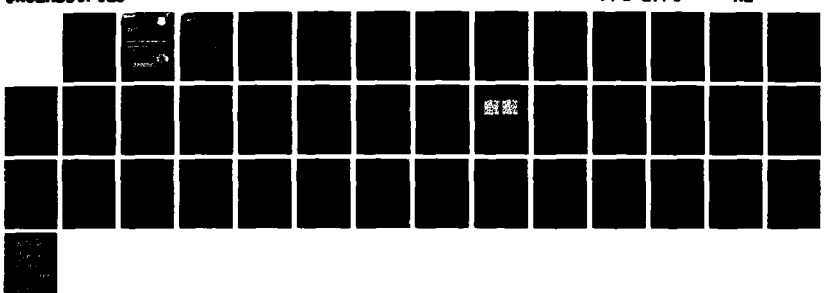
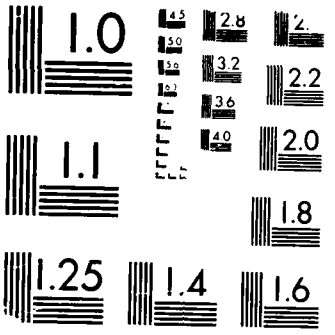
AD-A192 529 RESEARCH ON WIDE AREA WORKSTATIONS(U) DAYTON UNIV OH RESEARCH INST J T KAJIYA MAR 88 RADC-TR-87-226 F30602-81-C-0206 1/1

UNCLASSIFIED F/G 17/5 NL





AD-A192 529

RADC-TR-87-226
Final Technical Report
March 1988

RESEARCH ON WIDE AREA WORKSTATIONS

California Institute of Technology

Dr. James T. Kajiya

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-87-226 has been reviewed and is approved for publication.

APPROVED: Mark R Rosiek

MARK R. ROSIEK
Project Engineer

APPROVED: Garry W. Barringer

GARRY W. BARRINGER
Technical Director
Directorate of Intelligence & Reconnaissance

FOR THE COMMANDER: James W. Hyde III

JAMES W. HYDE III
Directorate of Plans & Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (IRRP) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

A192 529

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS: N/A
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): N/A	5. MONITORING ORGANIZATION REPORT NUMBER(S): RADC-TR-87-226
6a. NAME OF PERFORMING ORGANIZATION: California Institute of Technology; 6b. OFFICE SYMBOL (If applicable):	7a. NAME OF MONITORING ORGANIZATION: Rome Air Development Center (IRRP)
6c. ADDRESS (City, State, and ZIP Code): Computer Science 256-80, Pasadena CA 91125	7b. ADDRESS (City, State, and ZIP Code): Griffiss AFB NY 13441-5700
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Rome Air Development Center; 8b. OFFICE SYMBOL (If applicable): IRRP	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F30602-81-C-0206
8c. ADDRESS (City, State, and ZIP Code): Griffiss AFB NY 13441-5700	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO 62702F; PROJECT NO 4594; TASK NO 17; WORK UNIT ACCESSION NO P2

11. TITLE (Include Security Classification): RESEARCH ON WIDE AREA WORKSTATIONS

12. PERSONAL AUTHOR(S): Dr. James T. Kajiya

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM Jul 85 TO Jul 86	March 1988	44

16. SUPPLEMENTARY NOTATION: N/A

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB GROUP	digital images, image workstations, imagery interpretation, remote sensing
09	02		
12	01		

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

The use of dynamic random access memory (DRAM) for storing large digital images in a workstation environment was studied. The need and advantages of a workstation with access to a digital image stored in semiconductor memory are discussed. An architecture block diagram is shown and used to show the tradeoffs in designing a wide area workstation. Algorithms that take advantage of the wide image area for image warping, correlation, panning, roaming, and zooming are shown.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL: MARK R. BUSIEK	22b. TELEPHONE (Include Area Code): (315) 330-3144	22c. OFFICE SYMBOL: RADC (IRRP)

DD Form 1473, JUN 86 — Previous editions are obsolete — SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

This final report for the RADC PostDoctoral Subcontract is through the University of Dayton RI-40615. This document consists of four chapters. The first is a report on the wide area workstation which was delivered previously to Mr. Mark Rosiek of RADC. The second chapter is a report on the analysis of the architecture reported in the first chapter. The third chapter discusses an algorithm for the automatic stereo correlation of terrain images. The fourth chapter describes a new set of warp interpolation algorithms which will be useful to process the output of automatic stereo correlation algorithms.

CHAPTER 1
WIDE AREA WORKSTATION ARCHITECTURE

Section 1. The need for a wide area workstation

A wide area workstation capable of manipulating large images is a key component in modern cartographic and intelligence activities. The source imagery for these endeavors is needed at very high resolutions in order to effectively classify cultural features. Coupled with the requirement to cover wide geographical areas at high resolutions pushes the number pixels in a typical source image to very large counts. For example, even low resolution LANDSAT imagery runs 16K square pixels or more. A workstation capable of handling such images in ways useful to photointerpreters is a key requirement of many future government systems.

Unfortunately, the need to manipulate large, high resolution images does not arise in the commercial sector. The principal uses for image display devices are in computer graphics, computer aided design, and image processing. Commercial offerings of image workstations may reach 4K by 4K with a displayable segment of 1K by 1K. These images typically have many bits per pixel, (8-12, sometimes 24) and offer pixel roam but no continuous zoom or rotation. In other words, all commercial offerings assume that images will be very small. Systems that must offer large images typically segment a large image stored on disk. This implies that the user must suffer long update latencies when it is desired to display a portion of the image not buffered in fast memory.

Section 2. Discussion of tradeoffs

The principal tradeoff in in designing a wide area workstation is the

question of local buffering. How much local buffering is to be provided by the workstation? At one extreme is small local buffers, holding say only a viewable (1K by 1K) image. In this configuration, low update latencies imply that the communication bandwidth to the display station from the full image store must be very large. The operator of the workstation may require a large number of image changes quickly. To store the full image on rotating magnetic media implies that access latencies will be very large. In order to mask such latencies, a complex software and hardware system is needed to predict which data blocks will possibly called by the operator. Such system complexity is very expensive, both in terms of design and implementation. For this reason, we advocate the opposite extreme tradeoff.

Large local buffering allows the bandwidth to the image station from the global image store to be much smaller. Furthermore, access latencies to the central image store may be quite long. If an entire image is stored in fast dynamic RAM (DRAM) then no complex data access prediction is required. Thus, from a hardware and software design complexity viewpoint, large local buffering is preferable. The only factor mitigating the advantages of large local buffering is the cost of local store.

The economics of DRAMs has a dramatic history of cost decreases. In 1974 the 4Kb dynamic ram became available. Just prior to the introduction of this chip, Gordon Moore of Intel enunciated "Moore's Law": memory chips would be quadrupling in size every three years. The law has held true to this day. In 1977 the 16Kb ram was introduced, in 1980 the 64Kb, in 1983 the 256Kb, and in 1986 the 1Mb ram. The cost per bit of dynamic ram storage has been decreasing exponentially at a rate of 26% . Trade considerations have temporarily distorted this trend. However, the basic economic and physical parameters remain in place. The long term trend will be reasserted when trade restrictions are lifted.

To store a 16K by 16K image requires 256 MByte of storage. In 1974 a large image store would have been technologically infeasible, requiring half a million chips. Today the required number is 8,000 chips for a total cost of some $16,000. In late 1986 the 1Mb ram will be available.

Only 2,000 chips will be required for a full image store. The current research literature is reporting on 4Mb chips in the laboratory. The next ten years should see an equally dramatic cost decrease.

In terms of advancing technology, semiconductor memory is the best bargain known to man. Nothing—not disk, tape, computing cycles, network bandwidth—will grow as dramatically in price performance terms as semiconductor memory. It is easy to see that a good workstation architecture is one which judiciously trades off these other resources for semiconductor memory as much as possible. Clearly, the economics of the semiconductor industry favor a large buffer approach. This is the approach to the wide area workstation outlined here. Whenever a tradeoff exists, we will take the approach of using larger memories.

Section 3. Algorithms to reduce data bandwidth.

Proper display of antialiased images when significant resampling is required requires filtering of the images. Filtering is required whenever one of the following occurs: 1) a pan of the image on subpixel boundaries; 2) Zooms; 3) Rotation of the image at other than 90 degree angles. Given a full resolution image, it is necessary to convolve this image with a convolution kernel effecting a low pass filter. This eliminates frequencies above the Nyquist rate so that the image may be down sampled without creating aliases.

Directly convolving an image for downsampling can be very expensive. If an entire 16K by 16K image were to be downsampled to a 512 by 512 pixel display a decimation factor of 32 would be required. This requires an interpolation filter of at least 64 by 64 pixels. To compute a single output pixel requires 4096 multiplies and 4096 addtitions.

But we can downsample the image in stages, that is, filtering with a single large low pass kernel is equivalent to filtering with a cascade of smaller low pass kernels. Consider the case of downsampling in two stages. If we first decimate by 4 times, the interpolation kernel required

is only 8 by 8, requiring 64 multiplies and 64 adds. Decimating the resulting image again by a factor of 8 requires 256 multiplies and 256 adds. Thus the number of required multiplication and addition operations drops from 8192 to 640 operations.

Thus filtering and decimating in stages lowers the computational bandwidth because there are fewer and fewer pixels participating in the computation.

Minimizing the number of computations is not the only objective, however. If we may precompute and store intermediate data, then only the final stages of computation need be performed in the time critical step.

The principal technique we use for precomputing decimation data is the pyramidal image technique (Tanimono and Pavlidis 1975). The pyramid image technique stores an image as a hierarchy of images which have been decimated by a factor of two. The storage required is

$$n^2 + \frac{n^2}{4} + \frac{n^2}{16} + \cdots = \frac{4}{3}n^2$$

A 33% overhead in storage. To decimate the image by a factor other than a power of two we use the schemes due to Williams(1983).

To interpolate an image to other than a factor of two we find the closest pair of powers of two and interpolate the sample corresponding to each power.

Section 4. Architecture Block diagram and description of it.

The block diagram of the wide area workstation appears in figure 1. It is built around a 32 bit bus which can transfer four eight bit pixels at once. Side channels for pixel data provide high speed data transfer in the image chain. The architecture is built around a large multi-port memory, a video access chain and an interpolation/reformatting

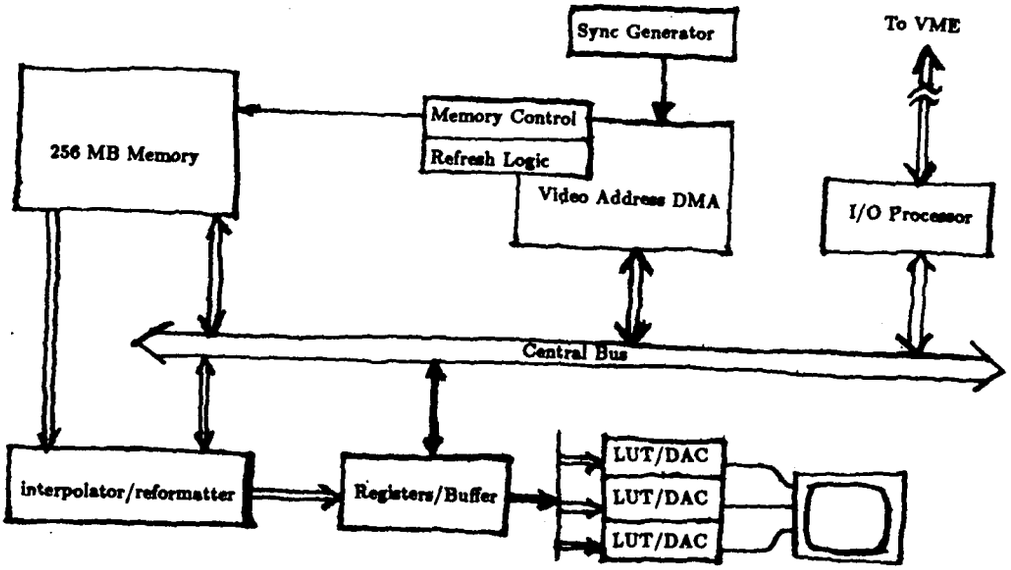


Figure 1. Architecture Block Diagram

chain.

Section 5. The video access chain.

The video access chain begins with a video sync generator. The sync generator provides timing signals which provide the memory access controller with enough information to synch to a television screen. The memory access controller computes the proper addresses to be sent to the image interpolation/reformatting chain, which interpolates the image data according to a preset plan. The output of the interpo-

lation/reformatting chain then feeds the video output registers which drive a color map that provides the video.

The video synch generator consists of a standard LSI chip which is available from a number of sources. It provides timing signals for horizontal and vertical synch as well as clocks to indicate when a (sub)pixel has advanced.

The memory access controller computes the addresses used for accessing the memory. For an image of r rows and c columns, the address generated is given by

$$a = (r-1)cx + (c-1)y \qquad 0 \leq x, y \leq 1. \tag{1}$$

In the above equation, a is the pixel offset in the linear memory, and x,y are quantities between 0 and 1 which index a particular pixel on the screen. We have used a fractional quantity rather than the traditional integer indices because the former are scale independent.

On each successive level l of the pyramid, the row and column double in size. Thus the pixel offset a is given by

$$\begin{aligned} a &= \text{off}(l) + (2^{2l} - 2^l)x + (2^l - 1)y \qquad 0 \leq x, y \leq 1 \\ &= \text{off}(l) + (2^l - 1)(2^l x + y) \end{aligned} \tag{2}$$

where off(l) is the pixel offset of the image at level l. The off(l) function is simply a table of offsets, for a 16K square image there are only 14 levels and hence 14 entries in the table. Note that the above address computation requires only one add, one subtract, and two shift operations.

For each pixel on the screen, the address generator accesses two sets of four pixels. Each set of four pixels surrounds the pixel of interest. The two sets are given by the two levels which are immediately above and below the current zoom factor.

To generate each x and y value the roam, zoom, and rotation matrix enter into the calculation. Essentially the transformation matrices are concatenated together and two separate inverse zooms are finally applied to normalize coordinates for the two image levels. These matrices need only be computed once per frame, and since there are only nine numbers in each it is easy to compute them on a microprocessor. Once the two matrices are computed, the matrix elements are used to generate coefficients for a pair of DDAs which generate all (x, y) pairs for the scan lines.

Section 6. The interpolation/reformatting chain

The interpolation/reformatting chain takes blocks of pixels to be interpolated to be scanned out to the video output registers. It relies on the proper supply of two sets of four pixels.

To access an image at a level of zoom between two levels l_1 and l_2, we simply bilinearly interpolate each of the four pixel image sets at both levels and compute a weighted average. That is, the intensity p of a pixel is determined by the intensity of the sampled pixels p_1, p_2 from the pixel sets.

$$p = p_1 + \alpha \times (p_2 - p_1) \tag{3}$$

Where the weight α is determined by the zoom factor.

$$\alpha = 1 - \text{zoom}/2^{l_1}. \tag{4}$$

Thus when the zoom factor is exactly a power of two, we have $l_1 =$ zoom and $\alpha = 0$.

Section 7. The i/o chain.

The i/o chain provides a path from the host CPU to the memory array as well as to the control registers in each of the function units of the display system. The i/o processor provides a standard interface which is capable of direct memory access to and from the i/o bus—a good choice would be the VME bus because of its wide acceptance. Architecturally, the design of this piece is quite straighforward, being virtually identical to conventional memory interfaces.

Section 8. Working out of the bandwidths in the design.

For each output pixel the above scheme requires 2 bilinear interpolations and 1 linear interpolation(lirp). Each bilinear interpolation can be calculated via a cascade of three lirps. Thus each output pixel requires four lirps. For an output image resolution of 1024 square with an image update rate of 60 frames/sec, we require 8 Mlirp/sec. A lirp consists of two fixed point multiplies and one add where the number of bits is slightly more than the number of bits stored in the image. For 8 bits per pixel, 12 bits of precision is adequate. Thus, for real time interpolation, we require 16 million fixed point multiplies/sec and 8 million fixed point adds/sec. These figures are quite modest with respect to today's multiplier chips.

Generating the interpolation coefficients is a small bit of random logic combined with a simple table lookup and thus does not impose a significant computation burden.

Generating addresses for the four pixel blocks is given by equations (1) and (2). Notice that only shifts, adds, and table lookups are required. A small bit of random logic for generating these addresses can fit easily on a single gate array.

CHAPTER 2
WIDE AREA WORKSTATION

Architecture Analysis

This chapter provides an analysis of the architecture proposed in the last chapter

Section 1. Design with DRAMS

Why would one want to use such large quantities of semiconductor memory instead of using a medium sized disk drive? There are several reasons why the use of random access semiconductor memory is desirable in a wide area workstation.

Foremost is the ease of roam and zoom. The process of roaming through a large image that has been stored on disk is a complex process. Because disk access speeds are too slow to allow direct reads from the disk, disk buffers must be cleverly allocated. The pixels to come into view as the user roams throughout a scene need to be buffered in random access memory. The speed of the average disk read determines the speed S (in pixels per second) at which roaming is allowed. This is given by

$$S = \frac{R}{N_1}$$

where the displayed image is N_1 by N_1 pixels and R is the *average* rate (in words/sec) that the disk can do random block reads.

Thus, if we have a very fast disk or a very small image the roam rates are very high. For high resolution displays N_1 is large (from 1000 to 4000 pixels). The average access time for a typical disk is on the order of 3MB/sec. This would imply that for a 1000 by 1000 image we could roam at over 3000 pixels per second (or 3 frames per second!).

Unfortunately, these peak speeds are valid only under the assumption of perfect buffering. For real buffering the roam speed is limited by the random access time of the disk. The time to fill a random buffer is given by

$$T = \frac{1}{S} = \frac{N_1 A}{W} + \frac{W}{R}.$$

Where, W is the size of a disk buffer and A is the access time of a random block read, R is the read rate of the disk. The first term measures the time to seek a given block, the second time measures the time needed to transfer it. For typical disk numbers we may ignore the second quantity. Given a disk buffer size of W = 4KB with an access time A = 25ms. We get a roam rate of 160 pixels/sec.

Evidently , increasing the buffer size will increase the roam rate, but then an additional problem pops up. If we make the buffer size very large, the user may not decide to roam in the same direction for an appreciable amount of time. Thus with very large disk buffers, roaming requires a prediction of where the user will roam next.

One solution to the above problem is adopted in this architecture: make the buffer so large as to hold the entire image. Thus no prediction will be necessary as to where the user will roam.

Section 2. Why two datapaths?

There are two main datapaths in the architecture because there are two conflicting requirements. The first requirement is for a flexible central bus that can be used to access and set the state of the various subsystems in the workstation. This is the function of the central bus. Speed is not a requirement for this datapath, since it is primarily used for maintenance, initialization, and for communication with the host. This is a slow speed link and could easily be handled by conventional CMOS gate arrays or logic such as the Xilinx logic cell array, a RAM based programmable logic chip.

The second datapath is used to flow digital refresh data to the video

monitor. This is a high-speed (7-700Mhz) continuous data stream. We devote a separate data path to it. Because of the high bandwidth requirement of this chain, the implementation of the chain should be done in ECL F100K gate arrays.

The real bandwidth requirements of the high speed datapath depend on the resolution of the display screen and the update rate for the frame. There are several standard choices:

H	V	Interlaced Bandwidth(MHz)	Noninterlaced Bandwidth(MHz)
640	480	7.8	15.7
1024	1024	9.2	18.4
1280	1024	31.5	62.9
4000	3000	NA	720

The very high resolution display represented in the last line of the table is available in a display manufactured by MegaScan Inc., of Pittsburg. They currently have a black and white one bit per pixel display. By late 1987 they expect to offer a grey scale monitor with the same resolution. They have plans to offer a similar high resolution color display in the late 1988-1989 time frame.

Section 3. What is the bandwidth required for the memory chips?

Given a 4Mb memory technology, which should be available by 1988, we need 512 chips to store 256 MB of image data. Error correction for the pixel data causes an overhead of at 9 chips (since to specify one defective chip out of 512 requires 9 bits) depending on the error correction codes adopted. If we use the common (72,64) double error detection, single error correction code, we need 8 check bits on a 64 bit word. Organizing the data memory into an array of 8 by 64 chips requires 8 additional columns for an increment of 8 by 8=64 chips. Thus the total memory chip count would be 512+64=576 chips.

The bandwidth of 512 memory chips at a 100ns cycle time is enormous, 640 Mbytes/second. This is more than enough to service the video refresh data stream. Thus ordinary dynamic rams may be used, if the interpolation scheme is such that the image is evenly balanced among the memory chips. This is not possible using a wide range of zoom factors. Thus additional memory buffering is required in the interpolator/reformatter stage.

When 16Mb chips become available then the bandwidth of each individual chip often becomes an issue. The architecture outline here uses very fast static rams as a buffer in the interpolator/reformatter stage. This relaxes the bandwidth requirements on the main store.

Section 4. Why store multiple resolutions?

The workstation algorithm stores the image at several levels of resolution. This results in a storage overhead of almost one third, which is a considerable amount of memory for a quarter gigabyte store. Why does the architecture do this?

To view an image at lower resolution one may at first thought simply display every nth pixel. But as is well known this gives rise to the aliasing phenomenon, in which diagonal edges are given a jagged appearance. Furthermore, fine periodic structures in the image undergo large degradations. Thus when displaying a picture at lower resolution, a low pass filter operation is necessary. This operation may be as simple as averaging an $n \times n$ block which serves as the output pixel. However, better results may be obtained from a low pass filter operation which convolves the image with a convolution kernel. The convolution operation is defined as:

$$O_{i,j} = \sum_{k,l=-\frac{n}{2}}^{i,j=\frac{n}{2}} I_{i-k,j-l} h_{k,l}.$$

Where $O_{i,j}$ is the output pixel at position i, j, I is the input image, and h is the convolution kernel.

Now when a zoom factor of n is used there are n^2 multiplications and additions required. This makes the direct use of convolution too expensive for real time applications.

The workstation algorithm avoids this difficulty by approximating a full convolution by precomputing a number of convolutions at fixed resolutions and then combining them for the desired zoom. The reduction in the number of computations is discussed in the architecture document.

Section 5. Form factor.

Surprisingly the space requirements for a memory display of this kind are not large. Thanks to the advancing state of the art in packaging and VLSI—surface mount technology, high density ECL gate arrays, and integrated RAM DACS (such as offered by the Brooktree), the entire architecture will fit on one triple high double deep Eurocard with a piggyback card. This is the standard board size used in Sun Microsystems workstations. Thus the unit can fit into a single slot of a Sun workstation.

The space requirements for memory chips alone are moderate. In a surface mount DIP each chip requires approximately .33 square inches. Giving a total chip requirement of 225 square inches. Thus one piggyback board of size 10 by 12 inches with memory chips mounted on both sides will suffice for the memory array. The sun board is approximately 225 square inches which would suffice for the rest of the circuitry assuming high density gate arrays.

CHAPTER 3
An algorithm for the automatic estimation of warp functions

Section 1. Introduction

An Image I is warped by a function $\mathbf{x} = (\xi(x,y), \eta(x,y))$ to produce a warped image

$$W(x,y) = I(\xi(x,y), \eta(x,y)).$$

The object of this algorithm is to determine ξ and η given sampled versions of W and I. The general case is very difficult to solve—indeed, at the present time the only successful known method is to rely on a human being directing some interactive process.

In the interactive approach the user selects pairs of points in I and W that he believes to correspond. These *tiepoints* then serve as an input to some polynomial interpolator to produce the warp function ξ, η (see the companion report on advanced warp algorithms).

Automatic techniques have attempted to replace the human by trying to detect of some set of interesting "features" in both images which then serve as tiepoints. Once these tiepoints are chosen, the automatic algorithms then proceed as in the interactive approach.

In general, the automatic methods have not enjoyed a wide success. Their principal difficulties are a lack of robustness and a extreme logical complexity. The automatic selection of features in real-world images is an extemely difficult task—one which rivals and probably exceeds the difficulty of the image warping problem at hand. Real-world images often do not conform to the assumptions upon which the automatic algorithms rely. Thus, say, when an edge based algorithm will attempt to extract edges form a natural image, the output is very

unreliable. Instead of a clean set of edges in both images, one may have missing edges or a profusion of spurious edges. Missing edges damage the robustness of an edge algorithm, while large numbers of spurious edges will exceed the capacity of the machine to cope with so many edges. This logical complexity arises not from the natural images themselves but rather from an error in the assumptions about the structure of such images.

Past methods are misguided in two ways. 1) The attempt to mimic the action of a human being in the interactive case has proceeded from some fundamental misconceptions as to the nature of the human visual system. 2) The algorithms which purport to discover tiepoints suffer from an overwhelming logical complexity in the attempt to overcome poor assumptions about the images.

For the first point, the whole procedure of selecting tiepoints is less than ideal. The tiepoint procedure rests on the assumption that the human visual system operates primarily in the cognitive mode—picking features which have recognizable meaning in both images and then finding a correspondence between them. The principal functions of the visual system do not occur at a semantic or pattern recognition level but rather are hardwired into the visual cortex as highly parallel signal processing type operations. Rather than chasing pointers in lists, the visual system performs massively parallel arithmetic computations. Of the known such operations, one is of direct interest to the image warping problem. This is the operation of cyclopean perception.

In cyclopean perception the two images from the right and left eyes are fused into a single apperception. Additionally, the disparities in each image serve as depth cue information for binocular stereoscopic depth perception. One of the most important discoveries about the cyclopean perception system is that images are fused *without* any feature detection at all. The evidence for this discovery is the fact that we can fuse the random dot stereograms of Bela Julesz with without a cognitive scrutiny of the images. A typical random dot stereogram looks like this:

Left Right

If you are skilled, you may fuse these two images without the aid of any optics by simply staring at them. Each eye receives an image which is totally random. Thus the individual images have no semantic features or patterns (or edges) upon which an ostensible cognitive component of the visual system can fix. Yet the two images are successfully correlated by our cyclopean perception system.

There are several models of how this process is effected by the system. They differ in detail, but share the common operation of automatically computing the warp of one image into another. These algorithms have a collective computation flavor and are naturally implemented as a massively parallel process.

Section 2. A generalized correlation operation

If two images are identical to one another save a translation, then the shift parameter may be effectively estimated by the classical correlation operation.

Let $I(\mathbf{x})$ be the original image, $W(\mathbf{x}) = I(\mathbf{x} + \alpha)$ be the translated image. Then (barring periodicities in $I(\mathbf{x})$) we can estimate α by

$$\hat{\alpha} = \alpha^*$$

where

$$r_{IW}(\alpha^*) = \sup_{\alpha \in \Re} r_{IW}(\alpha).$$

That is, α^* is simply the the peak of r_{IW}, the cross correlation of I and W.

$$\begin{aligned} r_{IW}(\alpha) &= \langle I(\mathbf{x}), W(\mathbf{x} + \alpha) \rangle \\ &= \int_{\Re^2} I(\mathbf{x}) W(\mathbf{x} + \alpha)\, d\mathbf{x} \\ &= \int_{\Re^2} I(\mathbf{x} + \alpha) W(\mathbf{x})\, d\mathbf{x} \\ &= \langle W(\mathbf{x}), I(\mathbf{x} + \alpha) \rangle \end{aligned}$$

Unfortunately, cross correlation is 1) a global operation and 2)limited to shifts. If only a part of W matches I then its contribution to the correlation may well be overwhelmed by the mismatched part when the mismatch is summed in for the total correlation value. The second well-known limitation is that cross correlation is sensitive to orientation and scale mismatches. If a warped image has been rotated or dilatated then correlation can't decide which shift to put out.

These two limitations may be mitigated by relatively simple procedures. To overcome the first difficulty let us *window* the warped image to a small patch and search via the correlation operation for the optimum shift of the small patch. That is, let $h(\mathbf{x})$ be some compact support smooth window function and calculate

$$r'_{IW}(\alpha, \beta) = \frac{\langle I(\mathbf{x} + \alpha), h(\mathbf{x} + \beta) W(\mathbf{x}) \rangle}{\| h(\mathbf{x} + \beta) W(\mathbf{x}) \|}$$

Now the correlation function is a function of two displacements α is the displacement needed to match the windowed signal $h(\mathbf{x} + \beta)W(\mathbf{x})$ against the reference image I; and β is the displacement of the window

h within the warped image W, viz. β determines where the window is centered on the warped image. The normalizing factor is now necessary because the energy in each window may vary as the window moves about. This variation must be compensated for when searching for peaks in r' as α varies.

To overcome the second difficulty is equally simple: what we must do is to include the rotation and dilatation of the warped image. Let $A(\theta, \delta)$ be the 2×2 matrix which accomplishes this. Then we have

$$r''(\alpha, \theta, \delta) = \langle I(\mathbf{x} + \alpha), W(A(\theta, \delta)\mathbf{x}) \rangle$$

In order to find the optimum affine warp (a warp which is just a combination of translations, rotations and dilatations) we need only to pick the peak of r''.

It's obvious how to combine both these techniques—the window and the affine warp—to be able to handle any sufficiently smooth *general warp*. This is because, locally, any warp can be described by an affine one. The translation factor can be evaluated pointwise and the $A(\theta, \delta)$ matrix may be calculated from the Jacobian of the warp function

$$J = \frac{\partial(\xi, \eta)}{\partial(x, y)}$$

and vice versa.

Section 3. Computational considerations

The task of using a function

$$r'''(\alpha, \beta, \theta, \delta) = \frac{\langle I(\mathbf{x} + \alpha), h(\mathbf{x} + \beta) W(A(\theta, \delta)\mathbf{x}) \rangle}{\| h(\mathbf{x} + \beta) W(A(\theta, \delta)\mathbf{x}) \|}$$

in a computer algorithm seems almost hopeless since a function of eight real variables requires so much storage as to be all but useless.

But the ranges of these parameters do not span an entire image (of say 512 by 512 pixels) but rather their values range over narrow slabs of the parameter space.

Including θ and δ in the computation at present may be too much to hope for at present. We may omit these parameters if we use small windows for correlation because the sensitivity of peak to sidelobe ratio for correlation is reduced for small images.

Using present day technology it is realistic to compute $r'(\alpha, \beta)$, a function of only four variables, on todays supercomputers.

Recall that

$$r'_{IW}(\alpha, \beta) = \frac{\langle I(\mathbf{x}+\alpha), h(\mathbf{x}+\beta)W(\mathbf{x})\rangle}{\|h(\mathbf{x}+\beta)W(\mathbf{x})\|}$$

The denominator operation can be reexpressed as

$$\begin{aligned}\|h(\mathbf{x}+\beta)W(\mathbf{x})\| &= \sqrt{\langle h(\mathbf{x}+\beta)W(\mathbf{x}), h(\mathbf{x}+\beta)W(\mathbf{x})\rangle} \\ &= \sqrt{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h^2(\mathbf{x}+\beta)W^2(\mathbf{x})\, d\mathbf{x}} \\ &= \sqrt{\langle h^2(\mathbf{x}+\beta), W^2(\mathbf{x})\rangle}\end{aligned}$$

This last expression is simply a convolution operation, readily computable in time $O(n \log n)$ via the FFT. The numerator operation is also $O(n \log n)$. To see this take the Fourier transform of the numerator in the variables α, β.

$$\begin{aligned}\mathbf{F}r'_{IW}(\alpha,\beta) &= \mathbf{F}\langle I(\mathbf{x}+\alpha), h(\mathbf{x}+\beta)W(\mathbf{x})\rangle \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\langle I(\mathbf{x}+\alpha), h(\mathbf{x}+\beta)W(\mathbf{x})\rangle \\ &\quad e^{-i\omega_1\cdot\alpha}e^{-i\omega_2\cdot\beta}\,d\alpha\,d\beta \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left\{\int I(\mathbf{x}+\alpha)h(\mathbf{x}+\beta)W(\mathbf{x})\,d\mathbf{x}\right\} \\ &\quad e^{-i\omega_1\cdot\alpha}e^{-i\omega_2\cdot\beta}\,d\alpha\,d\beta\end{aligned}$$

Switching the order of integration

$$\begin{aligned}\mathbf{F}r'_{IW}(\alpha,\beta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(\mathbf{x}+\alpha)h(\mathbf{x}+\beta)W(\mathbf{x}) \\ &\quad e^{-i\omega_1\cdot\alpha}e^{-i\omega_2\cdot\beta}\,d\alpha\,d\beta\,d\mathbf{x} \\ &= \int W(\mathbf{x})\left\{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(\mathbf{x}+\alpha)e^{-i\omega_1\cdot\alpha}\,d\alpha\right\} \\ &\quad \left\{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\mathbf{x}+\beta)e^{-i\omega_2\cdot\beta}\,d\beta\right\}\,d\mathbf{x}\end{aligned}$$

The terms in braces are themselves Fourier transforms of I and h so that

$$\mathbf{F}r'_{IW}(\alpha,\beta) = \int W(\mathbf{x})\hat{I}(\omega_1)e^{i\omega_1\cdot\mathbf{x}}\hat{h}(\omega_2)e^{i\omega_2\cdot\mathbf{x}}\,d\mathbf{x}$$

Now using the shift property of the fourier transform we get that

$$\mathbf{F}r'_{IW}(\alpha,\beta) = \hat{I}(\omega_1)\hat{h}(\omega_2)\hat{W}(\omega_1+\omega_2)$$

Thus the calculation of the numerator is again a matter of taking fourier transforms, multiplying, and taking inverse fourier transforms. This is an $nm\log nm$ operation when m is the number of "frequencies" we wish to calculate for the shift parameters α,β.

Section 4. The inhibition operation

In general, $r'_{IW}(\alpha, \beta)$ is not a well behaved function with a single set of peaks. There are several approaches to improving this situation. First one may prefilter the reference and test images for optimum peak to sidelobe ratio. This produces very satisfactory correlation behavior. An approximation, known as phase correlation, correlates the two images by prewhitening them. This approach is widely reported in the literature so we will not discuss it further.

A second complementary step is to use the global information available about warp functions to improve the estimates of which warps are necessary. One bit of behavior is that the warp is to be continuous, thus if we compute the warp at one point, the neighboring points should have similar displacements. The second global behavior is that a global warp is univalent. That is, any estimate of the warp should have only one specific value. Competing peaks in the autocorrelation function should be suppressed.

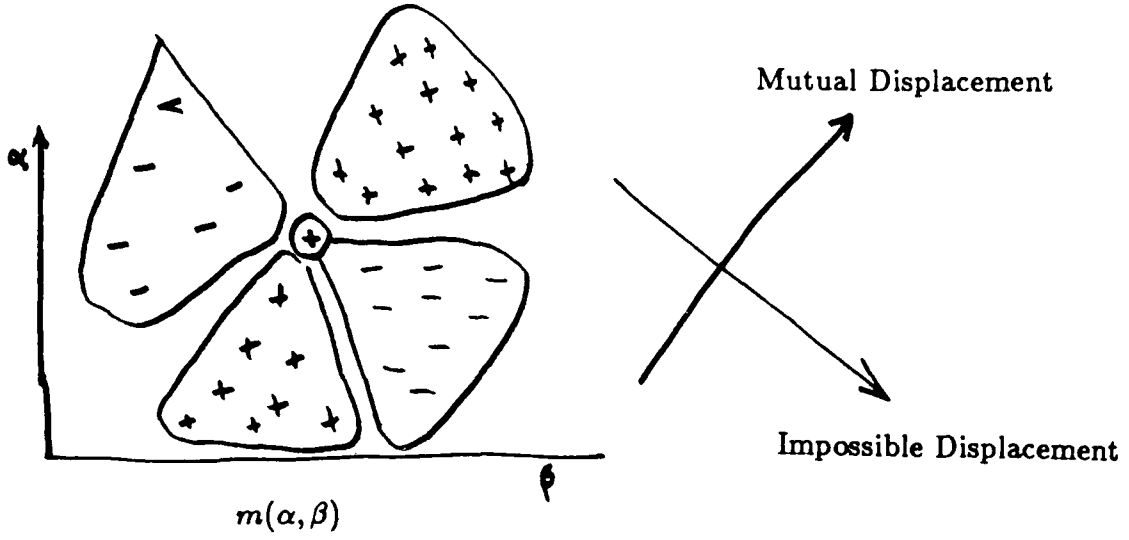
Marr and Poggio (1979) presented an algorithm which effectively processes correlation functions so that they become smooth and do not have ambiguous peaks. If we make a plot of $r'_{IW}(\alpha, \beta)$ we see that this can be accomplished by enhancing displacements which indicate continuity of warps and inhibiting displacements which indicate conflicting values for warps.

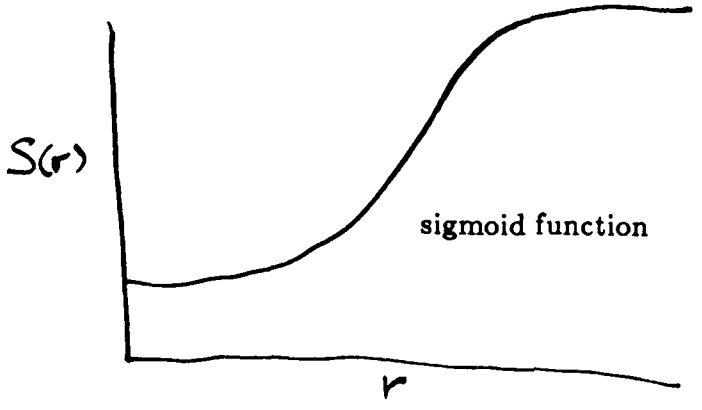
Performing this kind of inhibition and enhancement is a classical technique for the homomorphic lateral inhibition algorithm. We do it in the following way:

1. First convolve with a point spread function $m(\alpha, \beta)$ which has the general properites outlined in the figure above.

2. Pass the signal though a zero memory nonlineary sigmoid function

$$S(r) = \frac{1}{\sqrt{\pi\sigma}} \int_{-\infty}^{r} e^{\frac{x^2}{\sigma^2}} \, dx$$





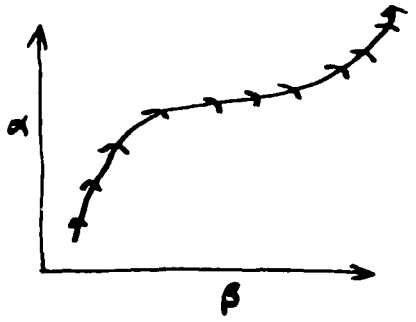
3. Repeat this cycle with a relaxation factor ϵ. That is if steps 2 and 3 are denoted

$$\Phi(r) = S[r(\alpha, \beta) \star m(\alpha, \beta)]$$

then we relax on Φ with parameter ϵ.

$$
\begin{aligned}
r_0 &= r(\alpha, \beta) \\
r_1 &= r_0 - \epsilon\Phi(r_0) \\
&\vdots \\
r_n &= r_{n-1} - \epsilon\Phi(r_{n-1})
\end{aligned}
$$

This computes a fixed point of Φ which has the nice property that for each peak α_0, β_0 the peak has only one possible new neighboring peak, i.e. if $\bar{r} = \lim_{n\to\infty} r_n$ then $\bar{r}$ looks like



The fuzzy line indicates a region of high value for $\bar{r}$. Everywhere else $\bar{r}$ is zero. Now all we need do is to follow the ridge to find the curve $\alpha = g(\beta)$ which is the warp function we desire.

CHAPTER 4
Advanced Warping algorithms with flexible capabilities

Section 1. Introduction

The ability to warp two dimensional data is a central capability required for the registration of multiple images of a given scene. Thus if we are to be able to multiply the effectiveness of image data collected through differing modes of production (Multi-specral, IR, SAR) then it is crucial to be able to warp images.

Present practice can be described as follows: multiple tiepoints are matched by hand and a low order bivariate polynomial approximant is fitted to the tie point data via least squares. In the immediate future we can envision automatic registration refinement of the tiepoints via correlation information (see the companion report on automatic calculation of warp functions).

These procedures give not only positional information but also estimates of the *Jacobian* of the warp transformation. That is, if the warp transformation is given by two functions

$$\xi = \xi(x, y)$$
$$\eta = \eta(x, y)$$

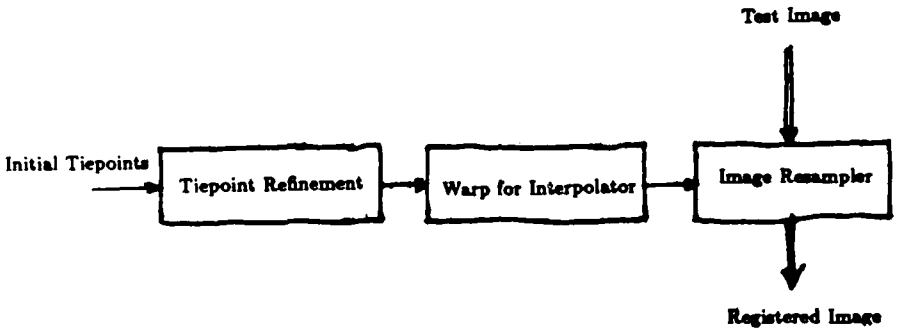
where $\{x, y\}$ are the spatial coordinates of the reference image and $\{\xi, \eta\}$ are the spatial coordinates of the test image, then the Jacobian of the warp transformation is written

$$J(x, y) = \frac{\partial(\xi, \eta)}{\partial(x, y)} = \begin{pmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \end{pmatrix}$$

Implicit within this matrix are the local scale, rotation, and shear

parameters at a given pair of points in test and reference image, that is, J represents the metric tensor g_{ij} of the warping transformation.

Advanced warping algorithms might be diagrammed as follows:



This report discusses new implementations of the warp function interpolator. The new implementations fulfill a number of desiderata. They are:

- The warp interpolator should be able to accept higher order information such as the Jacobian.
- The interpolator should have "local" capabilities.
- The interpolator should be able to accept *a priori* information about the warp geometry.

Before we examine these points let us discuss the present day warp interpolators.

Section 2. Present day warp approximators

The input to present day warp approximators is a set of *tiepoints*, a set of points $\{x_i, y_i : i = 1, 2, \ldots, N\}$ in the original image, hereafter known as the *reference image*, along with a corresponding set of points $\{\xi_i, \eta_i : i = 1, 2, \ldots, N\}$ in the image to be warped, hereafter known as the *test image*. The warp algorithm approximates these points by performing a least squares fit on the point data $\{x_i, y_i, \xi_i, \eta_i\}$. This procedure results in two low order bivariate polynomials

$$\xi(x, y) = \sum_{i,j} a_{ij} x^i y^j$$

$$\eta(x, y) = \sum_{i,j} b_{ij} x^i y^j.$$

Since the procedures for both $\xi(x, y)$ and $\eta(x, y)$ are identical, it suffices to just consider a polynomial approximant for a single surface

$$z(x, y) = \sum_{i,j} a_{ij} x^i y^j.$$

The present day warp approximators have a number of drawbacks.

- We really need an interpolant instead of an approximant. The tie-points have been chosen with high reliability and should be matched exactly.
- The method is global rather than local.
- No *a priori* information may be included.
- The method is unable to accept higher order information

If we try to use a Lagrange-type interpolation rather than a least square approximation there arise severe constraints on the number of points that a warp interpolator can accept. Too few points do not give a satisfactory fit. On the other hand, too many tiepoints requires a

high order polynomial interpolator bringing its well known attendant stability problems. Furthermore, under certain spatial configurations of the tiepoints, ill-conditioning of the solution procedure may crop up. For example, this will happen when many tiepoints are packed closely together with a few placed at a relatively large distance.

For these reasons, pure interpolation is not a feasible technique for general use. The new methods outlined in this report *do* interpolate, without the disadvantages outlined above.

The second disadvantage is that the current method is purely global. Thus local distortions can only be accommodated by increasing the order of the approximation. If there is a small area of the image whose warp is substantially different from the rest of the image—as in the side of a hill—then it will be difficult to match both the warps of the small area and of the rest of the image without going to a high degree approximant. This also leads to ill-conditioning.

Additionally, the current algorithms assume nothing about the *a priori* geometry of the warp. The basis functions of the current algorithms are the monomials $x^i y^j$ which are not matched to the possible forms that a warp may take, which are limited by the geometry of a spherical Earth imaged by an orbiting camera. Because of this, much more information is required in order to fit the natural tendencies of the basis functions to make them conform to the actual warp. An *a priori* model of the possible warps cuts down the number of possible warps down to physical reality. For example, NASA has used such information with great success. They have found that an *a priori* warp model along with knowledge of the spacecraft attitude combined with a simple autocorrelation refinement technique allows one to register images with hand entry of as little as three tiepoints. Unfortunately, their images have relatively low resolution and large field of view. It is probable that the wide area workstation under consideration for this project must deal with high ground resolution images which make the above results less relevant.

Finally, the standard algorithms are unable to accept data which furnishes information higher than zeroeth order. This makes the warp

interpolators unable to accept the output of more advanced warp estimators. For example, in the companion report we show an advanced algorithm which is capable of putting out only the correspondence between the shift of two points, but also the rotation and scale of the local patches.

Section 3. A new warp interpolator

The technique which we discuss uses algorithms for bivariate interpolation on randomly spaced points (Maude 1974, Vittitow 1978). The problem we are concerned with is as follows:

Given a set of triples $\{\langle x_i, y_i, z_i\rangle : i = 1, 2, \cdots, N\}$ find a smooth surface $F(x, y)$ such that

$$z_i = F(x_i, y_i), \qquad i = 1, 2, \cdots, N.$$

In our case, the tiepoints in the reference image are given by $\langle x_i, y_i\rangle$ and the tiepoints in the test images are given by $\langle \xi_i, \eta_i\rangle$. We interpolate the ξ and η warp functions separately: choose either set as the z_i in the above statement.

The method we describe here solves the first two of the four disadvantages inherent within the current algorithm. The last item also has a ready solution by this technique; but we won't include it here since the development is easy to infer from the present discussion.

The third point—no *a priori* information—is not directly addressed by this technique. We could attempt to use the new algorithm to work off a baseline warp which is generated by known imaging geometries. This would effectively incorporate such *a priori* knowledge.

Let us now describe Maude's method for bivariate interpolation. Given a set of triples $\{\langle x_i, y_i, z_i\rangle : i = 1, 2, \cdots, N\}$

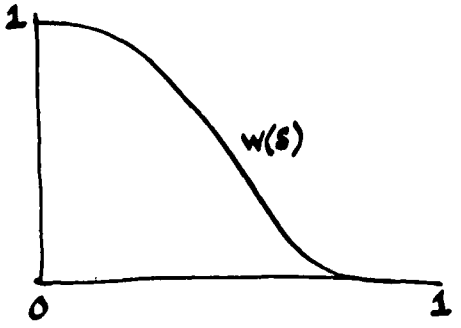
Step 0. Set $i = 0$. Choose a weighting function $w(s)$ such that

(i) $w(x) \in C^n(\Re^+)$

(ii) $w(x) > 0$ if $0 \leq s < 1$ and $w(s) = 0$ for $s > 1$

(iii) $\frac{d^i w}{ds^i}\big|_{s=0} = \frac{d^i w}{ds^i}\big|_{s=1} = 0$ for $i = 1, 2, \cdots, N$.

An example of such a function would be a hermitian polynomial



Step 1. For $\langle x_i, y_i \rangle$ find the five nearest tiepoints in the set.

Step 2. Define a disc with radius R_i so that the disc encloses the five nearest points but no others.

Step 3. Find the unique quadratice bivariate polynomial $Q_i(x, y)$ interpolating the six points in the disc. If this is the last point go to step 4, otherwise increment i and go to step 1.

Step 4. Over the union of the discs define

$$F(x, y) = \frac{\sum_{i=1}^{N} [w(d_i(x, y)/R_i) Q_i(x, y)]}{\sum_{i=1}^{N} w(d_i(x, y)/R_i)}$$

where $d_i(x, y)$ is the distance from $\langle x_i, y_i \rangle$ to $\langle x, y \rangle$.

It is easy to see that F is smooth and interpolates the data on the union of the discs. Note that if $d_i > R_i$ then $w(\frac{d_i}{R_i}) = 0$. Thus

for any given point, only the interpolating polynomials within a disc contribute to $F(x, y)$.

If a tiepoint were to be perturbed slightly, its effect would not be global. Indeed, only in a disc determined by the five closest tiepoints would be the interpolated function change. For larger degrees of freedom one could choose more points to determine the local disc.

For the weighting functions one can use an alternative method. Suppose the ith disc with center $\langle x_i, y_i \rangle$ has radius R_i. Define

$$U_i(x, y) = \max(0, R_i - (x - x_i)^2 - (y - y_i)^2).$$

Note that this function is positive inside the disk and zero on the boundary. Taking the nth power of u_i we obtain a function with an nth order zero at the boundary. So we can define

$$w_i(x, y) = \begin{cases} u_i^{n+1}(x, y), & \text{if } d_i(x, y) < R_i \\ 0, & \text{otherwise} \end{cases}$$

Maudes method has a number of advantages and disadvantages. The advantages are:

- Continuity to any order is easy to obtain
- Computation is local so that the complexity is linear in the number of points.
- Precision is quadratic or higher.
- The method generalizes to other basis functions.

Its disadvantages are:

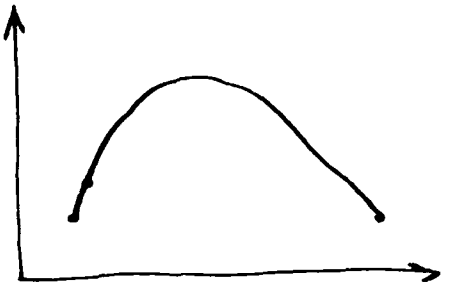
- Ill-conditioning in the computation of the quadratic polynomial can occur.
- The overlapping discs may have "holes" in the interpolation domain. The fact that higher order continuity is easy to obtain is especially

serendipitous considering that possibility that in the future interpolation with higher order information, e.g. the metric tensor, is desired. Because Maude's method is computationally local we could conceivably construct fast special purpose integrated circuits which could perform the computation in parallel. More immediately, a local computation assures us that the computation time is only linear with the number of tiepoints as opposed to say n^2 time. Quadratic precision is of little interest to us. The fact that the method generalizes to other basis functions—Maude's method works for any C^n interpolatory scheme, not just bivariate quadratics—prompts the observation that Maude's method really is a generalized localization scheme which takes any global interpolation method and localizes it. The reason that this observation is interesting is that the geometry of an imaging system is *not* quadratic but rather some function involving trigonometric rational polynomials arising from the curvature of the earth as well as spacecraft position and attitude. Thus, if we were to use more suitable interpolating basis functions, Maude's method could still be applied.

The disadvantages of Maude's method can be corrected. We will discuss the solution to the first shortcoming below. The second disadvantage, holes, is easily solved in an interactive environment. The user simply defines a tiepoint in the hole.

Fixing up the ill-conditioning of the Maude scheme is unfortunately not quite so easy. The difficulty where ill-conditioning occurs is illustrated by this univariate example.

Even though the z variation in the three points is small, there is a huge variation in the quadratic interpolator. It can easily be seen that this behavior can be made to be as terrible as we like simply by moving the first two points close to one another. A similar effect—although more complicated—obtains in the bivariate case. Basically this ill-conditioning arises when there are some tiepoints in a disc bunched up while other points are far off. The situation is apt to occur quite often in image warping. Therefore, it is imperative that we search for methods to mitigate ill-conditioning.



univariate example

Vittitow (1978) has investigated this question thoroughly. He gives three different methods for solving the ill-conditioning problem. We discuss the three methods here. It appears that methods II and III are most suitable for the image warping problem.

The three methods are all straightforward modifications of Maude's basic method.

Method I. *(a)* Proceed as in the Maude scheme except choose the discs for (and interpolate to) fewer than six points, say n. This creates an underdetermined system which is solved by

(b) Fitting a least squares bivariate quadratic $Q(x, y)$ with respect to the cost functional

$$J(Q) = \sum a_{ij}^2$$

where a_{ij} are the coefficients of $Q(x, y)$.

This method minimizes the size of the coefficients, in particular, the size of the higher degree monomials which are primarily responsible for overshoot.

Method II. *(a)* Same as in Method I. *(b)* fitting the least squares quadratic to the K next nearest neighbors where the total number of points $n + K > 6$.

This method looks at the behavior of the interpolating quadratic outside the domain of interest. If there is a significant amount of overshoot *inside* the disc, then this method capitalizes upon the fact that the quadratic will behave badly *outside* the disc.

Method III. *(a)* Approximate the surface by any given global scheme.

(b) Correct the approximation surface by a Maude surface that interpolates the error.

Of course, all three methods retain the flexibility and generalizability of the basic Maude scheme so that, for example, we will still be able to incorporate higher order information, or shift to a different set of basis functions.

References

Julesz,B. *Foundations of Cyclopean Perception*. Univ. Chicago Pr. 1971.

Marr,D. and Poggio, T., "A theory of human stereo vision", *Proc. Roy. Soc. London B*,**204**, 1979, pp.301-328.

Maude (1974) "Interpolation—Mainly for graphics plotters", *The Computer Journal*, v.16, 1974, pp.64-65.

S.L Tanimono, and T. Pavlidis(1975) "A hierarchical data structure for picture processing", *Computer Graphics and Image Processing*, v.4 no.2, June 1975.

Vittitow, W (1978) *Interpolation to arbitrarily spaced data*, Ph.D. Diss., Dept. Math. University of Utah.

L. Williams "Pyramidal Parametrics" *Computer Graphics*, v.17 no.3, July 1983, pp.1-11.

☆ U.S. Government Printing Office: 1988 513-117/64103

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control, Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C^3I systems. The areas of technical competence include communications, command and control, battle management, information processing, surveillance sensors, intelligence data collection and handling, solid state sciences, electromagnetics, and propagation, and electronic, maintainability, and compatibility.

END

DATE

FILMED

6-1988

DTIC