  0.011 ! -5.090  0.263 -0.068  0.015 ! -5.976  0.357 -0.069  0.015 ! -5.606  0.313 -0.062  0.014 ! 108
10X10 ! -0.146  0.004  0.022  0.006 ! -5.190  0.269 -0.066  0.014 ! -6.289  0.393 -0.071  0.016 ! -5.833  0.338 -0.063  0.015 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -0.581  8.382  1.573  0.279 ! -5.570  5.152  0.870  0.212 ! -6.477  2.853  0.491  0.151 ! -6.479  1.135 -0.211  0.083 ! 107
 7X 7 !  4.407 18.212  1.789  0.409 ! -0.567  9.728  1.358  0.300 ! -2.524  5.427  0.724  0.223 ! -0.657  2.459  0.400  0.151 ! 106
 8X 8 ! -0.227 11.187 -2.051  0.322 ! -5.270  6.121 -1.098  0.233 ! -7.515  3.362 -0.607  0.162 ! -5.952  1.573 -0.339  0.107 ! 108
 9X 9 !  3.729  7.558  1.552  0.262 ! -1.326  3.748  0.765  0.186 ! -1.698  1.831  0.382  0.129 ! -0.728  0.796  0.178  0.086 ! 108
10X10 !  1.440  4.593 -1.173  0.206 ! -3.615  2.649 -0.770  0.153 ! -3.777  1.585 -0.442  0.116 ! -3.580  0.979 -0.315  0.089 ! 105
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 !  0.141  0.158  0.284  0.038 ! -4.911  0.326 -0.128  0.031 ! -5.827  0.375 -0.098  0.024 ! -5.043  0.269 -0.082  0.018 ! 107
 7X 7 !  0.056  0.815  0.686  0.087 ! -5.008  0.694  0.388  0.065 ! -5.538  0.545  0.167  0.049 ! -4.562  0.325 -0.122  0.035 ! 108
 8X 8 !  0.249  0.139  0.178  0.036 ! -4.806  0.317 -0.143  0.031 ! -5.292  0.335 -0.102  0.026 ! -4.921  0.269 -0.074  0.020 ! 108
 9X 9 ! -0.129  0.090 -0.195  0.029 ! -5.176  0.287 -0.135  0.019 ! -6.132  0.384 -0.101  0.018 ! -5.736  0.331 -0.075  0.015 ! 108
10X10 ! -1.304  2.337 -1.512  0.147 ! -6.353  1.416 -0.785  0.098 ! -7.499  1.036 -0.427  0.069 ! -7.016  0.718 -0.241  0.049 ! 107
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -3.204 13.690  1.667  0.355 ! -8.216  7.764 -0.981  0.257 ! -8.925  4.552 -0.822  0.188 ! -9.885  2.342 -0.476  0.115 ! 106
 7X 7 ! -0.688  7.323  1.653  0.260 ! -5.681  3.756  0.739  0.179 ! -6.947  2.101  0.376  0.124 ! -5.551  1.153 -0.217  0.090 ! 103
 8X 8 !  4.505  8.334  1.485  0.275 ! -0.542  3.671  0.810  0.184 ! -1.404  1.703  0.390  0.125 ! -0.764  0.587 -0.163  0.073 ! 108
 9X 9 !  0.562 11.372  1.528  0.324 ! -4.504  5.569 -0.740  0.223 ! -4.625  2.570  0.667  0.148 ! -4.076  1.197 -0.312  0.098 ! 107
10X10 ! -3.429  9.130 -1.594  0.289 ! -8.468  5.338 -0.961  0.208 ! -8.811  2.894 -0.525  0.142 ! -8.239  1.654 -0.303  0.097 ! 107
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  8 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -0.245  1.340 -0.976  0.111 ! -5.295  0.867 -0.548  0.075 ! -6.056  0.638 -0.303  0.053 ! -5.286  0.408 -0.172  0.037 ! 108
 7X 7 ! -2.095  3.320 -1.587  0.174 ! -7.160  2.029 -0.868  0.120 ! -7.358  1.269 -0.501  0.084 ! -6.793  0.815 -0.274  0.060 ! 108
 8X 8 !  0.884  0.610  0.615  0.075 ! -4.164  0.478  0.279  0.054 ! -4.908  0.380  0.108  0.038 ! -4.465  0.253 -0.085  0.025 ! 108
 9X 9 !  0.425  0.237  0.385  0.047 ! -4.625  0.335  0.136  0.036 ! -5.718  0.388 -0.098  0.028 ! -5.350  0.314 -0.078  0.021 ! 108
10X10 ! -0.275  0.298 -0.447  0.052 ! -5.328  0.425 -0.275  0.039 ! -6.413  0.473 -0.172  0.029 ! -5.943  0.382 -0.114  0.023 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 !  2.050  9.204  1.597  0.291 ! -2.944  4.085  0.719  0.193 ! -3.465  2.008 -0.427  0.133 ! -3.393  1.134 -0.248  0.098 ! 108
 7X 7 !  1.964  2.673  0.806  0.156 ! -2.979  1.072 -0.363  0.096 ! -5.273  0.618 -0.222  0.058 ! -3.104  0.219 -0.134  0.035 ! 108
 8X 8 !  2.612  3.753 -1.177  0.185 ! -2.409  1.766 -0.694  0.126 ! -4.843  0.950 -0.360  0.082 ! -2.295  0.385 -0.211  0.056 ! 108
 9X 9 ! -1.561  4.665 -1.966  0.207 ! -6.618  2.869 -1.096  0.151 ! -6.337  1.740 -0.608  0.113 ! -4.989  0.937 -0.321  0.081 ! 108
10X10 !  1.408  0.513  0.358  0.068 ! -3.646  0.439  0.221  0.054 ! -3.241  0.252 -0.100  0.038 ! -2.446  0.114 -0.067  0.023 ! 108


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 !  0.858  1.185 -0.683  0.104 ! -4.168  0.748 -0.386  0.074 ! -5.115  0.503 -0.235  0.049 ! -3.117  0.188 -0.128  0.030 ! 108
 7X 7 !  0.400  0.187  0.263  0.041 ! -4.647  0.339 -0.135  0.036 ! -5.280  0.387 -0.178  0.035 ! -0.795  0.074 -0.085  0.025 ! 108
 8X 8 !  1.445  0.576  0.462  0.072 ! -3.647  0.481  0.383  0.058 ! -2.993  0.270  0.167  0.042 ! -2.337  0.135 -0.067  0.028 ! 107
 9X 9 !  0.201  0.066  0.112  0.025 ! -4.858  0.282 -0.111  0.024 ! -4.764  0.257 -0.092  0.021 ! -4.566  0.223 -0.067  0.017 ! 108
10X10 !  0.053  0.081  0.102  0.027 ! -4.999  0.302 -0.123  0.026 ! -5.913  0.379 -0.103  0.023 ! -5.525  0.318 -0.079  0.018 ! 108


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 ! -1.505  8.182  1.096  0.275 ! -6.457  4.841 -0.572  0.203 ! -7.793  2.636 -0.364  0.139 ! -7.453  1.119 -0.226  0.075 ! 107
 7X 7 !  1.992 17.243 -1.776  0.399 ! -2.941  8.417 -0.927  0.278 ! -5.234  4.625 -0.686  0.201 ! -1.442  1.517  0.304  0.118 ! 106
 8X 8 !  3.094  7.029 -1.191  0.253 ! -1.938  3.551 -0.682  0.180 ! -4.550  2.175 -0.476  0.136 ! -2.845  0.968 -0.207  0.091 ! 107
 9X 9 !  4.906  9.557  1.526  0.294 ! -0.139  4.162  0.736  0.196 !  0.352  1.624  0.361  0.123 !  0.985  0.598  0.168  0.074 ! 108
10X10 !  2.758 10.774  1.819  0.315 ! -2.282  5.800  1.103  0.231 ! -2.095  2.973  0.597  0.165 ! -2.289  1.763 -0.425  0.126 ! 106


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 !  0.614  0.999  0.500  0.096 ! -4.426  0.730  0.294  0.071 ! -4.950  0.507  0.132  0.051 ! -2.939  0.211 -0.154  0.035 ! 107
 7X 7 ! -0.239  2.856 -0.979  0.163 ! -5.344  1.751 -0.549  0.117 ! -4.061  0.984  0.321  0.088 ! -1.867  0.416  0.180  0.060 ! 107
 8X 8 !  1.708  1.070  0.503  0.098 ! -3.376  0.671  0.293  0.072 ! -1.653  0.395  0.176  0.059 ! -1.716  0.231  0.094  0.043 ! 108
 9X 9 !  0.470  1.442 -0.581  0.115 ! -4.579  0.758 -0.305  0.072 ! -4.294  0.407 -0.194  0.047 ! -4.301  0.276 -0.120  0.031 ! 108
10X10 !  5.048  4.936  1.419  0.209 ! -0.001  3.497  1.305  0.180 ! -0.814  2.007  0.637  0.136 ! -0.551  0.964  0.294  0.094 ! 108


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 ! -7.949  8.554 -1.384  0.272 !-12.926  6.049 -0.827  0.204 !-13.577  4.432 -0.565  0.159 !-14.715  3.411 -0.368  0.114 ! 103
 7X 7 ! -2.385  5.343 -1.097  0.221 ! -7.385  3.234  0.734  0.139 ! -7.330  1.857  0.361  0.112 ! -5.695  1.113 -0.216  0.087 ! 102
 8X 8 ! -0.389  8.537 -1.997  0.281 ! -5.428  4.190 -1.089  0.190 ! -5.513  1.993 -0.587  0.126 ! -5.069  0.872 -0.275  0.077 ! 107
 9X 9 !  0.610  7.269  1.331  0.259 ! -4.423  3.674 -0.685  0.180 ! -3.692  1.514 -0.353  0.113 ! -4.116  0.918 -0.320  0.084 ! 107
10X10 ! -1.017  5.249  1.607  0.220 ! -6.027  2.701  0.653  0.148 ! -5.993  1.535  0.359  0.106 ! -6.295  0.955 -0.207  0.074 ! 105


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                           2                             4                             8
WINDOW
 6X 6 !  1.996  5.808  1.770  0.231 ! -2.998  2.963  0.842  0.163 ! -3.433  1.503  0.394  0.114 ! -2.290  0.718 -0.229  0.079 ! 105
 7X 7 !  0.166  4.739 -1.276  0.209 ! -4.888  2.758 -0.879  0.153 ! -4.003  1.521 -0.501  0.113 ! -3.488  0.774 -0.279  0.078 ! 108
 8X 8 !  0.526  3.484 -0.978  0.180 ! -4.505  1.866 -0.580  0.125 ! -4.781  1.167 -0.369  0.094 ! -4.890  0.764 -0.247  0.071 ! 108
 9X 9 ! -0.081  2.563  0.964  0.154 ! -5.101  1.557  0.518  0.110 ! -6.191  1.073 -0.400  0.082 ! -6.626  0.625 -0.231  0.045 ! 108
10X10 ! -2.000  2.550 -1.189  0.153 ! -7.053  1.774 -0.640  0.110 ! -8.527  1.401 -0.373  0.082 ! -8.188  0.942 -0.244  0.055 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -2.288 10.256  1.567  0.307 ! -7.352  5.321 -0.864  0.211 ! -7.360  2.795 -0.617  0.146 ! -7.151  1.713 -0.387  0.107 ! 105
 7X 7 ! -0.657  4.217  1.086  0.197 ! -5.685  2.754 -0.944  0.151 ! -6.851  1.615 -0.577  0.105 ! -3.366  0.684 -0.279  0.073 ! 106
 8X 8 !  3.205  3.241  1.150  0.171 ! -1.828  1.934  0.725  0.133 ! -3.402  1.076  0.371  0.095 !  1.081  0.456  0.189  0.064 ! 108
 9X 9 !  1.380  1.096  0.604  0.100 ! -3.682  0.786  0.382  0.078 ! -3.524  0.564  0.231  0.064 ! -0.501  0.264  0.166  0.049 ! 107
10X10 !  0.834  1.257  0.600  0.108 ! -4.269  0.954 -0.293  9.085 ! -3.267  0.593  0.229  0.068 ! -0.888  0.287  0.162  0.051 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -1.562  4.061 -1.322  0.193 ! -6.648  2.612 -0.738  0.143 ! -5.354  1.397 -0.394  0.102 ! -7.303  0.977 -0.237  0.067 ! 102
 7X 7 ! -0.833  1.562  0.606  0.120 ! -5.915  1.168 -0.297  0.088 ! -4.709  0.659 -0.199  0.065 ! -4.673  0.480 -0.183  0.051 ! 108
 8X 8 ! -1.045  2.596 -0.957  0.155 ! -6.214  1.679 -0.536  0.111 ! -2.696  0.698 -0.268  0.076 ! -1.041  0.295 -0.135  0.051 ! 107
 9X 9 ! -0.884  1.054 -0.656  0.098 ! -6.094  1.042 -0.389  0.080 ! -2.332  0.510 -0.235  0.065 ! -1.389  0.297 -0.136  0.051 ! 106
10X10 ! -0.723  0.629 -0.283  0.076 ! -5.882  0.815 -0.309  0.068 ! -3.404  0.442 -0.172  0.056 ! -2.350  0.260 -0.123  0.044 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -7.908 14.440 -1.475  0.358 !-12.927  8.740 -1.096  0.258 !-11.902  5.017 -0.620  0.185 !-16.875  4.479 -0.470  0.131 ! 105
 7X 7 ! -5.844 20.296 -1.957  0.430 !-10.870 10.945 -1.226  0.302 !-10.535  5.846 -0.692  0.211 ! -8.110  2.810 -0.425  0.143 ! 106
 8X 8 !  3.012 13.083 -1.633  0.347 ! -2.085  7.768 -0.906  0.267 ! -0.970  4.143 -0.509  0.196 !  1.350  1.985  0.278  0.135 ! 107
 9X 9 !  5.296 11.261 -1.767  0.319 !  0.184  5.467  1.175  0.225 !  1.547  2.473  0.592  0.151 !  3.451  1.241  0.349  0.102 ! 105
10X10 !  2.784  6.822 -1.174  0.250 ! -2.313  3.653  0.725  0.183 ! -0.388  2.189  0.497  0.142 !  1.165  1.243  0.272  0.107 ! 105
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -3.633 16.573 -1.645  0.390 ! -8.656  8.891 -1.054  0.275 ! -9.130  5.244 -0.660  0.203 !-15.064  4.483 -0.573  0.149 ! 105
 7X 7 ! -2.216 11.454 -1.386  0.325 ! -7.258  5.684 -0.965  0.219 ! -7.076  2.922 -0.470  0.151 ! -8.925  2.023 -0.308  0.109 ! 104
 8X 8 ! -0.949  9.681 -1.254  0.299 ! -6.127  6.200 -0.838  0.233 ! -2.091  3.312 -0.551  0.174 ! -1.813  1.858 -0.329  0.130 ! 107
 9X 9 ! -0.501 13.017  1.800  0.347 ! -5.661  7.808  1.050  0.264 ! -2.010  4.464  0.814  0.202 ! -1.456  2.356  0.457  0.147 ! 107
10X10 ! -3.717 12.054 -1.600  0.332 ! -8.881  7.444 -1.167  0.249 ! -6.928  3.780 -0.593  0.176 ! -5.440  1.790 -0.356  0.118 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -5.170 19.126 -1.850  0.418 !-10.131  9.063 -1.114  0.274 !-11.028  4.545 -0.677  0.178 !-17.139  4.606 -0.599  0.132 ! 102
 7X 7 !  0.606 13.904 -1.571  0.359 ! -4.371  7.189 -0.951  0.255 ! -5.244  3.825 -0.564  0.182 ! -4.902  1.684 -0.374  0.116 ! 103
 8X 8 ! -3.865  4.784  1.054  0.207 ! -8.960  3.498 -0.455  0.160 ! -7.661  2.120 -0.374  0.121 ! -6.237  1.028 -0.250  0.079 ! 104
 9X 9 ! -6.277  8.125 -1.433  0.268 !-11.423  6.494 -1.344  0.221 ! -9.535  4.380 -0.945  0.181 ! -8.220  2.865 -0.630  0.144 ! 106
10X10 ! -6.303  5.864 -1.130  0.226 !-11.439  4.153 -0.850  0.165 !-10.065  2.535 -0.507  0.122 ! -7.606  1.418 -0.327  0.090 ! 103
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 !  2.757 16.930  1.800  0.395 ! -2.208  8.461  0.941  0.279 ! -3.496  4.201  0.578  0.195 ! -7.872  2.533  0.402  0.135 ! 104
 7X 7 ! -7.247 18.094 -1.989  0.404 !-12.259 11.126 -1.106  0.300 !-13.603  7.347 -0.706  0.228 !-14.610  4.849 -0.560  0.163 ! 104
 8X 8 ! -3.872  9.016  1.456  0.287 ! -8.983  5.029 -0.838  0.199 ! -7.392  2.596 -0.446  0.139 !  7.510  1.446 -0.300  0.092 ! 105
 9X 9 ! -5.669  8.553 -1.331  0.276 !-10.818  5.052 -0.789  0.192 ! -9.990  3.052 -0.477  0.140 ! -8.833  1.803 -0.334  0.100 ! 105
10X10 ! -7.294 11.083  1.567  0.313 !-12.488  7.454 -1.094  0.236 !-11.597  5.292 -0.898  0.194 ! -8.809  3.093 -0.558  0.148 ! 105
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -3.787  2.298  0.567  0.142 ! -8.924  2.316 -0.436  0.121 !-10.379  1.996 -0.306  0.096 ! -6.512  0.848 -0.223  0.065 !  88
 7X 7 ! -0.312  1.918 -0.529  0.133 ! -5.441  1.510 -0.441  0.107 ! -5.449  0.994 -0.268  0.082 ! -1.773  0.434  0.142  0.061 ! 104
 8X 8 !  1.632  1.452  0.400  0.115 ! -3.457  1.033 -0.238  0.092 ! -4.913  0.796 -0.205  0.073 ! -2.051  0.456 -0.231  0.062 ! 107
 9X 9 !  0.027  0.989  0.400  0.096 ! -5.052  0.929 -0.233  0.080 ! -5.434  0.693 -0.201  0.062 ! -2.859  0.365 -0.184  0.052 ! 108
10X10 ! -0.843  0.762  0.300  0.084 ! -5.950  0.867 -0.213  0.071 ! -6.070  0.639 -0.184  0.052 ! -3.651  0.320 -0.153  0.043 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -4.245  2.163  0.471  0.136 ! -9.383  2.343 -0.460  0.119 ! -5.324  1.272 -0.264  0.097 ! -3.650  0.679 -0.193  0.072 !  91
 7X 7 ! -1.857  1.378 -0.433  0.112 ! -6.970  1.519 -0.483  0.100 ! -2.912  0.724 -0.217  0.077 ! -1.254  0.343 -0.121  0.055 ! 102
 8X 8 ! -0.389  0.931 -0.267  0.093 ! -5.484  1.042 -0.279  0.084 ! -1.038  0.522 -0.170  0.069 !  0.223  0.383 -0.211  0.060 ! 107
 9X 9 ! -0.944  0.710 -0.225  0.081 ! -6.055  0.913 -0.258  0.073 ! -2.885  0.450 -0.164  0.059 ! -1.352  0.296 -0.168  0.051 ! 108
10X10 ! -1.677  0.670  0.344  0.077 ! -6.790  0.923 -0.221  0.068 ! -3.890  0.429 -0.168  0.052 ! -1.804  0.244 -0.151  0.045 ! 108
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -7.422  2.695 -0.475  0.142 !-12.544  3.091 -0.449  0.123 ! -9.519  1.904 -0.288  0.099 ! -5.559  0.749 -0.196  0.065 !  62
 7X 7 ! -6.835  3.175 -0.600  0.159 !-11.943  3.555 -0.567  0.144 ! -8.028  2.200 -0.471  0.122 ! -5.708  1.116 -0.335  0.087 !  67
 8X 8 ! -2.660  2.851  0.567  0.161 ! -7.668  2.501 -0.366  0.135 ! -4.212  1.406 -0.252  0.107 ! -1.720  0.770 -0.221  0.083 !  68
 9X 9 ! -2.487  5.959 -1.100  0.234 ! -7.467  3.687 -0.485  0.171 ! -5.098  2.154 -0.370  0.133 ! -2.780  1.269 -0.304  0.105 !  80
10X10 ! -0.990  4.930 -0.850  0.213 ! -6.011  3.730  0.562  0.177 ! -3.991  2.342  0.377  0.143 ! -0.871  1.381  0.331  0.113 !  85
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -1.935  2.613  0.471  0.155 ! -7.119  2.200  0.386  0.127 ! -2.580  1.202  0.267  0.103 ! -1.085  0.638  0.185  0.076 !  62
 7X 7 ! -2.670  2.395 -0.600  0.147 ! -7.862  2.050 -0.456  0.117 ! -2.645  0.811 -0.250  0.083 ! -1.266  0.447  0.160  0.063 !  67
 8X 8 ! -1.368  2.070  0.471  0.138 ! -6.564  1.821 -0.379  0.115 ! -0.937  0.786 -0.229  0.085 !  1.324  0.648 -0.299  0.076 !  68
 9X 9 ! -2.184  3.196  0.614  0.171 ! -7.386  2.583 -0.525  0.139 ! -2.843  1.339 -0.376  0.108 !  1.264  0.966 -0.301  0.094 !  80
10X10 ! -1.985  3.193  0.700  0.171 ! -7.255  2.954 -0.579  0.151 ! -4.121  1.678 -0.397  0.119 !  2.329  0.956 -0.235  0.092 !  85
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -6.069  3.617 -0.700  0.174 !-11.203  3.479 -0.450  0.146 ! -7.878  2.009  0.280  0.115 ! -5.960  1.014 -0.195  0.080 !  53
 7X 7 ! -5.105  3.366  0.650  0.170 !-10.155  3.041  0.392  0.139 ! -8.976  2.298 -0.345  0.120 ! -5.288  1.223 -0.218  0.094 !  62
 8X 8 ! -6.012  4.160  0.900  0.188 !-11.162  3.743  0.445  0.155 ! -7.446  2.253  0.350  0.127 ! -3.297  1.464 -0.318  0.112 !  67
 9X 9 ! -6.535  5.059 -1.100  0.208 !-11.631  4.581 -0.650  0.176 ! -9.324  3.019 -0.412  0.143 ! -4.584  1.761 -0.347  0.120 !  80
10X10 ! -4.776  5.867  0.733  0.229 ! -9.889  4.898 -0.567  0.192 ! -7.161  3.161 -0.427  0.158 ! -1.066  1.941 -0.286  0.134 !  86
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -6.798  3.567 -0.600  0.170 !-11.948  3.677 -0.550  0.148 ! -8.085  2.041 -0.313  0.115 ! -6.312  1.199 -0.237  0.088 !  53
 7X 7 ! -4.111  2.995  0.567  0.162 ! -9.077  2.936 -0.443  0.142 ! -5.048  1.630 -0.303  0.114 ! -3.192  1.033  0.224  0.093 !  57
 8X 8 ! -5.220  3.920 -0.600  0.184 !-10.370  4.062 -0.600  0.168 ! -6.841  2.860 -0.434  0.150 !  1.003  1.667 -0.276  0.124 !  69
 9X 9 ! -6.090  4.795 -0.900  0.203 !-11.185  4.821 -0.733  0.184 ! -9.041  3.557 -0.564  0.161 !  0.500  2.233 -0.374  0.144 !  81
10X10 ! -5.411  3.679  0.614  0.178 !-10.649  3.786 -0.567  0.159 ! -3.564  2.217 -0.394  0.139 !  4.696  1.877 -0.265  0.124 !  79
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -5.654  1.175  0.300  0.090 !-10.804  1.806 -0.305  0.082 !-12.210  1.954 -0.240  0.073 !-10.084  1.272 -0.178  0.055 !  75
 7X 7 ! -1.710  0.902  0.329  0.096 ! -6.822  1.199  0.214  0.084 ! -7.257  1.091  0.179  0.075 ! -5.215  0.611  0.136  0.058 !  98
 8X 8 !  0.789  0.83[illegible]  0.329  0.087 ! -4.324  0.854  0.201  0.079 ! -4.430  0.736  0.162  0.072 ! -2.477  0.477  0.153  0.062 ! 108
 9X 9 ! -0.712  0.52[illegible]  0.233  0.069 ! -5.807  0.777 -0.150  0.066 ! -5.686  0.680 -0.148  0.059 ! -3.698  0.419 -0.139  0.052 ! 108
10X10 ! -1.700  0.407  0.186  0.059 ! -6.796  0.778 -0.150  0.057 ! -6.583  0.690 -0.147  0.052 ! -4.558  0.414 -0.125  0.045 ! 108


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -5.991  1.177 -0.300  0.088 !-11.104  1.926 -0.350  0.085 ! -9.960  1.520 -0.238  0.075 ! -8.090  0.953 -0.167  0.057 !  77
 7X 7 ! -1.943  0.743  0.200  0.081 ! -7.056  1.086 -0.150  0.076 ! -5.899  0.814  0.156  0.067 ! -4.018  0.464  0.125  0.054 !  97
 8X 8 !  0.245  0.676  0.186  0.079 ! -4.855  0.809 -0.150  0.074 ! -3.690  0.598 -0.138  0.066 ! -1.873  0.388 -0.142  0.057 ! 108
 9X 9 ! -1.026  0.466  0.173  0.065 ! -6.121  0.769 -0.150  0.063 ! -4.941  0.570 -0.138  0.056 ! -3.063  0.352 -0.128  0.050 ! 108
10X10 ! -2.010  0.375  0.167  0.056 ! -7.102  0.792 -0.150  0.055 ! -5.910  0.588 -0.138  0.049 ! -3.991  0.352 -0.118  0.043 ! 108


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 2 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -8.090  1.113  0.233  0.069 !-13.140  2.115  0.183  0.069 !-11.878  1.716  0.154  0.062 ! -9.826  1.145 -0.117  0.048 !  18
 7X 7 ! -6.212  1.316  0.300  0.094 !-11.262  2.064  0.225  0.091 ! -9.999  1.635  0.237  0.081 ! -7.948  1.046  0.123  0.065 !  31
 8X 8 ! -5.501  1.911 -0.500  0.123 !-10.551  2.378 -0.350  0.112 ! -9.288  1.872 -0.238  0.100 ! -7.383  1.259 -0.167  0.084 !  47
 9X 9 ! -4.549  1.659  0.400  0.117 ! -9.599  2.078 -0.317  0.107 ! -8.336  1.619 -0.221  0.095 ! -6.445  1.116  0.174  0.082 !  58
10X10 ! -3.703  2.164 -0.700  0.137 ! -8.753  2.271 -0.450  0.120 ! -7.490  1.683 -0.288  0.104 ! -5.558  1.114 -0.192  0.088 !  72


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 2 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -6.899  1.217  0.300  0.085 !-11.949  2.028  0.183  0.081 !-10.686  1.594  0.196  0.071 ! -8.635  1.008 -0.117  0.054 !  18
 7X 7 ! -5.555  1.286  0.275  0.096 !-10.605  1.950  0.225  0.092 ! -9.343  1.529  0.237  0.082 ! -7.291  0.951  0.165  0.065 !  31
 8X 8 ! -5.985  1.501 -0.300  0.104 !-11.035  2.082 -0.305  0.094 ! -9.804  1.626 -0.265  0.083 ! -6.909  0.947 -0.219  0.068 !  47
 9X 9 ! -6.493  1.703 -0.500  0.110 !-11.543  2.376 -0.517  0.103 !-10.386  1.921 -0.404  0.092 ! -6.749  1.146 -0.286  0.082 !  58
10X10 ! -6.173  1.494 -0.433  0.103 !-11.223  2.180 -0.460  0.097 !-10.166  1.789 -0.370  0.088 ! -5.862  0.947 -0.246  0.076 !  72


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 3 TYPE 1

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -8.057  1.225  0.300  0.076 !-13.107  2.171  0.250  0.073 !-11.845  1.709  0.162  0.062 ! -9.793  1.117 -0.117  0.046 !  10
 7X 7 ! -6.218  2.083 -0.500  0.126 !-11.268  2.640 -0.350  0.116 !-10.005  2.061  0.363  0.102 ! -7.954  1.279 -0.213  0.080 !  24
 8X 8 ! -6.376  2.558 -0.600  0.142 !-11.426  3.113 -0.600  0.133 !-10.213  2.437 -0.434  0.117 ! -6.343  1.270 -0.280  0.091 !  35
 9X 9 ! -7.586  3.072 -0.600  0.153 !-12.636  3.666 -0.614  0.142 !-11.436  3.028 -0.532  0.130 ! -7.427  1.787 -0.373  0.109 !  47
10X10 ! -7.673  3.138 -0.600  0.155 !-12.723  3.773 -0.588  0.145 !-11.573  3.138 -0.550  0.133 ! -6.871  1.829 -0.371  0.114 !  62


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  1 BITS
ALGORITHM 3 TYPE 2

RUNNING AVE LENGTH  1                             2                             4                             8
WINDOW
 6X 6 ! -9.114  1.137 -0.300  0.058 !-14.164  2.189 -0.250  0.055 !-12.902  1.782 -0.188  0.047 !-10.850  1.233 -0.142  0.036 !  10
 7X 7 ! -5.683  2.758  0.500  0.151 !-10.733  3.316  0.450  0.144 ! -9.471  2.706  0.437  0.132 ! -7.419  1.803  0.335  0.109 !  30
 8X 8 ! -6.643  2.967 -0.433  0.154 !-11.693  3.477 -0.460  0.143 !-10.614  2.909 -0.395  0.131 ! -3.388  1.416  0.308  0.110 !  37
 9X 9 ! -6.775  3.306 -0.529  0.163 !-11.825  3.891 -0.552  0.155 !-10.813  3.274 -0.494  0.142 ! -2.857  1.703 -0.348  0.123 !  49
10X10 ! -7.318  3.166 -0.529  0.157 !-12.439  3.762 -0.531  0.147 ! -9.623  2.875 -0.479  0.137 ! -2.892  1.732 -0.348  0.124 !  66
>

11.37.08 >los
520.85 ARU'S, .83 CONNECT HRS
LOGGED OFF AT 11.37.11 ON  03MAR82_
```

STAGE II

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  0.047  1.043  0.351  0.098 ! 105
 7X 7 ! -5.508  1.429 -0.305  0.103 ! 103
 8X 8 !  1.086  1.587  0.269  0.121 ! 104


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.158  0.641  0.218  0.077 ! 105
 7X 7 ! -0.692  0.394 -0.174  0.060 ! 105
 8X 8 !  1.316  0.608  0.172  0.074 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.068  1.061 -0.290  0.099 ! 101
 7X 7 !-13.185  5.108 -0.563  0.180 ! 101
 8X 8 ! -2.220  1.043 -0.311  0.096 ! 100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -5.279  3.931 -0.544  0.184 !  98
 7X 7 ! -2.637  1.921 -0.295  0.181 ! 104
 8X 8 ! -1.207  1.233 -0.359  0.106 ! 103


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -1.746  0.539 -0.219  0.069 !  65
 7X 7 !  9.604  1.842  0.400  0.096 !  81
 8X 8 !  9.633  1.684  0.359  0.087 ! 105


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -7.326  0.760 -0.150  0.049 !  57
 7X 7 ! 10.303  1.368  0.233  0.060 !  73
 8X 8 ! -6.250  0.905 -0.197  0.071 !  94
>

13.46.56 >query time
123.70 ARU'S, .45 CONNECT HRS

13.47.02 >run pipec gauss image
EXECUTION:
>
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !   0.164  4.345 -0.609  0.201 ! 102
 7X 7 !-11.025  3.130 -0.520  0.136 !  98
 8X 8 !-11.080  3.514 -0.419  0.148 ! 101


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !   0.017  0.299 -0.240  0.053 ! 107
 7X 7 !  -0.305  0.369 -0.149  0.058 ! 105
 8X 8 !   0.139  0.247 -0.174  0.048 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !   0.150  5.143 -0.683  0.218 !  98
 7X 7 !  -6.675  2.298 -0.332  0.132 !  99
 8X 8 !  -4.658  1.772 -0.411  0.121 !  93


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  -7.018  4.440 -0.616  0.192 ! 102
 7X 7 !  -0.050  1.787 -0.421  0.129 ! 101
 8X 8 !   4.863  1.606  0.324  0.113 ! 103


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  -6.481  1.310 -0.261  0.092 !  65
 7X 7 !   4.445  2.295  0.482  0.140 !  81
 8X 8 !   6.757  1.076  0.319  0.078 ! 105


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-11.273  1.861 -0.234  0.080 !  57
 7X 7 !   7.149  1.085  0.245  0.075 !  73
 8X 8 !  -8.805  1.536 -0.228  0.087 !  94
>
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  0.029 13.947 -0.879  0.359 !  89
 7X 7 !-13.648  7.176 -0.688  0.275 !  91
 8X 8 !-20.701 10.117 -0.719  0.239 !  96


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.049  0.446 -0.263  0.064 ! 108
 7X 7 ! -2.334  0.784 -0.232  0.082 ! 106
 8X 8 ! -1.346  0.388 -0.179  0.059 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  0.062 16.369 -0.950  0.389 !  84
 7X 7 !-21.739  9.231 -0.729  0.212 !  92
 8X 8 !-20.951  8.033 -0.679  0.192 !  92


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -5.012  2.093 -0.407  0.131 ! 103
 7X 7 ! -7.201  5.533 -0.651  0.216 ! 101
 8X 8 ! -6.738  0.938 -0.269  0.069 ! 101


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-21.098  7.285 -0.518  0.171 !  63
 7X 7 !-11.136  5.259 -0.536  0.195 !  79
 8X 8 ! -7.925  1.871 -0.330  0.109 ! 104


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-20.561  7.036 -0.606  0.170 !  57
 7X 7 !-10.513  3.405 -0.376  0.149 !  73
 8X 8 !-22.348  6.684 -0.557  0.138 !  94
>

14.36.26 >los
570.78 ARU'S, 1.27 CONNECT HRS
LOGGED OFF AT 14.36.30 ON  03MAR82H
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.222 41.101 -1.357  0.617 !  81
 7X 7 !-34.117 21.157 -1.024  0.310 !  84
 8X 8 !-25.815 15.688 -0.967  0.297 !  92


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !  0.014  0.140  0.146  0.036 ! 107
 7X 7 ! -2.876  0.853 -0.282  0.085 ! 104
 8X 8 ! -3.435  0.758 -0.267  0.077 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.184 26.795 -0.928  0.498 !  73
 7X 7 !-21.410  8.756 -0.804  0.204 !  83
 8X 8 !-38.721 19.150 -0.893  0.221 !  92


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -6.400  3.057 -0.508  0.157 ! 102
 7X 7 !-25.759  8.491 -0.653  0.147 !  90
 8X 8 !-23.907  9.377 -0.899  0.194 !  93


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-29.743 12.544 -0.664  0.201 !  59
 7X 7 !-23.967 12.503 -0.682  0.258 !  79
 8X 8 !-24.752  8.620 -0.513  0.165 ! 104


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-34.473 14.575 -0.633  0.182 !  57
 7X 7 !-30.145 12.870 -0.628  0.203 !  73
 8X 8 ! -7.319  3.795  0.413  0.175 !  93
>
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.614 48.288 -1.331  0.669 !  71
 7X 7 !-60.552 48.101 -1.239  0.362 !  77
 8X 8 !-51.893 35.905 -1.201  0.319 !  82


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.124  0.707 -0.346  0.081 ! 105
 7X 7 ! -3.565  0.632 -0.248  0.069 ! 106
 8X 8 !  0.421  0.670 -0.230  0.079 ! 106


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.389 58.753 -1.722  0.738 !  71
 7X 7 !-51.495 32.383 -1.115  0.269 !  83
 8X 8 !-57.098 38.599 -0.934  0.279 !  82


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-11.836  5.645 -0.727  0.201 ! 100
 7X 7 !-25.187  9.223 -0.756  0.176 !  94
 8X 8 !-11.927  3.289 -0.495  0.135 ! 100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 1

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-51.472 31.703 -0.882  0.258 !  57
 7X 7 !-28.751 15.008 -0.864  0.261 !  75
 8X 8 !-26.800  8.758 -0.630  0.140 ! 103


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
ALGORITHM 1 TYPE 2

RUNNING AVE LENGTH  8
WINDOW
 6X 6 !-47.542 26.756 -0.886  0.232 !  57
 7X 7 !-34.170 15.732 -0.772  0.213 !  73
 8X 8 !-19.458  6.554 -0.644  0.168 !  94
>
```

STAGE III

```
EXECUTION:
>.05,.1,0.,58,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.100 PIXELS
MEAN INTENSITY IS 0.491
STD DEV IS 0.198
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.008  0.034 -0.103  0.033 ! 107
 7X 7 ! -0.001  0.009 -0.024  0.009 ! 107
 8X 8 !  0.000  0.007 -0.024  0.007 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.002  0.049 -0.126  0.049 ! 107
 7X 7 !  0.001  0.014  0.039  0.014 ! 107
 8X 8 !  0.004  0.010  0.035  0.009 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.093  0.094 -0.100  0.013 !  97
 7X 7 ! -0.088  0.089 -0.100  0.016 ! 103
 8X 8 ! -0.088  0.089 -0.100  0.015 ! 107
>



 .05,.3,0.,589,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.300 PIXELS
MEAN INTENSITY IS 0.553
STD DEV IS 0.181
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.024  0.042 -0.101  0.035 ! 107
 7X 7 ! -0.015  0.051 -0.171  0.049 ! 107
 8X 8 ! -0.012  0.032 -0.095  0.030 ! 106


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.024  0.044 -0.117  0.037 ! 107
 7X 7 ! -0.020  0.054 -0.194  0.051 ! 106
 8X 8 ! -0.012  0.034 -0.113  0.031 ! 106
```

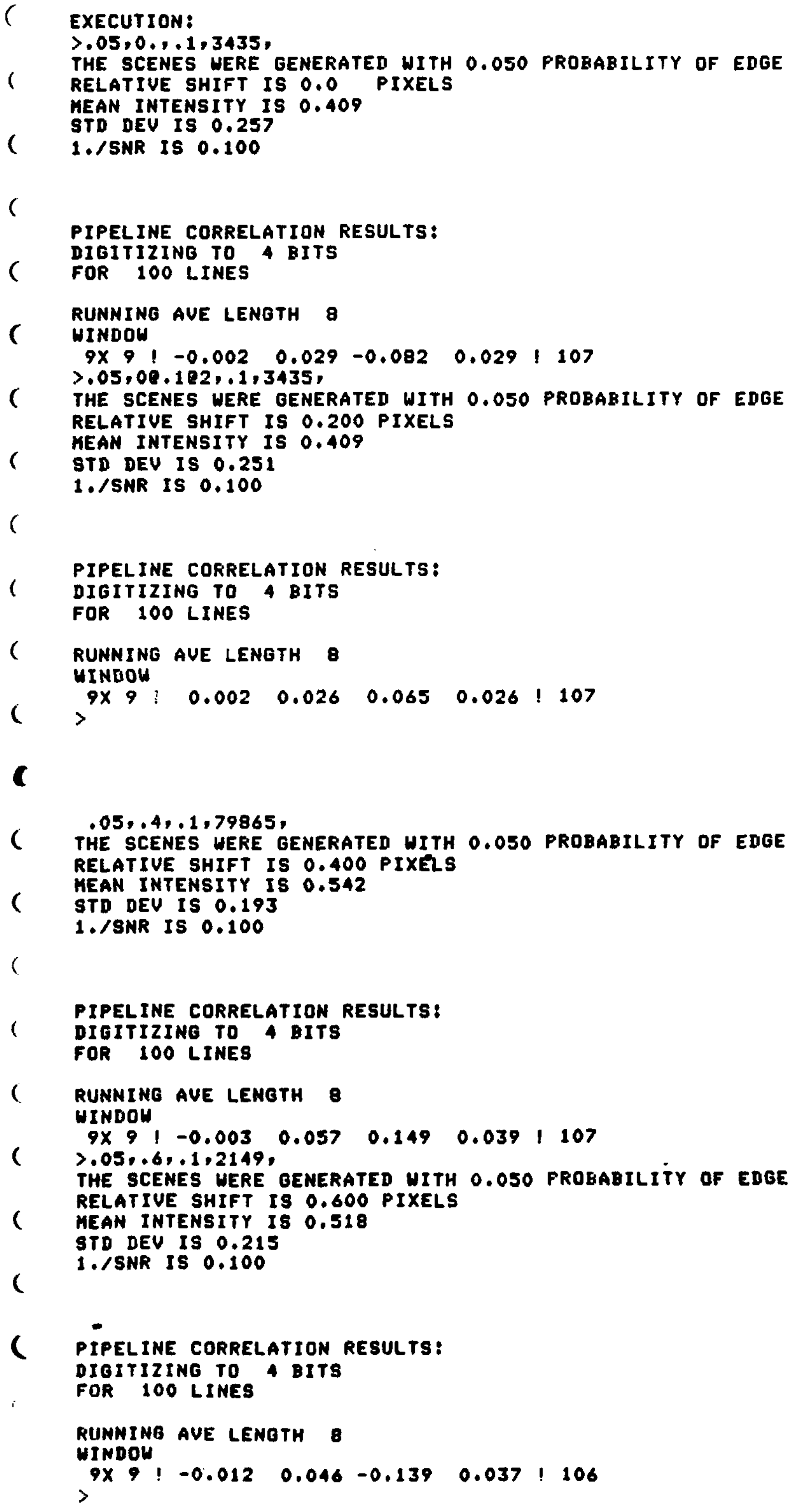
```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.092  0.140 -0.358  0.105 !  93
 7X 7 ! -0.064  0.119 -0.324  0.100 !  99
 8X 8 ! -0.064  0.105 -0.295  0.083 ! 100
>


 .05,.5,0.,43,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.500 PIXELS
MEAN INTENSITY IS 0.534
STD DEV IS 0.195
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.020  0.048  0.114  0.043 ! 106
 7X 7 !  0.008  0.025  0.085  0.024 ! 107
 8X 8 !  0.005  0.018  0.066  0.017 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.020  0.047  0.113  0.043 ! 107
 7X 7 !  0.005  0.027  0.089  0.026 ! 107
 8X 8 !  0.003  0.019  0.069  0.019 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.137  0.181 -0.466  0.117 !  95
 7X 7 ! -0.078  0.130 -0.318  0.104 ! 106
 8X 8 ! -0.073  0.100 -0.255  0.068 ! 106
>


 .05,.7,0.,753,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.700 PIXELS
MEAN INTENSITY IS 0.493
STD DEV IS 0.223
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.036  0.109 -0.282  0.103 ! 107
 7X 7 ! -0.002  0.053 -0.123  0.053 ! 107
 8X 8 !  0.008  0.039  0.115  0.038 ! 107


PIPELINE CORRELATION RESULTS:
```

```
PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.023  0.105 -0.282  0.102 ! 107
 7X 7 !  0.007  0.044 -0.126  0.044 ! 107
 8X 8 !  0.013  0.036  0.107  0.033 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.135  0.196 -0.504  0.143 !  97
 7X 7 ! -0.103  0.167 -0.481  0.131 ! 101
 8X 8 ! -0.088  0.148 -0.517  0.119 ! 105
>
```

```
 .05,1.1,0.,32,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.100 PIXELS
MEAN INTENSITY IS 0.519
STD DEV IS 0.242
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.059  0.095 -0.203  0.074 ! 107
 7X 7 ! -0.025  0.058 -0.185  0.052 ! 107
 8X 8 ! -0.019  0.051 -0.190  0.047 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.065  0.099 -0.236  0.075 ! 107
 7X 7 ! -0.022  0.065 -0.191  0.061 ! 107
 8X 8 ! -0.016  0.055 -0.198  0.052 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.146  0.229 -0.556  0.176 !  91
 7X 7 ! -0.083  0.165 -0.431  0.142 ! 103
 8X 8 ! -0.058  0.134 -0.278  0.121 ! 107
>
```

```
 .05,.9,0.,98754,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.900 PIXELS
MEAN INTENSITY IS 0.483
STD DEV IS 0.211
1./SNR IS 0.0


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  6 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.101  0.159 -0.357  0.123 ! 106
 7X 7 ! -0.008  0.033 -0.122  0.032 ! 107
 8X 8 ! -0.010  0.042 -0.159  0.041 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 6X 6 ! -0.102  0.166 -0.335  0.131 ! 105
 7X 7 ! -0.006  0.036 -0.129  0.036 ! 107
 8X 8 ! -0.010  0.049 -0.184  0.048 ! 107


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  2 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOI'
 6X 6 ! -0.185  0.241 -0.507  0.154 !  91
 7X 7 ! -0.030  0.152 -0.409  0.149 ! 100
 8X 8 ! -0.025  0.100 -0.243  0.097 ! 104
>

14.59.11 >los
188.39 ARU'S, .25 CONNECT HRS
LOGGED OFF AT 14.59.14 ON  05MAR82I
```

STAGE IV



```
EXECUTION:
>.05,0.,.1,3435,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.0   PIXELS
MEAN INTENSITY IS 0.409
STD DEV IS 0.257
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.002  0.029 -0.082  0.029 ! 107
>.05,0@.1@2,.1,3435,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.200 PIXELS
MEAN INTENSITY IS 0.409
STD DEV IS 0.251
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 !  0.002  0.026  0.065  0.026 ! 107
>


 .05,.4,.1,79865,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.400 PIXELS
MEAN INTENSITY IS 0.542
STD DEV IS 0.193
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.003  0.057  0.149  0.039 ! 107
>.05,.6,.1,2149,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.600 PIXELS
MEAN INTENSITY IS 0.518
STD DEV IS 0.215
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.012  0.046 -0.139  0.037 ! 106
>
```

```
 .05,.8,.1,6551,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.800 PIXELS
MEAN INTENSITY IS 0.525
STD DEV IS 0.198
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.026  0.043  0.084  0.029 ! 107
>



 .05,1.,.1,415,7,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.000 PIXELS
MEAN INTENSITY IS 0.484
STD DEV IS 0.210
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.050  0.105 -0.358  0.060 ! 107

FROM RJESCED :  Job cannot run until you release CAISOFTW's disks.
>



 .05,0.,.2,54546,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.0   PIXELS
MEAN INTENSITY IS 0.468
STD DEV IS 0.212
1./SNR IS 0.200


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.002  0.048  0.134  0.037 ! 107
>
```

```
 .05,1.,.2,1557,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.000 PIXELS
MEAN INTENSITY IS 0.528
STD DEV IS 0.185
1./SNR IS 0.200


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.091  0.120 -0.299  0.055 ! 107
>
```

```
 .05,0.,.4,43,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.0   PIXELS
MEAN INTENSITY IS 0.483
STD DEV IS 0.207
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 !  0.015  0.100 -0.234  0.048 ! 107
>
```

```
 .05,1.,.4,653,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.000 PIXELS
MEAN INTENSITY IS 0.489
STD DEV IS 0.232
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.192  0.223 -0.429  0.065 ! 107
>
```

```
 .05,.8,.4,45,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.800 PIXELS
MEAN INTENSITY IS 0.483
STD DEV IS 0.171
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.176  0.197 -0.352  0.059 ! 107
>
```

```
 .05,.6,.4,6543,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.600 PIXELS
MEAN INTENSITY IS 0.543
STD DEV IS 0.227
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.073  0.115 -0.308  0.055 ! 107
>
```

```
 .05,.4,.4,6574,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.400 PIXELS
MEAN INTENSITY IS 0.424
STD DEV IS 0.241
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.071  0.126 -0.239  0.049 ! 107
>
```

```
 .05,.2,.4,654,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.200 PIXELS
MEAN INTENSITY IS 0.485
STD DEV IS 0.179
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.046  0.108 -0.234  0.048 ! 107
>

FROM RJESCED :  Job cannot run until you release CAISOFTW's disk.

18.11.57 > .
```

STAGE V

```
EXECUTION:
>.05,0.,.1,3435,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.0   PIXELS
MEAN INTENSITY IS 0.409
STD DEV IS 0.257
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.002  0.029 -0.082  0.029 ! 107 ! -0.001  0.027 -0.066  0.027 ! 107
>





 .05,.8,.1,6551,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.800 PIXELS
MEAN INTENSITY IS 0.525
STD DEV IS 0.198
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.026  0.043  0.084  0.034 ! 107 ! -0.021  0.052 -0.110  0.047 ! 107
>.05,1.,.1,415,




THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.000 PIXELS
MEAN INTENSITY IS 0.484
STD DEV IS 0.210
1./SNR IS 0.100


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.050  0.105 -0.358  0.092 ! 107 ! -0.054  0.128 -0.520  0.116 ! 107
>





 .05,.8,.4,45,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.800 PIXELS
MEAN INTENSITY IS 0.483
STD DEV IS 0.171
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.176  0.197 -0.352  0.088 ! 107 ! -0.164  0.187 -0.335  0.091 ! 107
>
```

```
 .05,.6,.4,6543,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.600 PIXELS
MEAN INTENSITY IS 0.543
STD DEV IS 0.227
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.073  0.115 -0.308  0.088 ! 107 ! -0.071  0.112 -0.311  0.086 ! 107
>
```

```
 /@.05,1.,.4,653,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 1.000 PIXELS
MEAN INTENSITY IS 0.489
STD DEV IS 0.232
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.192  0.223 -0.429  0.114 ! 107 ! -0.183  0.215 -0.429  0.113 ! 107
>
```

```
 .05,.4,.4,6574,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.400 PIXELS
MEAN INTENSITY IS 0.424
STD DEV IS 0.241
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 ! -0.071  0.126 -0.239  0.104 ! 107 ! -0.069  0.118 -0.237  0.095 ! 106
>
```

```
 .05,0.,.4,43,
THE SCENES WERE GENERATED WITH 0.050 PROBABILITY OF EDGE
RELATIVE SHIFT IS 0.0   PIXELS
MEAN INTENSITY IS 0.483
STD DEV IS 0.207
1./SNR IS 0.400


PIPELINE CORRELATION RESULTS:
DIGITIZING TO  4 BITS
FOR  100 LINES

RUNNING AVE LENGTH  8
WINDOW
 9X 9 !  0.015  0.100 -0.234  0.099 ! 107 !  0.017  0.102 -0.234  0.100 ! 107
>
```

STAGE I

```
C      PROGRAM: PIPELINE CORRELATION
C      CAI DIVISON RECON OPTICAL
C      D. DEFOE
C      DATE: 10 FEBRUARY 1982
C
       INTEGER SEED
       REAL NS
       DIMENSION P1(7,11), P2(7,11), ILN(10,3,2,6,5),
      >IASN(3,2,6,5), IASD(3,2,6,5), WR(8,3,2,6,5), L(4),
      >EA(3,2,6,5,4), E2(3,2,6,5,4), EMX(3,2,6,5,4), KO(3,2),
      >KD(6), KF(4), SF(4), ILD(10,3,2,6,5), FR(3,2,6,5,4),
      >SDF(4),PL1(10),PL2(10),RKF(4),KE(3,2,6,5)
       DATA KD/10,8,6,4,2,1/,                        KF/1,2,4,8/
       NM=7; NL=100; NW=8; NAL=3; NAD=2; ND=6; NF=4
       JWIN=6; JWIX=10
       CM=.25; W=.02; CWH=4.
       NA=JWIX; NN=JWIX+1; NWI=JWIX-JWIN+1
       RH=SQRT(.5)
       KS1=NM+NA-1; KS2=KS1+NW-1
       NMM=NM-1; NMP=NM+1; NLM=NL+KS2
       NNM=NN-1; NNP=NN+1; NAP=NA+1
       NWM=NW-1; NWP=NW+1; NWIM=NWI-1; NWIP=NWI+1
       JWINM=JWIN-1
       WNH=CNH*W; WN=WNH+WNH; WNP=1.-WN
       DO 8 IW=1,NWI
       DO 7 ID=1,ND
       DO 6 IAD=1,NAD
       DO 5 IA=1,NAL
       KE(IA,IAD,ID,IW)=0
     5 CONTINUE
     6 CONTINUE
     7 CONTINUE
     8 CONTINUE
       DO 9 IF=1,NF
     9 RKF(IF)=1./KF(IF)
     1 SP=SP2=0.; NP=NE=0
       READ(5,5000) PE,D1,NS,SEED
5000   FORMAT(3F10.4,I10)
       IF(PE.EQ.0) GO TO 9999
       D2=1-D1 ; C=(.5-ABS(.5-D1))*CM ; MP=NM
       MPM=MP-1
       MA=NA ; MW=NW
       DO 499 M=1,NLM

           MPM=MP ; MP=MP+1
       IF(MP.EQ.NMP) MP=1
       IF(M.LT.NM) GO TO 10
       MA=MA+1
       IF(MA.EQ.NAP) MA=1
       IF(M.LT.KS1) GO TO 10
       MW=MW+1
       IF(MW.EQ.NWP) MW=1
       NE=NE+1
10     DO 199 NR=1,NN
       N=NNP-NR ; U=RAN(SEED)
       IF(U.GE.PE) GO TO 20
       IU=5
       GO TO 30
20     U=RAN(SEED) ; IU=4*U+1 ; L(1)=0
30     IF(IU.GE.3) L(1)=1
       L(2)=L(3)=L(4)=L(1)
       IF(IU.GE.4) GO TO 100
       U=RAN(SEED)
       IF(IU.EQ.2) GO TO 50
       JU=4*U+1
       IF(IU.NE.1) GO TO 40
       L(JU)=1
       GO TO 100
40     L(JU)=0
       GO TO 100
50     JU=6*U
       IF(JU.GE.3) GO TO 60
       L(1)=1 ; L(JU+2)=1
       GO TO 100
60     L(JU-1)=1
       IF(JU.EQ.5) JU=2
       L(JU)=1
100    T1=T2=T3=T4=.5
       IF(N.LT.NN) T1=P1(MP,N+1)
       IF(M.EQ.1) GO TO 110
       T3=P1(MPM,N)
       IF(N.LT.NN) T2=P1(MPM,N+1)
       IF(N.GT.1) T4=P1(MPM,N-1)
110    WA=(T1*L(1)+T3*L(3)+(T2*L(2)+T4*L(4))*RH)/(L(1)+L(3)+
      >(L(2)+L(4))*RH)
       IF(IU.EQ.5) GO TO 120
       Q=IU*U ; Q=GAUSS(SEED,WA,Q) ; GO TO 150
120    U=RAN(SEED)
       IF(WA.LT.WN) GO TO 130
       IF(WA.GT.WNP) GO TO 140
       Q=WNP*U
       IF(U.GT.WA) Q=Q+WN
       GO TO 150
```

```
130    Q=(WA+WNH)*(1.-U)+U
       GO TO 150
140    Q=(WA-WNH)*U
150    P1(MP,N)=Q ; SP=SP+Q
       SP2=SP2+Q*Q ; NP=NP+1
       IF(M.EQ.1) GO TO 199
       T1=P1(MPM,N)
       T3=P1(MP,N)
       T2=T1-T3
       Q=T1-D1*T2
       P2(MPM,N)=Q ; SP=SP+Q
       SP2=SP2+Q*Q ; NP=NP+1
199    CONTINUE
6000   FORMAT(2(1X,10(F5.3,1X)))
200    IF(M.LT.NM) GO TO 499
       SIP=SQRT((SP2-SP*SP/NP)/NP) ; SIN=NS*SIP
       IF(M.GT.NM) GO TO 210
       DO 209 MM=1,NMM
       DO 208 N=1,NNM
       T1=P1(MM,N)+GAUSS(SEED,0.,SIN)
       T2=P2(MM,N)+GAUSS(SEED,0.,SIN)
       T1=AMAX1(AMIN1(T1,1.),0.)
       T2=AMAX1(AMIN1(T2,1.),0.)
       P1(MM,N)=PL1(N)=T1
       P2(MM,N)=PL2(N)=T2
208    CONTINUE
       CALL IMAGE(10,PL1,PL2)
209    CONTINUE
       GO TO 220
210    DO 219 N=1,NNM
       T1=P1(MPM,N)+GAUSS(SEED,0.,SIN)
       T2=P2(MPM,N)+GAUSS(SEED,0.,SIN)
       T1=AMAX1(AMIN1(T1,1.),0.)
       T2=AMAX1(AMIN1(T2,1.),0.)
       P1(MPM,N)=PL1(N)=T1
       P2(MPM,N)=PL2(N)=T2
219    CONTINUE
       CALL IMAGE(10,PL1,PL2)
   220 DO 498 IA=1,NAL


       DO 497 IAD=1,NAD






       DO 496 ID=1,ND
       KSH=KD(ID) ; KS=2**KSH
       IF(M.LE.KS1) GO TO 230
       MAM=MA+JWINM
       IF(MAM.GT.NA) MAM=MAM-NA
       DO 229 IW=1,NWI
       MAM=MAM-1
       IF(MAN.LE.0) MAM=MAM+NA
       IASN(IA,IAD,ID,IW)=IASN(IA,IAD,ID,IW)-ILN(MAM,IA,IAD,ID,IW)
       IASD(IA,IAD,ID,IW)=IASD(IA,IAD,ID,IW)-ILD(MAM,IA,IAD,ID,IW)
229    CONTINUE
   230 IS1=IS2=0
       DO 249 N=1,JWIN





249    CONTINUE
       ILN(MA,IA,IAD,ID,1)=IS1
       ILD(MA,IA,IAD,ID,1)=IS2
       IF(NWI.EQ.1) GO TO 260
       DO 259 IWM=1,NWIM
       N=JWIN+IWM ; IW=IWM+1





       ILN(MA,IA,IAD,ID,IW)=IS1
       ILD(MA,IA,IAD,ID,IW)=IS2
259    CONTINUE
260    IF(M.LT.KS1) GO TO 496
       IF(M.GT.KS1) GO TO 280
       DO 279 IWM=1,NWI
       IS1=IS2=0
       IW=NWIP-IWM
       DO 269 MAM=IWM,JWIX
       IS1=IS1+ILN(MAM,IA,IAD,ID,IW)
       IS2=IS2+ILD(MAM,IA,IAD,ID,IW)
```

```
  269 CONTINUE
      IASN(IA,IAD,ID,IW)=IS1
      IASD(IA,IAD,ID,IW)=IS2
  279 CONTINUE
      GO TO 300
 280  DO 289 IW=1,NWI
      IASN(IA,IAD,ID,IW)=IASN(IA,IAD,ID,IW)+ILN(MA,IA,IAD,ID,IW)
      IASD(IA,IAD,ID,IW)=IASD(IA,IAD,ID,IW)+ILD(MA,IA,IAD,ID,IW)
 289  CONTINUE
 290  IF(M.LE.KS2) GO TO 300
      DO 299 IW=1,NWI
      DO 299  IF=1,NF
      MM=MW-KF(IF)$ RF=RKF(IF)
      IF(MM.LE.0) MM=MM+NW
      FR(IA,IAD,ID,IW,IF)=FR(IA,IAD,ID,IW,IF)-WR(MM,IA,IAD,ID,IW)*RF
 299  CONTINUE
  300 DO 495 IW=1,NWI
      T1=IASN(IA,IAD,ID,IW) $ T2=IASD(IA,IAD,ID,IW)
      T3=ABS(T2)
      IF(T3.GT..5*ABS(T1).AND.T3.GE.5.) GO TO 301
                     GO TO 302
  301 KE(IA,IAD,ID,IW)=KE(IA,IAD,ID,IW)+1
  302 WR(MW,IA,IAD,ID,IW)
      IF(M.LT.KS2) GO TO 495
      I1=0$ S1=0.$ MM=NWP
      DO 494 IF=1,NF
      RF=RKF(IF)
      IF(M.GT.KS2) GO TO 330
      I2=KF(IF) $ I3=I2-I1 $ I1=I2
      DO 309 I=1,I3
      MM=MM-1 $ S1=S1+WR(MM,IA,IAD,ID,IW)
 309  CONTINUE
      S1=S1*RF
      FR(IA,IAD,ID,IW,IF)=S1
      GO TO 340
 330  S1=FR(IA,IAD,ID,IW,IF)+WR(MW,IA,IAD,ID,IW)*RF
      FR(IA,IAD,ID,IW,IF)=S1
 340  S1=S1-D1
      EA(IA,IAD,ID,IW,IF)=EA(IA,IAD,ID,IW,IF)+S1
      E2(IA,IAB,ID,IW,IF)=E2(IA,IAB,ID,IW,IF)+S1*S1
      S2=EMX(IA,IAD,ID,IW,IF) $ S2=ABS(S2)
      IF(ABS(S1).GT.S2) EMX(IA,IAD,ID,IW,IF)=S1
  494 CONTINUE
  495 CONTINUE
  496 CONTINUE
  497 CONTINUE
  498 CONTINUE
  499 CONTINUE
      SP=SP/NP
      WRITE(6,6001) PE,D1,SP,SIP,NS
 6001 FORMAT(' THE ABOVE SCENES WERE GENERATED WITH ',F5.3,
     >' PROBABILITY OF EDGE '/' RELATIVE SHIFT IS ',F5.3,' PIXELS'/
     >' MEAN INTENSITY IS ',F5.3,/' STD DEV IS ',F5.3,/' 1./SNR IS '
     >,F5.3)
      RNE=1./NE
      DO 599 ID=1,ND
      DO 599 IA=1,NAL
      DO 599 IAD=1,NAD
      WRITE(6,6002) KD(ID),IA,IAD,(KF(IF),IF=1,NF)
 6002 FORMAT(///' PIPELINE CORRELATION RESULTS: '/' DIGITIZING TO '
     >,I2,' BITS'/' ALGORITHM ',I1,' TYPE ',I1//' RUNNING AVE ',
     >'LENGTH  ',3(I1,29X),I1/' WINDOW')
      I1=JWIN-1
      DO 599 IW=1,NWI
      I1=I1+1
      DO 598 IF=1,NF
      S1=EA(IA,IAD,ID,IW,IF)
      SDF(IF)=SQRT((E2(IA,IAD,ID,IW,IF)-S1*S1*RNE)*RNE)
 598  CONTINUE
      WRITE(6,6003) I1,I1,(EA(IA,IAD,ID,IW,IF),E2(IA,IAD,ID,IW,IF),
     >EMX(IA,IAD,ID,IW,IF),SDF(IF),IF=1,NF),KE(IA,IAD,ID,IW)
 6003 FORMAT(I3,'X',I2,4(' !',4F7.3),' !',I4)
 599  CONTINUE
      GO TO 1
 9999 STOP
      END
```

```
13.21.39 >printf gauss fortran

C   D.DEFOE 10/22/80
C        ROUTINE  TO FORM GAUSSIAN DISTRIBUTION OF RANDOM VARIABLES
      FUNCTION GAUSS(ISEED,A,S)
      GAUSS=0.
      DO 1 I=1,12
      GAUSS=GAUSS+RAN(ISEED)
    1 CONTINUE
      GAUSS=(GAUSS-6.)*S+A
      RETURN
      END


13.21.58 >printf image fortran

C   D. DEFOE 9/25/80 PROGRAM TO PRODUCE IMAGE ON TELETYPE FROM ARRAY P
C                    VALUES OF P BETWEEN 0. AND 1.$ 'IS' IS SCALE FACTOR
C                    IMAGE SEGMENTED TO FIT ON 132 CHAR. LINE IF
C                    NECESSARY
      SUBROUTINE IMAGE(NX,P1,P2)
      DIMENSION P1(10),P2(10)
      LOGICAL*1 NCR,NFF,NLF,N(6,26),NN(6,6),NL1(10,6),NL2(10,6)
      DATA NCR/Z15/,NLF/Z0D/,NFF/Z0C/
      DATA  N/ZC8,ZAD,ZBD,Z7B,ZC9,Z40,
     >        ZC8,ZAD,ZBD,Z7B,ZC9,Z40,
     >        ZC8,ZAD,ZBD,Z7B,Z7A,Z40,
     >        ZC8,ZAD,ZBD,Z4D,Z5D,Z40,
     >        ZC8,ZD6,ZF0,Z4D,Z5D,Z5C,
     >        ZC1,ZD6,ZE5,Z7A,Z7B,Z40,
     >        ZC1,ZD6,ZE5,Z7E,Z4E,Z40,
     >        ZC1,ZD6,ZE5,Z7E,Z60,Z7A,
     >        ZC1,ZD6,ZE5,Z7E,Z60,Z40,
     >        ZC8,ZF0,Z7A,Z40,Z40,Z40,
     >        ZC8,ZF0,Z4E,Z40,Z40,Z40,
     >        ZC8,ZD6,Z7A,Z40,Z40,Z40,
     >        ZC8,ZD6,Z4E,Z40,Z40,Z40,
     >        ZD6,Z60,Z7A,Z40,Z40,Z40,
     >        ZD6,Z4E,Z40,Z40,Z40,Z40,
     >        ZD6,Z7E,Z40,Z40,Z40,Z40,
     >        Z4C,Z6E,Z7E,Z40,Z40,Z40,
     >        Z4C,Z6E,Z60,Z40,Z40,Z40,
     >        Z4C,Z6E,Z40,Z40,Z40,Z40,
     >        Z60,Z7A,Z40,Z40,Z40,Z40/
      DATA NN/Z4E,Z40,Z40,Z40,Z40,Z40,
     >        Z7E,Z40,Z40,Z40,Z40,Z40,
     >        Z6A,Z40,Z40,Z40,Z40,Z40,
     >        Z60,Z40,Z40,Z40,Z40,Z40,
     >        Z7B,Z40,Z40,Z40,Z40,Z40,
     >        Z40,Z40,Z40,Z40,Z40,Z40/
      EQUIVALENCE (N(1,21),NN(1,1))
      CALL CSS(IERR,'TERMDEF ','$       ','XL      ','ON      ')
      DO 10 IX=1,10
      L1=25.*P1(IX)+1.5$ L2=25.*P2(IX)+1.5
      DO 10 LX=1,6
      NL1(IX,LX)=N(LX,L1)$ NL2(IX,LX)=N(LX,L2)
   10 CONTINUE
      DO 12 LX=1,6
   12 WRITE(6,6000)(NL1(IX,LX),IX=1,10),(NL2(IX,LX),IX=1,10)
 6000 FORMAT('+',10A1,3X,10A1)
      WRITE (6,6000) NCR
      WRITE(6,6000) NLF
      CALL CSS(IERR,'TERMDEF ','$       ','XL      ','OFF     ')
      RETURN
      END
```

STAGE II

```
C     PROGRAM: PIPELINE CORRELATION
C     CAI DIVISON RECON OPTICAL
C     D. DEFOE
C     DATE: 10 FEBRUARY 1982
C
      INTEGER SEED
      REAL NS
      DIMENSION P1(7,9), P2(7,9), ILN(8,1,2,3,3),
     >IASN(1,2,3,3), IASD(1,2,3,3), WR(8,1,2,3,3), L(1),
     >EA(1,2,3,3,1), E2(1,2,3,3,1), EMX(1,2,3,3,1), KO(1,2),
     >KD(3), KF(1), SF(1), ILD(8,1,2,3,3), FR(1,2,3,3,1),
     >SDF(1),PL1(9),PL2(9),RKF(1),KE(1,2,3,3)
      DATA KD/6,4,2/           KF/8/
      NM=7; NL=100; NW=8; NAL=1; NAD=2; ND=3; NF=1
      JWIN=6; JWIX=8
      CM=.25; W=.02; CWH=4.
      NA=JWIX; NN=JWIX+1; NWI=JWIX-JWIN+1
      RH=SQRT(.5)
      KS1=NM+NA-1; KS2=KS1+NW-1
      NMM=NM-1; NMP=NM+1; NLM=NL+KS2
      NNM=NN-1; NNP=NN+1; NAP=NA+1
      NWM=NW-1; NWP=NW+1; NWIM=NWI-1; NWIP=NWI+1
      JWINM=JWIN-1
      WNH=CWH*W; WN=WNH+WNH; WNP=1.-WN
      DO 1 IF=1,NF
    1 RKF(IF)=1./KF(IF)
    2 SP=SP2=0.; NP=NE=0
      DO 8 IW=1,NWI
      DO 7 ID=1,ND
      DO 6 IAD=1,NAD
      DO 5 IA=1,NAL
      KE(IA,IAD,ID,IW)=0
      DO 4 IF=1,NF
      EA(IA,IAD,ID,IW,IF)=E2(IA,IAD,ID,IW,IF)=EMX(IA,IAD,ID,IW,IF)=0.
    4 CONTINUE
    5 CONTINUE
    6 CONTINUE
    7 CONTINUE
    8 CONTINUE
      READ(5,5000) PE,D1,NS,SEED
5000  FORMAT(3F10.4,I10)
      IF(PE.EQ.0) GO TO 9999
      D2=1-D1 ; C=(.5-ABS(.5-D1))*CM ; MP=NM
      MPM=MP-1
      MA=NA ; MW=NW
      DO 499 M=1,NLM

          MPM=MP ; MP=MP+1
      IF(MP.EQ.NMP) MP=1
      IF(M.LT.NM) GO TO 10
      MA=MA+1
      IF(MA.EQ.NAP) MA=1
      IF(M.LT.KS1) GO TO 10
      MW=MW+1
      IF(MW.EQ.NWP) MW=1
      NE=NE+1
10    DO 199 NR=1,NN
      N=NNP-NR ; U=RAN(SEED)
      IF(U.GE.PE) GO TO 20
      IU=5
      GO TO 30
20    U=RAN(SEED) ; IU=4*U+1 ; L(1)=0
30    IF(IU.GE.3) L(1)=1
      L(2)=L(3)=L(4)=L(1)
      IF(IU.GE.4) GO TO 100
      U=RAN(SEED)
      IF(IU.EQ.2) GO TO 50
      JU=4*U+1
      IF(IU.NE.1) GO TO 40
      L(JU)=1
      GO TO 100
40    L(JU)=0
      GO TO 100
50    JU=6*U
      IF(JU.GE.3) GO TO 60
      L(1)=1 ; L(JU+2)=1
      GO TO 100
60    L(JU-1)=1
      IF(JU.EQ.5) JU=2
      L(JU)=1
100   T1=T2=T3=T4=.5
      IF(N.LT.NN) T1=P1(MP,N+1)
      IF(M.EQ.1) GO TO 110
      T3=P1(MPM,N)
      IF(N.LT.NN) T2=P1(MPM,N+1)
      IF(N.GT.1) T4=P1(MPM,N-1)
110   WA=(T1*L(1)+T3*L(3)+(T2*L(2)+T4*L(4))*RH)/(L(1)+L(3)+
     >(L(2)+L(4))*RH)
      IF(IU.EQ.5) GO TO 120
      Q=IU*W ; Q=GAUSS(SEED,WA,Q) ; GO TO 150
```

```
120   U=RAN(SEED)
      IF(WA.LT.WN) GO TO 130
      IF(WA.GT.WNP) GO TO 140
      Q=WNP*U
      IF(U.GT.WA) Q=Q+WN
      GO TO 150
130   Q=(WA+WNH)*(1.-U)+U
      GO TO 150
140   Q=(WA-WNH)*U
150   P1(MP,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
      IF(M.EQ.1) GO TO 199
      T1=P1(MPM,N)
      T3=P1(MP,N)
      T2=T1-T3
      Q=T1-D1*T2
      P2(MPM,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
199   CONTINUE
6000  FORMAT(2(1X,10(F5.3,1X)))
200   IF(M.LT.NM) GO TO 499
      SIP=SQRT((SP2-SP*SP/NP)/NP) ; SIN=NS*SIP
      IF(M.GT.NM) GO TO 210
      DO 209 MM=1,NMM
      DO 208 N=1,NNM
      T1=P1(MM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MM,N)=PL1(N)=T1
      P2(MM,N)=PL2(N)=T2
208   CONTINUE
      CALL IMAGE(8,PL1,PL2)
209   CONTINUE
      GO TO 220
210   DO 219 N=1,NNM
      T1=P1(MPM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MPM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MPM,N)=PL1(N)=T1
      P2(MPM,N)=PL2(N)=T2
219   CONTINUE
      CALL IMAGE(8,PL1,PL2)
  220 DO 498 IA=1,NAL


      DO 497 IAD=1,NAD





      DO 496 ID=1,ND
      KSH=KD(ID) ; KS=2**KSH
      IF(M.LE.KS1) GO TO 230
      MAM=MA+JWINM
      IF(MAM.GT.NA) MAM=MAM-NA
      DO 229 IW=1,NWI
      MAM=MAM-1
      IF(MAM.LE.0) MAM=MAM+NA
      IASN(IA,IAD,ID,IW)=IASN(IA,IAD,ID,IW)-ILN(MAM,IA,IAD,ID,IW)
      IASD(IA,IAD,ID,IW)=IASD(IA,IAD,ID,IW)-ILD(MAM,IA,IAD,ID,IW)
229   CONTINUE
  230 IS1=IS2=0
      DO 249 N=1,JWIN

                                                     IS1=IS1+I4

                                 IS2=IS2+I4
249   CONTINUE
      ILN(MA,IA,IAD,ID,1)=IS1
      ILD(MA,IA,IAD,ID,1)=IS2
      IF(NWI.EQ.1) GO TO 260
      DO 259 IWM=1,NWIM
      N=JWIN+IWM ; IW=IWM+1

                                                     IS1=IS1+I4

                                 IS2=IS2+I4
      ILN(MA,IA,IAD,ID,IW)=IS1
      ILD(MA,IA,IAD,ID,IW)=IS2
```

```
259    CONTINUE
260    IF(M.LT.KS1) GO TO 496
       IF(M.GT.KS1) GO TO 280
       DO 279 IWM=1,NWI
       IS1=IS2=0
       IW=NWIP-IWM
       DO 269 MAM=IWM,JWIX
       IS1=IS1+ILN(MAM,IA,IAD,ID,IW)
       IS2=IS2+ILD(MAM,IA,IAD,ID,IW)
   269 CONTINUE
       IASN(IA,IAD,ID,IW)=IS1
       IASD(IA,IAD,ID,IW)=IS2
   279 CONTINUE
       GO TO 300
280    DO 289 IW=1,NWI
       IASN(IA,IAD,ID,IW)=IASN(IA,IAD,ID,IW)+ILN(MA,IA,IAD,ID,IW)
       IASD(IA,IAD,ID,IW)=IASD(IA,IAD,ID,IW)+ILD(MA,IA,IAD,ID,IW)
289    CONTINUE
290    IF(M.LE.KS2) GO TO 300
       DO 299 IW=1,NWI
       DO 299  IF=1,NF
       MM=MW-KF(IF)$ RF=RKF(IF)
       IF(MM.LE.0) MM=MM+NW
       FR(IA,IAD,ID,IW,IF)=FR(IA,IAD,ID,IW,IF)-WR(MM,IA,IAD,ID,IW)*RF
299    CONTINUE
   300 DO 495 IW=1,NWI
       T1=IASN(IA,IAD,ID,IW) $ T2=IASD(IA,IAD,ID,IW)
       T3=ABS(T2)
       IF(T3.GT..5*ABS(T1).AND.T3.GE.5.) GO TO 301
                      GO TO 302
   301 KE(IA,IAD,ID,IW)=KE(IA,IAD,ID,IW)+1
   302 WR(MW,IA,IAD,ID,IW)
       IF(M.LT.KS2) GO TO 495
       I1=0$ S1=0.$ MM=NWP
       DO 494 IF=1,NF
       RF=RKF(IF)
       IF(M.GT.KS2) GO TO 330
       I2=KF(IF) $ I3=I2-I1 $ I1=I2
       DO 309 I=1,I3
       MM=MM-1 $ S1=S1+WR(MM,IA,IAD,ID,IW)
309    CONTINUE
       S1=S1*RF
       FR(IA,IAD,ID,IW,IF)=S1
       GO TO 340
330    S1=FR(IA,IAD,ID,IW,IF)+WR(MW,IA,IAD,ID,IW)*RF
       FR(IA,IAD,ID,IW,IF)=S1
340    S1=S1-D1
       EA(IA,IAD,ID,IW,IF)=EA(IA,IAD,ID,IW,IF)+S1
       E2(IA,IAD,ID,IW,IF)=E2(IA,IAD,ID,IW,IF)+S1*S1
       S2=EMX(IA,IAD,ID,IW,IF) $ S2=ABS(S2)
       IF(ABS(S1).GT.S2) EMX(IA,IAD,ID,IW,IF)=S1
   494 CONTINUE
   495 CONTINUE
   496 CONTINUE
   497 CONTINUE
   498 CONTINUE
   499 CONTINUE
       SP=SP/NP
       WRITE(6,6001) PE,D1,SP,SIP,NS
6001   FORMAT(' THE ABOVE SCENES WERE GENERATED WITH ',F5.3,
      >' PROBABILITY OF EDGE '/' RELATIVE SHIFT IS ',F5.3,' PIXELS'/
      >' MEAN INTENSITY IS ',F5.3,/' STD DEV IS ',F5.3,/' 1./SNR IS '
      >,F5.3)
       RNE=1./NE
       DO 599 ID=1,ND
       DO 599 IA=1,NAL
       DO 599 IAD=1,NAD
       WRITE(6,6002) KD(ID),IA,IAD,(KF(IF),IF=1,NF)
6002   FORMAT(///' PIPELINE CORRELATION RESULTS: '/' DIGITIZING TO '
      >,I2,' BITS'/' ALGORITHM ',I1,' TYPE ',I1//' RUNNING AVE ',
      >'LENGTH  ',I1/' WINDOW')
       I1=JWIN-1
       DO 599 IW=1,NWI
       I1=I1+1
       DO 598 IF=1,NF
       S1=EA(IA,IAD,ID,IW,IF)
       SDF(IF)=SQRT((E2(IA,IAD,ID,IW,IF)-S1*S1*RNE)*RNE)
598    CONTINUE
       WRITE(6,6003) I1,I1,(EA(IA,IAD,ID,IW,IF),E2(IA,IAD,ID,IW,IF),
      >EMX(IA,IAD,ID,IW,IF),SDF(IF),IF=1,NF),KE(IA,IAD,ID,IW)
6003   FORMAT(I3,'X',I2,1(' !',4F7.3),' !',I4)
599    CONTINUE
       GO TO 2
9999   STOP
       END


15.30.37 >
```

STAGE III

```
C     PROGRAM: PIPELINE CORRELATION
C     CAI DIVISON RECON OPTICAL
C     D. DEFOE
C     DATE: 10 FEBRUARY 1982
C
      INTEGER SEED
      REAL NS
      DIMENSION P1(6,9), P2(6,9), ILN(8,3,3),
     >IASN(3,3), IASD(3,3,2), WR(8,3,3), L(4),
     >EA(3,3), E2(3,3), EMX(3,3),
     >KD(3), ILD(8,3,3,2), FR(3,3),
     >KE(3,3)
      DATA KD/6,4,2/
      NM=6; NL=100; NW=8; ND=3
      JWIN=6; JWIX=8
      CM=.25; W=.04; CWH=4.
      NA=JWIX; NN=JWIX+1; NWI=JWIX-JWIN+1
      RH=SQRT(.5)
      KS1=NM+NA-1; KS2=KS1+NW-1
      NMM=NM-1; NMP=NM+1; NLM=NL+KS2-1
      NNM=NN-1; NNP=NN+1; NAP=NA+1
      NWM=NW-1; NWP=NW+1; NWIM=NWI-1; NWIP=NWI+1
      JWINM=JWIN-1
      WNH=CNH*W; WN=WNH+WNH; WNP=1.-WN
      KF=8; RF=1./KF
    1 SP=SP2=0.; NP=NE=0
      DO 9 IW=1,NWI
      DO 8 ID=1,ND
      KE(ID,IW)=0
      EA(ID,IW)=E2(ID,IW)=EMX(ID,IW)=0.
    8 CONTINUE
    9 CONTINUE
      READ(5,5000) PE,D1,NS,SEED
 5000 FORMAT(3F10.4,I10)
      IF(PE.EQ.0) GO TO 9999
      D2=1-D1 ; C=(.5-ABS(.5-D1))*CM
      MP=NM; MPM=MP-1;
      MA=NA ; MW=NW
      DO 499 M=1,NLM

                                        MPM=MP
      MP=MP+1
      IF(MP.EQ.NMP) MP=1
      IF(M.LT.NM) GO TO 10
      MA=MA+1
      IF(MA.EQ.NAP) MA=1
      IF(M.LT.KS1) GO TO 10
      MW=MW+1
      IF(MW.EQ.NWP) MW=1
      IF(M.LT.KS2) GO TO 10
      NE=NE+1
 10   DO 199 NR=1,NN
      N=NNP-NR ; U=RAN(SEED)
      IF(U.GE.PE) GO TO 20
      IU=5
      GO TO 30
 20   U=RAN(SEED) ; IU=4*U+1 ; L(1)=0
 30   IF(IU.GE.3) L(1)=1
      L(2)=L(3)=L(4)=L(1)
      IF(IU.GE.4) GO TO 100
      U=RAN(SEED)
      IF(IU.EQ.2) GO TO 50
      JU=4*U+1
      IF(IU.NE.1) GO TO 40
      L(JU)=1
      GO TO 100
 40   L(JU)=0
      GO TO 100
 50   JU=6*U
      IF(JU.GE.3) GO TO 60
      L(1)=1 ; L(JU+2)=1
      GO TO 100
 60   L(JU-1)=1
      IF(JU.EQ.5) JU=2
      L(JU)=1
 100  T1=T2=T3=T4=.5
      IF(N.LT.NN) T1=P1(MP,N+1)
      IF(M.EQ.1) GO TO 110
      T3=P1(MPM,N)
      IF(N.LT.NN) T2=P1(MPM,N+1)
      IF(N.GT.1) T4=P1(MPM,N-1)
 110  WA=(T1*L(1)+T3*L(3)+(T2*L(2)+T4*L(4))*RH)/(L(1)+L(3)+
     >(L(2)+L(4))*RH)
      IF(IU.EQ.5) GO TO 120
      Q=IU*W ; Q=GAUSS(SEED,WA,Q) ; GO TO 150
 120  U=RAN(SEED)
      IF(WA.LT.WN) GO TO 130
      IF(WA.GT.WNP) GO TO 140
      Q=WNP*U
      IF(U.GT.WA) Q=Q+WN
      GO TO 150
```

```
 130  Q=(WA+WNH)*(1.-U)+U
      GO TO 150
 140  Q=(WA-WNH)*U
 150  P1(MP,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
      IF(M.EQ.1) GO TO 199
      T1=P1(MPM,N)
      T3=P1(MP,N)
      T2=T1-T3
      Q=T1-D1*T2
      P2(MPM,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
 199  CONTINUE
 6000 FORMAT(2(1X,10(F5.3,1X)))
 200  IF(M.LT.NM) GO TO 499
      SIP=SQRT((SP2-SP*SP/NP)/NP) ; SIN=NS*SIP
      IF(M.GT.NM) GO TO 210
      DO 209 MM=1,NMM
      DO 208 N=1,NNM
      T1=P1(MM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MM,N)=T1
      P2(MM,N)=T2
 208  CONTINUE
 209  CONTINUE
      GO TO 220
 210  DO 219 N=1,NNM
      T1=P1(MPM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MPM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MPM,N)=T1
      P2(MPM,N)=T2
 219  CONTINUE
  220 DO 496 ID=1,ND
      KS=4**KD(ID)
      IF(M.LE.KS1) GO TO 230
      MAM=MA+NWI
      IF(MAM.GT.NA) MAM=MAM-NA
      DO 229 IW=1,NWI
      MAM=MAM-1
      IF(MAM.LE.0) MAM=MAM+NA
      IASN(ID,IW)=IASN(ID,IW)-ILN(MAM,ID,IW)
       DO 229 JD=1,2
      IASD(ID,IW,JD)=IASD(ID,IW,JD)-ILD(MAM,ID,IW,JD)
 229  CONTINUE
  230 IS1=IS21=IS22=0
      DO 249 N=1,JWIN

                                          IS1=IS1+IS

                              IS21=IS21+IS

                              IS22=IS22+IS
 249  CONTINUE
      ILN(MA,ID,1)=IS1
      ILD(MA,ID,1,1)=IS21; ILD(MA,ID,1,2)=IS22
      IF(NWI.EQ.1) GO TO 260
      DO 259 IWM=1,NWIM
      N=JWIN+IWM ; IW=IWM+1

                                          IS1=IS1+IS

                              IS21=IS21+IS

                              IS22=IS22+IS
      ILN(MA,ID,IW)=IS1
      ILD(MA,ID,IW,1)=IS21; ILD(MA,ID,IW,2)=IS22
 259  CONTINUE
 260  IF(M.LT.KS1) GO TO 496
      IF(M.GT.KS1) GO TO 280
      DO 279 IWM=1,NWI
      IS1=IS21=IS22=0
      IW=NWIP-IWM
      DO 269 MAM=IWM,JWIX
      IS1=IS1+ILN(MAM,ID,IW)
      IS21=IS21+ILD(MAM,ID,IW,1); IS22=IS22+ILD(MAM,ID,IW,2)
  269 CONTINUE
      IASN(ID,IW)=IS1
      IASD(ID,IW,1)=IS21; IASD(ID,IW,2)=IS22
  279 CONTINUE
      GO TO 300
```

```
280    DO 289 IW=1,NWI
       IASN(ID,IW)=IASN(ID,IW)+ILN(MA,ID,IW)
       DO 289 JD=1,2
       IASD(ID,IW,JD)=IASD(ID,IW,JD)+ILD(MA,ID,IW,JD)
289    CONTINUE
290    IF(M.LE.KS2) GO TO 300
       DO 299 IW=1,NWI
       MM=MW-KF
       IF(MM.LE.0) MM=MM+NW
       FR(ID,IW)=FR(ID,IW)-WR(MM,ID,IW)*RF
299    CONTINUE
  300 DO 495 IW=1,NWI
       T1=IASN(ID,IW)
       IF(T1.GE.0.) GO TO 301
       T2=IASD(ID,IW,1)$ GO TO 302
  301 T2=IASD(ID,IW,2)
  302 T3=ABS(T2)
       IF(T3.GT..5*ABS(T1).AND.T3.GE.5.) GO TO 303
                        GO TO 304
  303 KE(ID,IW)=KE(ID,IW)+1
  304 WR(MW,ID,IW)
       IF(M.LT.KS2) GO TO 495
       S1=0.$ MM=NWP
       IF(M.GT.KS2) GO TO 330
       DO 309 I=1,KF
       MM=MM-1 $ S1=S1+WR(MM,ID,IW)
309    CONTINUE
       S1=S1*RF
       FR(ID,IW)=S1
       GO TO 340
330    S1=FR(ID,IW)+WR(MW,ID,IW)*RF
       FR(ID,IW)=S1
340    S1=S1-D1
       EA(ID,IW)=EA(ID,IW)+S1
       E2(ID,IW)=E2(ID,IW)+S1*S1
       S2=EMX(ID,IW) $ S2=ABS(S2)
       IF(ABS(S1).GT.S2) EMX(ID,IW)=S1
  495 CONTINUE
  496 CONTINUE
  499 CONTINUE
       SP=SP/NP
       WRITE(6,6001) PE,D1,SP,SIP,NS
6001   FORMAT(' THE SCENES WERE GENERATED WITH ',F5.3,
      >' PROBABILITY OF EDGE '/' RELATIVE SHIFT IS ',F5.3,' PIXELS'/
      >' MEAN INTENSITY IS ',F5.3,/' STD DEV IS ',F5.3,/' 1./SNR IS '
      >,F5.3)
       RNE=1./NE
       DO 599 ID=1,ND
       WRITE(6,6002) RD(ID),NE,KF
6002   FORMAT(///' PIPELINE CORRELATION RESULTS: '/' DIGITIZING TO '
      >,I2,' BITS'/' FOR',I5,' LINES'//' RUNNING AVE ',
      >'LENGTH  ',I1/' WINDOW')
       I1=JWIN-1
       DO 599 IW=1,NWI
       I1=I1+1
       S1=EA(ID,IW)
       S2=E2(ID,IW)
       SDF=SQRT((S2-S1*S1*RNE)*RNE)
       S1=S1*RNE$ S2=SQRT(S2*RNE)
       WRITE(6,6003) I1,I1,S1,S2,
      >EMX(ID,IW),SDF,KE(ID,IW)
6003   FORMAT(I3,'X',I2,1(' !',4F7.3),' !',I4)
599    CONTINUE
       GO TO 1
9999   STOP
       END

17.32.06 >
```

```
          lo@@printf pipec fortran
                                                  STAGE IV
C     PROGRAM: PIPELINE CORRELATION
C     CAI DIVISON RECON OPTICAL
C     D. DEFOE
C     DATE: 10 FEBRUARY 1982
C
      INTEGER SEED
      REAL NS
      DIMENSION P1(6,10), P2(6,10), ILN(9), ILD(9,2)
     >WR(8), L(4)
      KD=4; KS=256
      NM=6; NL=100; NW=8
      JWIN=JWIX=9
      CM=.25; W=.04; CWH=4.
      NA=JWIX; NN=JWIX+1; NWI=JWIX-JWIN+1
      RH=SQRT(.5)
      KS1=NM+NA-1; KS2=KS1+NW-1
      NMM=NM-1; NMP=NM+1; NLM=NL+KS2-1
      NNM=NN-1; NNP=NN+1; NAP=NA+1
      NWM=NW-1; NWP=NW+1; NWIM=NWI-1; NWIP=NWI+1
      JWINM=JWIN-1
      WNH=CNH*W; WN=WNH+WNH; WNP=1.-WN
      KF=8; RF=1./KF
    1 SP=SP2=0.; NP=NE=0
      KE=0; EA=E2=EMX=0.
      READ(5,5000) PE,D1,NS,SEED
5000  FORMAT(3F10.4,I10)
      IF(PE.EQ.0) GO TO 9999
      D2=1-D1 ; C=(.5-ABS(.5-D1))*CM
      MP=NM; MPM=MP-1
      MA=NA ; MW=NW
      DO 499 M=1,NLM

                                 MPM=      MP
      MP=MP+1
      IF(MP.EQ.NMP) MP=1
      IF(M.LT.NM) GO TO 10
      MA=MA+1
      IF(MA.EQ.NAP) MA=1
      IF(M.LT.KS1) GO TO 10
      MW=MW+1
      IF(MW.EQ.NWP) MW=1
      IF(M.LT.KS2) GO TO 10
      NE=NE+1
10    DO 199 NR=1,NN
      N=NNP-NR ; U=RAN(SEED)
      IF(U.GE.PE) GO TO 20
      IU=5
      GO TO 30
20    U=RAN(SEED) ; IU=4*U+1 ; L(1)=0
30    IF(IU.GE.3) L(1)=1
      L(2)=L(3)=L(4)=L(1)
      IF(IU.GE.4) GO TO 100
      U=RAN(SEED)
      IF(IU.EQ.2) GO TO 50
      JU=4*U+1
      IF(IU.NE.1) GO TO 40
      L(JU)=1
      GO TO 100
40    L(JU)=0
      GO TO 100
50    JU=6*U
      IF(JU.GE.3) GO TO 60
      L(1)=1 ; L(JU+2)=1
      GO TO 100
60    L(JU-1)=1
      IF(JU.EQ.5) JU=2
      L(JU)=1
100   T1=T2=T3=T4=.5
      IF(N.LT.NN) T1=P1(MP,N+1)
      IF(M.EQ.1) GO TO 110
      T3=P1(MPM,N)
      IF(N.LT.NN) T2=P1(MPM,N+1)
      IF(N.GT.1) T4=P1(MPM,N-1)
110   WA=(T1*L(1)+T3*L(3)+(T2*L(2)+T4*L(4))*RH)/(L(1)+L(3)+
     >(L(2)+L(4))*RH)
      IF(IU.EQ.5) GO TO 120
      Q=IU*W ; Q=GAUSS(SEED,WA,Q) ; GO TO 150
120   U=RAN(SEED)
      IF(WA.LT.WN) GO TO 130
      IF(WA.GT.WNP) GO TO 140
      Q=WNP*U
      IF(U.GT.WA) Q=Q+WN
      GO TO 150
130   Q=(WA+WNH)*(1.-U)+U
      GO TO 150
140   Q=(WA-WNH)*U
150   P1(MP,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
      IF(M.EQ.1) GO TO 199
      T1=P1(MPM,N)
      T3=P1(MP,N)
      T2=T1-T3
      Q=T1-D1*T2
      P2(MPM,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
199   CONTINUE
6000  FORMAT(2(1X,10(F5.3,1X)))
```

```
200  IF(M.LT.NM) GO TO 499
     SIP=SQRT((SP2-SP*SP/NP)/NP) ; SIN=NS*SIP
     IF(M.GT.NM) GO TO 210
     DO 209 MM=1,NMM
     DO 208 N=1,NNM
     T1=P1(MM,N)+GAUSS(SEED,0.,SIN)
     T2=P2(MM,N)+GAUSS(SEED,0.,SIN)
     T1=AMAX1(AMIN1(T1,1.),0.)
     T2=AMAX1(AMIN1(T2,1.),0.)
     P1(MM,N)=T1
     P2(MM,N)=T2
208  CONTINUE
209  CONTINUE
     GO TO 220
210  DO 219 N=1,NNM
     T1=P1(MPM,N)+GAUSS(SEED,0.,SIN)
     T2=P2(MPM,N)+GAUSS(SEED,0.,SIN)
     T1=AMAX1(AMIN1(T1,1.),0.)
     T2=AMAX1(AMIN1(T2,1.),0.)
     P1(MPM,N)=T1
     P2(MPM,N)=T2
219  CONTINUE
  220 IF(M.LE.KS1) GO TO 230
     IASN=IASN-ILN(MA)
     IASD1=IASD1-ILD(MA,1); IASD2=IASD2-ILD(MA,2)
  230 IS1=IS21=IS22=0
     DO 249 N=1,JWIN

                                  IS1=IS1+IS

                    IS21=IS21+IS

                   IS22=IS22+IS
249  CONTINUE
     ILN(MA)=IS1
     ILD(MA,1)=IS21; ILD(MA,2)=IS22
260  IF(M.LT.KS1) GO TO 499
     IF(M.GT.KS1) GO TO 280
     IASN=IASD1=IASD2=0
     DO 269 MAM=1,JWIX
     IASN=IASN+ILN(MAM)
     IASD1=IASD1+ILD(MAM,1); IASD2=IASD2+ILD(MAM,2)
  269 CONTINUE
     GO TO 300
  280 IASN=IASN+ILN(MA)
     IASD1=IASD1+ILD(MA,1); IASD2=IASD2+ILD(MA,2)
290  IF(M.LE.KS2) GO TO 300
     MM=MW-KF
     IF(MM.LE.0) MM=MM+NW
     FR=FR-WR(MM)*RF
  300 T1=IASN
     IF(T1.GE.0.) GO TO 301
     T2=IASD1; GO TO 302
  301 T2=IASD2
  302 T3=ABS(T2)
     IF(T3.GT..5*ABS(T1).AND.T3.GE.5.) GO TO 303
                  GO TO 304
  303 KE=KE+1
  304 WR(MW)
     IF(M.LT.KS2) GO TO 499
     MM=NWP
     IF(M.GT.KS2) GO TO 330
     FR=0.
     DO 309 I=1,KF
     MM=MM-1 ; FR=FR+WR(MM)
309  CONTINUE
     FR=FR*RF
     GO TO 340
330  FR=FR+WR(MW)*RF
340  S1=FR-D1
     EA=EA+S1
     E2=E2+S1*S1
     S2=ABS(EMX)
     IF(ABS(S1).GT.S2) EMX=S1
  499 CONTINUE
     SP=SP/NP
     WRITE(6,6001) PE,D1,SP,SIP,NS
6001 FORMAT(' THE SCENES WERE GENERATED WITH ',F5.3,
    >' PROBABILITY OF EDGE '/' RELATIVE SHIFT IS ',F5.3,' PIXELS'/
    >' MEAN INTENSITY IS ',F5.3,/' STD DEV IS ',F5.3,/' 1./SNR IS '
    >,F5.3)
     RNE=1./NE
     WRITE(6,6002) KD,NE,KF
6002 FORMAT(///' PIPELINE CORRELATION RESULTS: '/' DIGITIZING TO '
    >,I2,' BITS'/' FOR',I5,' LINES'//' RUNNING AVE ',
    >'LENGTH  ',I1/' WINDOW')
     I1=JWIN-1
     I1=I1+1
     SDF=SQRT((S2-S1*S1*RNE)*RNE)
     EA=EA*RNE; E2=SQRT(E2*RNE)
     WRITE(6,6003) I1,I1,EA,E2,EMX,SDF,KE
6003 FORMAT(I3,'X',I2,1(' !',4F7.3),' !',I4)
     GO TO 1
9999 STOP
     END
```

STAGE V

```
16.20.24 >printf pipec fortran

C      PROGRAM: PIPELINE CORRELATION
C      CAI DIVISON RECON OPTICAL
C      D. DE?OE
C      DATE: 10 FEBRUARY 1982
C
       INTEGER SEED
       REAL NS
       DIMENSION P1(6,10), P2(6,10), ILN(9), ILD(9,2), ILDZ(9),
      >WR(8), WRZ(8) L(4)
       KD=4; KS=256
       NM=6; NL=100; NW=8
       JWIN=JWIX=9
       CM=.25; W=.04; CWH=4.
       NA=JWIX; NN=JWIX+1; NWI=JWIX-JWIN+1
       RH=SQRT(.5); R2=SQRT(2.)
       KS1=NM+NA-1; KS2=KS1+NW-1
       NMM=NM-1; NMP=NM+1; NLM=NL+KS2-1
       NNM=NN-1; NNP=NN+1; NAP=NA+1
       NWM=NW-1; NWP=NW+1; NWIM=NWI-1; NWIP=NWI+1
       JWINM=JWIN-1
       WNH=CNH*W; WN=WNH+WNH; WNP=1.-WN
       KF=8; RF=1./KF
     1 SP=SP2=0.; NP=NE=0
       KE=0; EA=E2=EMX=0.
       KEZ=0; EAZ=E2Z=EMXZ=0.
       READ(5,5000) PE,D1,NS,SEED
5000   FORMAT(3F10.4,I10)
       IF(PE.EQ.0) GO TO 9999
       D1H=D1*.5
       D2=1-D1 ; C=(.5-ABS(.5-D1))*CM
       D2H=D2*.5; D3=D1*D2
       MP=NM; MPM=MP-1
       MA=NA ; MW=NW
       DO 499 M=1,NLM

                                     MPM      =MP
       MP=MP+1
       IF(MP.EQ.NMP) MP=1
       IF(M.LT.NM) GO TO 10
       MA=MA+1
       IF(MA.EQ.NAP) MA=1
       IF(M.LT.KS1) GO TO 10
       MW=MW+1
       IF(MW.EQ.NWP) MW=1
       IF(M.LT.KS2) GO TO 10
       NE=NE+1
10     DO 199 NR=1,NN
       N=NNP-NR ; U=RAN(SEED)
       IF(U.GE.PE) GO TO 20
       IU=5
       GO TO 30
20     U=RAN(SEED) ; IU=4*U+1 ; L(1)=0
30     IF(IU.GE.3) L(1)=1
       L(2)=L(3)=L(4)=L(1)
       IF(IU.GE.4) GO TO 100
       U=RAN(SEED)
       IF(IU.EQ.2) GO TO 50
       JU=4*U+1
       IF(IU.NE.1) GO TO 40
       L(JU)=1
       GO TO 100
40     L(JU)=0
       GO TO 100
50     JU=6*U
       IF(JU.GE.3) GO TO 60
       L(1)=1 ; L(JU+2)=1
       GO TO 100
60     L(JU-1)=1
       IF(JU.EQ.5) JU=2
       L(JU)=1
100    T1=T2=T3=T4=.5
       IF(N.LT.NN) T1=P1(MP,N+1)
       IF(M.EQ.1) GO TO 110
       T3=P1(MPM,N)
       IF(N.LT.NN) T2=P1(MPM,N+1)
       IF(N.GT.1) T4=P1(MPM,N-1)
```

```
110   WA=(T1*L(1)+T3*L(3)+(T2*L(2)+T4*L(4))*RH)/(L(1)+L(3)+
     >(L(2)+L(4))*RH)
      IF(IU.EQ.5) GO TO 120
      Q=IU*W ; Q=GAUSS(SEED,WA,Q) ; GO TO 150
120   U=RAN(SEED)
      IF(WA.LT.WN) GO TO 130
      IF(WA.GT.WNP) GO TO 140
      Q=WNP*U
      IF(U.GT.WA) Q=Q+WN
      GO TO 150
130   Q=(WA+WNH)*(1.-U)+U
      GO TO 150
140   Q=(WA-WNH)*U
150   P1(MP,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
      IF(M.EQ.1) GO TO 199
      T1=P1(MPM,N)
      T3=P1(MP,N)
      T2=T1-T3
      Q=T1-D1*T2
      P2(MPM,N)=Q ; SP=SP+Q
      SP2=SP2+Q*Q ; NP=NP+1
199   CONTINUE
6000  FORMAT(2(1X,10(F5.3,1X)))
200   IF(M.LT.NM) GO TO 499
      SIP=SQRT((SP2-SP*SP/NP)/NP) ; SIN=NS*SIP
      IF(M.GT.NM) GO TO 210
      DO 209 MM=1,NMM
      DO 208 N=1,NNM
      T1=P1(MM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MM,N)=T1
      P2(MM,N)=T2
208   CONTINUE
209   CONTINUE
      GO TO 220
210   DO 219 N=1,NNM
      T1=P1(MPM,N)+GAUSS(SEED,0.,SIN)
      T2=P2(MPM,N)+GAUSS(SEED,0.,SIN)
      T1=AMAX1(AMIN1(T1,1.),0.)
      T2=AMAX1(AMIN1(T2,1.),0.)
      P1(MPM,N)=T1
      P2(MPM,N)=T2
219   CONTINUE
  220 IF(M.LE.KS1) GO TC 230
      IASN=IASN-ILN(MA)
      IASD1=IASD1-ILD(MA,1); IASD2=IASD2-ILD(MA,2)
      IASDZ=IASDZ-ILDZ(MA)
  230 IS1=IS21=IS22=0
      IS2Z=0
      DO 249 N=1,JWIN

                                      IS1=IS1+IS

                          IS21=IS21+IS


                         IS22=IS22+IS
249   CONTINUE
      ILN(MA)=IS1
      ILD(MA,1)=IS21; ILD(MA,2)=IS22
      ILDZ(MA)=IS2Z
260   IF(M.LT.KS1) GO TO 499
      IF(M.GT.KS1) GO TO 280
      IASN=IASD1=IASD2=0
      IASDZ=0
      DO 269 MAM=1,JWIX
      IASN=IASN+ILN(MAM)
      IASD1=IASD1+ILD(MAM,1); IASD2=IASD2+ILD(MAM,2)
      IASDZ=IASDZ+ILDZ(MAM)
  269 CONTINUE
      GO TO 300
  280 IASN=IASN+ILN(MA)
      IASD1=IASD1+ILD(MA,1); IASD2=IASD2+ILD(MA,2)
      IASDZ=IASDZ+ILDZ(MA)
 290  IF(M.LE.KS2) GO TO 300
      MM=MW-KF
      IF(MM.LE.0) MM=MM+NW
      FR=FR-WR(MM)*RF
      FRZ=FRZ-WRZ(MM)*RF
  300 T1=T4=IASN
      IF(T1.GE.0.) GO TO 301
      T2=IASD1; GO TO 302
  301 T2=IASD2
  302 T3=ABS(T2)
      IF(T3.GT..5*ABS(T1).AND.T3.GE.5.) GO TO 303
                     GO TO 304
  303 KE=KE+1
  304 WR(MW)
      T2=IASDZ; T3=ABS(T2)
      IF(T3.GT.ABS(T4).AND.T3.GE.5.) GO TO 305
                     GO TO 306
  305 KEZ=KEZ+1
  306 WRZ(MW)
      IF(M.LT.KS2) GO TO 499
      MM=NWP
      IF(M.GT.KS2) GO TO 330
      FR=0.
      FRZ=0.
      DO 309 I=1,KF
      MM=MM-1 ; FR=FR+WR(MM)
      FRZ=FRZ+WRZ(MM)
 309  CONTINUE
      FR=FR*RF
      FRZ=FRZ*RF
      GO TO 340
 330  FR=FR+WR(MW)*RF
      FRZ=FRZ+WRZ(MW)*RF
 340  S1=FR-D1
      EA=EA+S1
      E2=E2+S1*S1
      S2=ABS(EMX)
      IF(ABS(S1).GT.S2) EMX=S1
      S1Z=FRZ-D1; EAZ=EAZ+S1Z; E2Z=E2Z+S1Z*S1Z
      S2Z=ABS(EMXZ); IF(ABS(S1Z).GT.S2Z)EMXZ=S1Z
  499 CONTINUE
      SP=SP/NP
      WRITE(6,6001) PE,D1,SP,SIP,NS
 6001 FORMAT(' THE SCENES WERE GENERATED WITH ',F5.3,
     >' PROBABILITY OF EDGE '/' RELATIVE SHIFT IS ',F5.3,' PIXELS'/
     >' MEAN INTENSITY IS ',F5.3,/' STD DEV IS ',F5.3,/' 1./SNR IS '
     >,F5.3)
      RNE=1./NE
      WRITE(6,6002) KD,NE,KF
 6002 FORMAT(////' PIPELINE CORRELATION RESULTS: '/' DIGITIZING TO '
     >,I2,' BITS'/' FOR',I5,' LINES'//' RUNNING AVE ',
     >'LENGTH  ',I1/' WINDOW')
      I1=JWIN-1
      I1=I1+1
      EA=EA*RNE; E2=E2*RNE; SDF=SQRT(E2-EA*EA); E2=SQRT(E2)
      EAZ=EAZ*RNE; E2Z=E2Z*RNE; SDFZ=SQRT(E2Z-EAZ*EAZ); E2Z=SQRT(E2Z)
      WRITE(6,6003) I1,I1,EA,E2,EMX,SDF,KE,EAZ,E2Z,EMXZ,SDFZ,KEZ
 6003 FORMAT(I3,'X',I2,2(' !',4F7.3,' !',I4))
      GO TO 1
 9999 STOP
      END


16.22.11 >los
328.81 ARU'S, .85 CONNECT HRS
LOGGED OFF AT 16.22.33 ON  07MAR82T(~
```

ATE

LME