Hybrid Image & Signal Processing II

83

H992

PROCEEDINGS
SPIE—The International Society for Optical Engineering

Hybrid Image and Signal Processing II

David P. Casasent
Andrew G. Tescher
Chairs/Editors

18-20 April 1990
Orlando, Florida

Sponsored by
SPIE—The International Society for Optical Engineering
Cooperating Organization
CREOL/University of Central Florida

Published by
SPIE--The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA

Volume 1297

9250128

SPIE (The Society of Photo-Optical Instrumentation Engineers) is a nonprofit society dedicated to advancing engineering and scientific applications of optical, electro-optical, and optoelectronic instrumentation, systems, and technology.

9250121

DS36/21

The papers appearing in this book comprise the proceedings of the meeting mentioned on the cover and title page. They reflect the authors' opinions and are published as presented and without change, in the interests of timely dissemination. Their inclusion in this publication does not necessarily constitute endorsement by the editors or by SPIE.

Please use the following format to cite material from this book:
Author(s), "Title of Paper," *Hybrid Image and Signal Processing II,* David P. Casasent, Andrew G. Tescher, Editors, Proc. SPIE 1297, page numbers (1990).

Library of Congress Catalog Card No. 90-52789
ISBN 0-8194-0348-2

Published by
SPIE—The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA
Telephone 206/676-3290 (Pacific Time) • Fax 206/647-1445

Copyright © 1990, The Society of Photo-Optical Instrumentation Engineers.

Copying of material in this book for sale or for internal or personal use beyond the fair use provisions granted by the U.S. Copyright Law is subject to payment of copying fees. The Transactional Reporting Service base fee for this volume is $2.00 per article and should be paid directly to Copyright Clearance Center, 27 Congress Street, Salem, MA 01970. For those organizations that have been granted a photocopy license by CCC, a separate system of payment has been arranged. The fee code for users of the Transactional Reporting Service is 0-8194-0348-2/90/$2.00.

Individual readers of this book and nonprofit libraries acting for them are permitted to make fair use of the material in it, such as to copy an article for teaching or research, without payment of a fee. Republication or systematic or multiple reproduction of any material in this book (including abstracts) is prohibited except with the permission of SPIE and one of the authors.

Permission is granted to quote excerpts from articles in this book in other scientific or technical works with acknowledgment of the source, including the author's name, the title of the book, SPIE volume number, page number(s), and year. Reproduction of figures and tables is likewise permitted in other articles and books provided that the same acknowledgment of the source is printed with them, permission of one of the original authors is obtained, and notification is given to SPIE.

In the case of authors who are employees of the United States government, its contractors or grantees, SPIE recognizes the right of the United States government to retain a nonexclusive, royalty-free license to use the author's copyrighted article for United States government purposes.

Printed in the United States of America

HYBRID IMAGE AND SIGNAL PROCESSING II

Volume 1297

CONFERENCE COMMITTEE

Chairs
David P. Casasent, Carnegie Mellon University
Andrew G. Tescher, Lockheed Palo Alto Research Laboratory

Program Committee
George Eichmann, City College/CUNY
Ivan Kadar, Grumman Aerospace Corporation
Edward R. Washwell, Lockheed Missiles & Space Company, Inc.

Session Chairs
Session 1—Multisensor Recognition and Tracking
Ivan Kadar, Grumman Aerospace Corporation

Session 2—Optical Correlators and Pattern Recognition
David P. Casasent, Carnegie Mellon University

Session 3—Optical Processing Applications
Edward R. Washwell, Lockheed Missiles & Space Company, Inc.

Session 4—Phase-Only Filters
B. V. K. Vijaya Kumar, Carnegie Mellon University

Session 5—Digital Techniques
George Eichmann, City College/CUNY

Session 6—Neural Nets for Image Processing
Andrew G. Tescher, Lockheed Palo Alto Research Laboratory

Conference 1297, *Hybrid Image and Signal Processing II*, was part of a three-conference program on Image Processing and Artificial Intelligence held at the 1990 SPIE Technical Symposium on Optical Engineering and Photonics in Aerospace Sensing, 16–20 April 1990, in Orlando, Florida. The other conferences were:

Conference 1293, *Applications of Artificial Intelligence VIII*
Conference 1295, *Real-Time Image Processing II.*

Program Chair: **David P. Casasent,** Carnegie Mellon University

INTRODUCTION

This second conference in a series was again a most successful part of the program on Image Processing and Artificial Intelligence. The 36 papers collected in this proceedings cover a representative selection of digital, optical, and hybrid systems for a variety of applications.

The volume begins with a collection of papers on multisensor processing. This is followed by papers on various hardware optical correlators and new distortion-invariant filters for them. Several approaches to numerical and digital optical processing are described in the papers on optical processing. A unified set of papers on the very topical subject of phase-only filters provides current results and new approaches. Digital processing techniques are also presented. The final session presents work on advanced techniques and neural nets for image processing.

We sincerely thank all authors for their contributions and our program committee (George Eichmann, City College/CUNY; Ivan Kadar, Grumman Aerospace Corporation; and Ed Washwell, Lockheed Missiles & Space Company, Inc.) for their organizational help.

David P. Casasent
Carnegie Mellon University

Andrew G. Tescher
Lockheed Palo Alto Research Center

此为试读,需要完整PDF请访问: www.ertongbook.com

HYBRID IMAGE AND SIGNAL PROCESSING II

Volume 1297

CONTENTS

(continued)

HYBRID IMAGE AND SIGNAL PROCESSING II

Volume 1297

SESSION 1

Multisensor Recognition and Tracking

Chair
Ivan Kadar
Grumman Aerospace Corporation

9250121

Sensor data integration for real-time environmental analysis

Vibeke Libby
Lockheed Missiles & Space Company, Inc., Research & Development Division
3251 Hanover Street, Palo Alto, California 94304-1187

ABSTRACT

Efficient, high-speed algorithms are in great demand for applications in which the geometrical configuration of the environment must be assessed before a subsequent move can be performed. The knowledge of the spatial configuration of the object distribution is either a priori or obtained from fused and converted sensor data. The new method can: (1) readily implement known and new sensor data inputs, (2) process the resulting geometry in three-dimensional space for location and intersect, and (3) permit a system response with a best path in less than a second. Due to its simple architecture, the system can treat threats, targets, terrain, and moving objects in the same hardware.

1. INTRODUCTION

The method described in this paper permits quick assessment of relative locations of two-dimensional (2-D) and three-dimensional (3-D) rectangular objects with respect to an arbitrary number of points and to a real or fictive line of sight. To the degree that a given object environment conforms to such an approximation, the method has a wide range of applications from assessment of robotics movements to electronic warfare.[1] The algorithm is implemented in a small hardware unit called the TIGER (Three-dimensional Intersect & Geometrical Evaluation in Real-time) that can be incorporated in unmanned as well as manned systems in space, under water, or avionics.

In general, the hardware algorithm and accelerator not only provides instantaneous, relative information about objects with respect to different points in space but also evaluates which objects a given line or path intersects and how to avoid or home in on them. In many respects, the hardware functions as a "filter" that instantaneously identifies which paths are feasible (few intersects) as opposed to high risk (numerous intersects) (see Fig. 1). This approach significantly reduces the software tasks associated with geometrical intersects and, by dedicating more time to the overall path planning, will produce more accurate and reliable solutions.

The approach is a microcode, single-chip implementation of two algorithms, one of which can evaluate, sort, and identify 4 million spatial relations per second. If an additional intersect analysis is required, the second hardware algorithm is activated. It operates at an average rate of 1 million intersect evaluations per second.

Due to its high design flexibility, the hardware accelerator works with any 16-bit microprocessor and is totally independent of the system software language. All inputs and outputs from the unit consist of microcoded words that can be read and interpreted by the microprocessor on a 16-bit data bus. This makes the algorithm a suitable Ada subroutine.

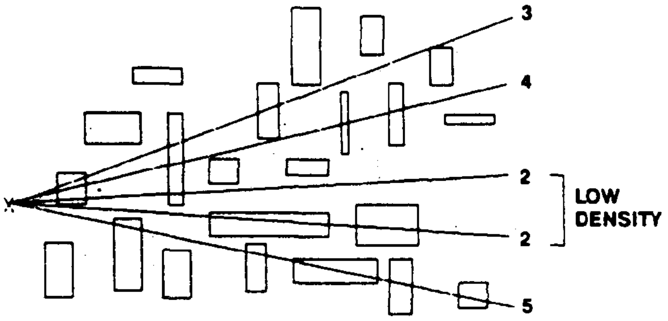

Fig. 1. Preliminary filter of object space shown in two dimensions for simplicity. In the illustrated case all objects carry the same weight. The region identified with low obstacle density will be used by software for further path optimization.

2. CAPABILITIES

The TIGER architecture and algorithm are particularly useful for rapid decision-making applications in which the environment is cluttered with a variety of known or detected objects (like terrain, targets, moving objects). In these cases, many different possible movements must be assessed within a critical time limit.

The movements might, for instance, originate from the same object like a robot's arm, on which several points must be tested for interference before a movement can be initiated. Or, the 3-D movement of a vehicle in a complex environment must be assessed. In such 3-D cases, most software programs are overwhelmed by the vast number of options and are unable to respond within the required time.

The major difference (as far as this implementation is concerned) between the movement of a robotic arm and a low-flying aircraft is the speed at which the environment changes in conjunction with the number of points to be analyzed in each case. These differences strongly affect the processing speed. For example, the TIGER memory loading time for N objects is given by:

$$\text{loading time} = (N \times 6 \times \text{clock-speed})\ \text{ns} \qquad (1)$$

$$\text{analysis time} = (\text{clock} - \text{speed})\ \text{ns}$$

as opposed to data analysis time per point, which is only one clock cycle. The loading time in Eq. (1) is a factor of 1.8 improvement over state-of-the-art VHSIC devices.

In the case of a robotic arm, its movements are most often restricted to a small, well-defined environment. Due to the required simultaneous analysis of several points distributed along the arm, such computations can be performed at a much higher speed than quoted above because of the slow rate at which the environment must be updated. As a consequence, this processing scheme will allow a highly refined and precise sequence of arm movements. As loading time is far more "costly" than analysis time, this improvement significantly affects the overall system performance.

3. ARCHITECTURE

The heart of the two algorithms lies in an associative comparative memory device that can perform comparisons between 16 bits of data limits. For instance, one can use the limits 5 and 10 to define a range (namely, the numbers between or equal to 5 and 10) of values. Using binary notation in the above example, a data point of value 2 will cause the associative comparator (AC) to respond with a 0 whereas a data point of value 8 will produce a 1. Originally, the first practical hardware implementation was designed and built in very large-scale integration (VLSI) by IBM.[2] Later in 1983 and 1984, TRW produced the much faster Window Addressable Memory (WAM) chip that complies with all very high speed integrated circuit (VHSIC) requirements.

The TIGER architecture utilizes three major building blocks (see Fig. 2):

1. Comparative memory bank
2. Decoder and sorter
3. Ambiguity resolver

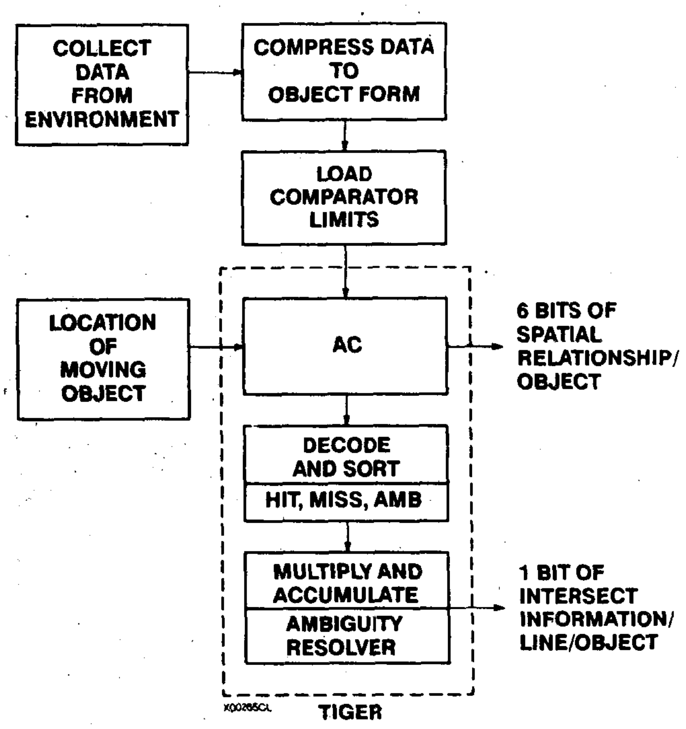
COLLECT DATA FROM ENVIRONMENT
COMPRESS DATA TO OBJECT FORM
LOAD COMPARATOR LIMITS
LOCATION OF MOVING OBJECT
AC
6 BITS OF SPATIAL RELATIONSHIP/ OBJECT
DECODE AND SORT
HIT, MISS, AMB
MULTIPLY AND ACCUMULATE
AMBIGUITY RESOLVER
1 BIT OF INTERSECT INFORMATION/ LINE/OBJECT
X00265CL
TIGER

Fig. 2. Data flow using TIGER hardware.

3.1 The comparative memory bank or associative comparator (AC)

The inputs are sets of 48-bit data words. In the case of intersect analysis, the first set is called A data, the second set B data. The line AB is also referred to as the line of sight under evaluation. There are 16 bits allocated to each of the coordinates x, y, and z.

As the A data are processed first, but must be decoded in conjunction with the B data, which are processed one clock cycle later, the result from the A data processing must wait for the B data to be processed. This is done by latching 6 bits of output A data until the B data are ready. This results in a total of 12 bits per point per object to enter the decoder. Due to the many output lines, a VLSI implementation of the TIGER will be far more efficient than using available discrete parts.

3.2 Decoder and sorter

Following the relative comparator bank is a highly specialized decoder that sorts the 12 bits into 10 possible 3-D geometrical configurations. Without sorting, two points represent 729 (27 × 27) possible geometrical relations in three dimensions. The 10 possible decoder outputs are given in Table 1. The number P indicates the

此为试读，需要完整PDF请访问：www.ertongbook.com

probability that a given configuration will occur when all configurations are given equal statistical weight.

Table 1. Distribution of the 729 spatial possibilities and their relative occurrence.

Occurrence/Configuration	N	P (%)
1. The line intersects the object	59	8.09
2. The line does not intersect the object	386	52.9
3. The line might intersect the object	284	39.0
3a. 2-D ambiguity with vanishing x-coordinate	28	3.84
3b. 2-D ambiguity with vanishing y-coordinate	28	3.84
3c. 2-D ambiguity with vanishing z-coordinate	28	3.84
3d. 3-D ambiguity of the form vertex-vertex	8	1.10
3e. 3-D ambiguity of the form vertex-edge	48	6.58
3f. 3-D ambiguity of the form vertex-plane	80	11.0
3g. 3-D ambiguity of the form edge-edge	16	2.19
3h. 3-D ambiguity of the form edge-plane	48	6.58

The terminology of vertex-vertex, vertex-edge, vertex-plane, edge-edge, and edge-plane identifies a given spatial line-object relation with intersect characteristics that require special analysis and hardware to resolve.

Note that the nonambiguous solutions constitute 60% (cases 1 and 2) of all configurations in three dimensions.

The category for which a check must be performed in order to establish a possible intersect condition is called an ambiguity "AMB" (cases 3a to 3h in Table 1). When the line definitely intersects the region of interest (ROI), the configuration is called a "HIT" (case 1 in Table 1), as opposed to a "MISS" (case 2 in Table 1) which is characterized as a nonintersecting geometry.

3.3 How ambiguities occur

To evaluate the spatial relation between a point and one or several objects, the object coordinates are stored in specified memory locations. Using an architecture that treats 16 sets of coordinates in parallel and also decodes the output in parallel, the location of the point in question with respect to the 16 objects is calculated at a rate of 4 million per second.

Note that no ambiguities arise from the analysis of a single point's location with respect to the object. Ambiguities first occur when two points are connected by a straight line. If the information regarding the relative location of the two end points was not available, it would be necessary to check the line with respect to all vertices by a slope comparison. By utilizing the algorithm described here, the problem is reduced to checking no or at most two vertices, and only for the ambiguous cases. This constitutes a 60% reduction of the computational effort.

3.3.1 2-D ambiguities

In a multiple-object environment, a given point will have a different relative location with respect to different objects. Therefore, a given point will, for instance, be in one spatial section with respect to one object and in another section with respect to a second object. In turn, this type of information holds the key to how to maneuver around the objects presented.

Having found the relative location of one data point, one can find the location of another data point, thereby defining the location of the two endpoints of a straight line. For some applications, it is of value to assess whether this line intersects the ROI (see Fig. 3). For instance, If the ROI is a region of target or danger, this kind of information is of particular interest.

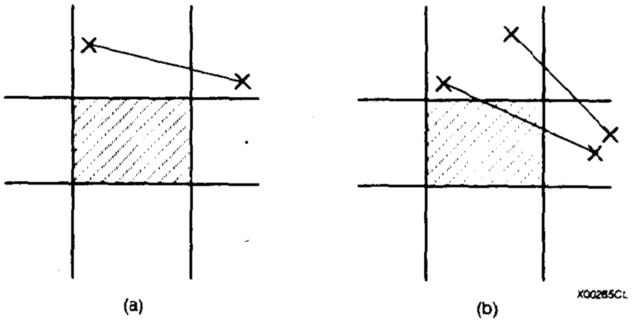

Fig. 3 Occurrence of two-dimensional ambiguity. Figure 3a shows how a line connecting two points in the indicated sections never can intersect the center rectangle. On the other hand, Fig. 3b illustrates a geometrical configuration in which both an intersect and a nonintersect can occur. Such configurations are called "ambiguous."

Table 2 lists all HIT, MISS, and AMB configurations for two points in two dimensions.

Table 2. All HIT, MISS and AMB configurations for two points in two dimensions.

Point 1 Location	Point 2 Location	Planar Configuration		
1	1	Miss		
1	2	Miss		
1	3	Miss		
1	4	Miss		
1	5		Hit	
1	6			Amb
1	7	Miss		
1	8			Amb
1	9			Amb
2	1	Miss		
2	2	Miss		
2	3	Miss		
2	4			Amb
2	5		Hit	
2	6			Amb
2	7			Amb
2	8		Hit	
2	9			Amb
3	1	Miss		
3	2	Miss		
3	3	Miss		
3	4			Amb
3	5		Hit	
3	6	Miss		
3	7			Amb
3	8			Amb
3	9	Miss		
4	1	Miss		
4	2			Amb
4	3			Amb
4	4	Miss		
4	5		Hit	
4	6		Hit	
4	7	Miss		
4	8			Amb
4	9			Amb
5	1		Hit	
5	2		Hit	
5	3		Hit	
5	4		Hit	
5	5		Hit	
5	6		Hit	
5	7		Hit	
5	8		Hit	
5	9		Hit	
6	1			Amb
6	2			Amb
6	3	Miss		
6	4		Hit	
6	5		Hit	
6	6	Miss		
6	7			Amb
6	8			Amb
6	9	Miss		
7	1	Miss		
7	2			Amb
7	3			Amb
7	4	Miss		
7	5		Hit	
7	6			Amb
7	7	Miss		
7	8	Miss		
7	9	Miss		
8	1			Amb
8	2		Hit	
8	3			Amb
8	4			Amb
8	5		Hit	
8	6			Amb
8	7	Miss		
8	8	Miss		
8	9	Miss		
9	1			Amb
9	2			Amb
9	3	Miss		
9	4			Amb
9	5		Hit	
9	6	Miss		
9	7	Miss		
9	8	Miss		
9	9	Miss		
Sum		32	21	28

In the figure no attempt has been made to distinguish between the different ways that ambiguities are resolved but only the difference in geometrical configuration.

3.3.2 3-D ambiguities

The hardware accelerator makes relative comparisons between a point and any section in space that is confined by a rectangular shape. This means that a box-shaped object (which need only be identified by two correctly chosen points) automatically divides the surrounding space into 27 regions of interest (see Fig. 4). Even when an adjacent object also divides space and thereby causes the spatial relations to overlap, no complexity is added to the analysis. Note that this is a result of the implementation of the algorithm because all evaluations of points with respect to objects are relative. Thus, the location of a point is assessed with respect to the objects individually independent of the location of the remaining objects. An example of a vertex-edge ambiguity configuration is given in Fig. 5.

3.3.3 Ambiguity resolver

To resolve the ambiguity configurations, the object coordinates which were originally loaded into the comparator bank must be used again for the more detailed ambiguity evaluation. These coordinates are stored in a random access memory (RAM) at the same time as the comparator bank and with the same data. Depending on the type of ambiguity, different pointers are generated to the appropriate RAM locations.

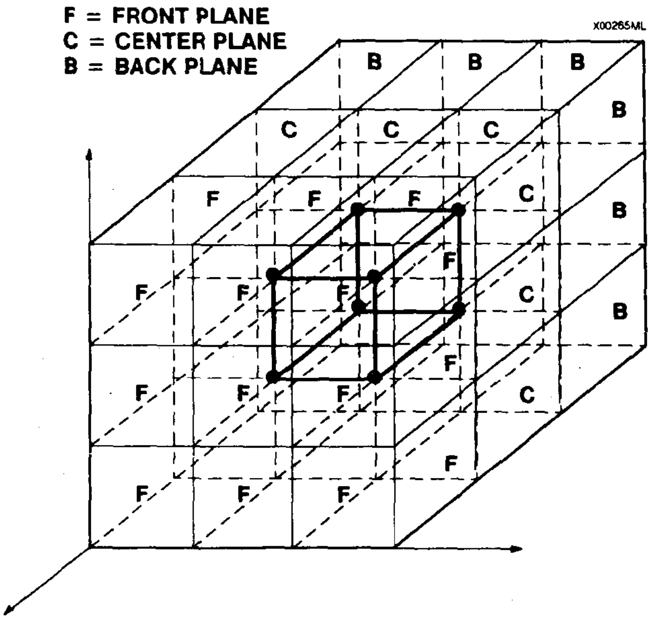

Fig. 4 Space in a single box-shaped object divided into 27 sections. For simplicity, three planes are named F, C, and B, and each contains nine subsections. The extent of all sections but the center is considered infinity but is, in practice, limited by the width of the data bus. The bold-faced box represents the ROI (= C5) in three dimensions.

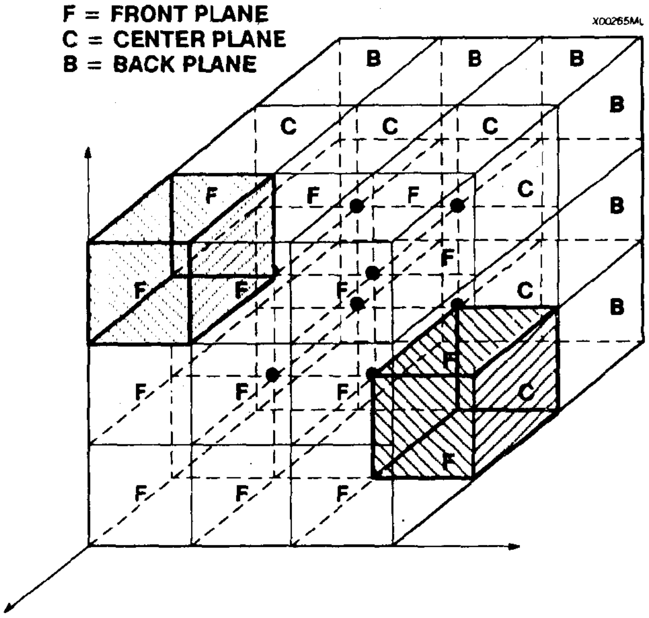

Fig. 5 Example of vertex-edge ambiguity spatial configuration of two points located in F1 and C9 with respect to the region of interest (C5).

The ambiguity evaluation is a slope comparison in two dimensions and a plane intersect analysis in three dimensions. Independent of the dimension, the computation only involves 16- × 16-bit multiplication and addition. Two 16-bit multipliers work in parallel at this stage. The output from the two MAC (multiply and accumulate) units are compared in a 16-bit comparator. The comparator output will determine whether an ambiguity is resolved as an intersect or a nonintersect.

The final results from all the objects are presented on an output bus in terms of zeroes (nonintersect) and ones (intersect).

A summary of the ambiguity distribution is given in Table 3.

The first three cases possess the same ambiguities found in two dimensions and does not require additional analysis here. However, all of the remaining cases contain spatial configurations which are ambiguous. A graphical representation of all possible geometric combinations of HIT, MISS, and AMB in two and three dimensions are given in Figs. 6 and 7.

Table 3. Description of the types of ambiguities.

The Three 2-D Types of Ambiguities
CASE 1: Each of the two points is in the YZ plane, characterized by X = 0.
CASE 2 : Each of the two points is in the XZ plane, characterized by Y = 0.
CASE 3 : Each of the two points is in the XY plane characterized by Z = 0.
The Five 3-D Types of Ambiguities
CASE 1: Each of the two points is located in a section that touches the obstacle corner (i.e., shares a point with the object). This state of ambiguity is called the vertex-vertex case.
CASE 2: Each of the two points is in a section that shares an edge with the object. This is the edge-edge ambiguity case.
CASE 3: One of the sections shares a point and the other an edge with the central box. This state of ambiguity is called the vertex-edge case.
CASE 4: One of the points of interest is in a corner section and the other in a section that shares a plane (one of the faces) with the box. This is called the vertex-plane ambiguous case.
CASE 5: The final case involves one point in an edge section and the other in a plane section. This is called the edge-plane ambiguous case.

4. A 2-D EXAMPLE

To apply the comparative capability described above to 3-D spatial analysis, we will illustrate the technique in two dimensions. This procedure will show that the method also can be applied with the same principle to three dimensions.

The novelty of the algorithm described here lies in the interpretation of the results (the output patterns of 0's and 1's) from the AC. Instead of comparing values of points as in the one-dimensional case, we will look at half-planes, confined by the minimum and maximum limits (see Fig. 8).

The ROI is always the central region, confined by values x-min, x-max, y-min, and y-max. In an AC, it can easily (in one clock cycle) be verified if a point lies within or outside the ROI. (Note that the ROI is a rectangle in two dimensions and a box in three dimensions.) In signal processing, the limits in the y dimension could, for instance, be frequency, whereas the power of the signal could represent the x dimension.

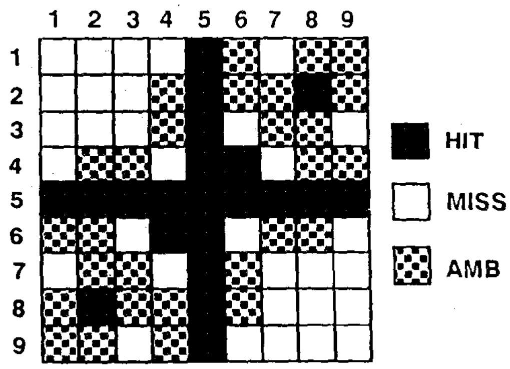

Fig. 6 Distribution of HIT, MISS, and AMB configurations in two dimensions for two points. Each point can be located in any one of nine segments in the plane.

When using the AC in two dimensions, we can use up to 24 bits for each x and y value. In three dimensions, 16 bits can be allocated for each x, y, and z value. The comparator performs a logical AND of the two results before the output is available. This means that both the x-value and the y-value of the data point must fall within the prestored limits. This results in an evaluation of whether or not the data point is within the central area/volume (ROI).

However, we note that if the data point is outside the limits, the AC will in this configuration not indicate in which of the outside sections the point is located. Here it is important to note that the two sets of limits actually divide the plane into nine regions. Below we will show how a different programming method and interpretation of the results can lead to a differentiation of not only the central region with respect to the rest of the plane, but actually tell in which of the remaining eight subsections the data point is located. In three dimensions, this capability becomes even more powerful, as space in this case is divided into 27 sections.

The above method must be modified as follows:

1. The x and y limits shall be treated separately so that the AND between the two comparisons is bypassed.
2. An additional set of confining planes must be introduced.

In fact, it is immaterial to the analysis which two set of planes is chosen so long as consistency is kept. Also, the modification holds for both two and three dimensions: four limits are needed for two dimensions and six limits for three dimensions. To keep the description close to the

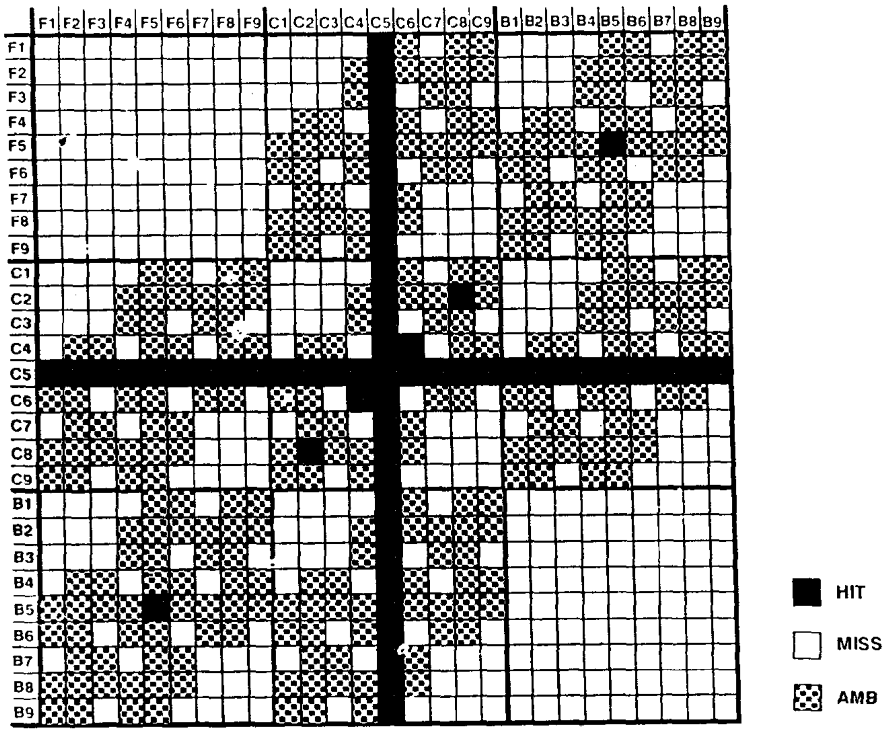
F1 F2 F3 F4 F5 F6 F7 F8 F9 C1 C2 C3 C4 C5 C6 C7 C8 C9 B1 B2 B3 B4 B5 B6 B7 B8 B9
F1
F2
F3
F4
F5
F6
F7
F8
F9
C1
C2
C3
C4
C5
C6
C7
C8
C9
B1
B2
B3
B4
B5
B6
B7
B8
B9
HIT
MISS
AMB

Fig. 7 Distribution of HIT, MISS, and AMB configurations in three dimensions for two points. Each point can be located in any of 27 segments in space.

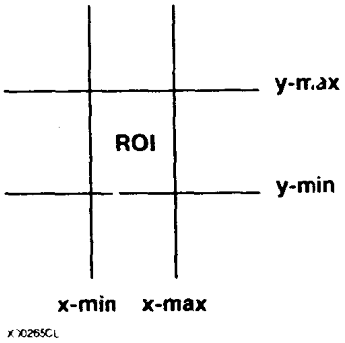

Fig. 8 Definition of region of interest (ROI) in two dimensions.

above example, we will chose the two sets of limits as follows:

x-min, x-max
y-min, y-max

x-max, infinity
y-max, infinity

In this case, infinity is the maximum number which can be evaluated in the AC, and thus is related to the number of bits available. In the 2-D case, this number is represented by the value of $2^{24}-1$.

Analyzing a data point consisting of 24 bits of x data and 24 bits of y data will produce a coded output consisting of 4 bits—1 bit from each comparison. For reference, the different regions are numbered as indicated in Fig. 9. This numbering system is maintained in three dimensions.

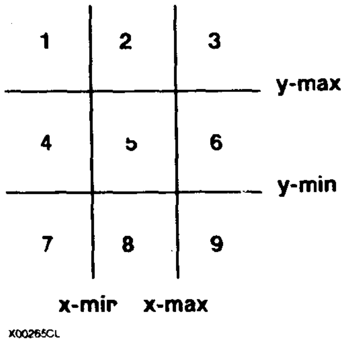

Fig. 9 Division of the plane defined by ROI (region 5) in two dimensions.

The information contained in each of the four output bits is as follows:

bit 1 = 0: x is in region 1, 3, 4, 6, 7, or 9
bit 1 = 1: x is in region 2, 5, or 8

bit 2 = 0: y is in region 1, 2, 3, 7, 8, or 9
bit 2 = 1: y is in region 4, 5, or 6

bit 3 = 0: x is in region 1, 2, 4, 5, 7, or 8
bit 3 = 1: x is in region 3, 6, or 9

bit 4 = 0: y is in region 4, 5, 6, 7, 8, or 9
bit 4 = 1: y is in region 1, 2 or 3

In one example, an AC output is 1100. Looking at bits 1 and 3 (which are the results from the x coordinates) isolates regions 2, 5 and 8, whereas bits 2 and 4 isolate regions 4, 5, and 6. As region 5 is the only region that occurs in both cases, the data point is located within this region. Notice that there are more possible binary output combinations (16) than there are regions (9). Thus, certain outputs like 1101 can never occur during proper operation and they can, therefore, be used as error detection.

Another output example can be 0110. This corresponds to finding the overlap between regions 3, 6, 9 and 4, 5, 6, and identifying the data point as being in region 6.

These examples demonstrate that a simple decoder can be built using standard logical gates.[3] Note that no actual calculations have been used to obtain these comparative results. It is also obvious that one can treat several regions or objects in parallel and that there is no reason why any of the regions cannot overlap. This is because each region is treated separately (the AND function is eliminated) and the results are comparative with respect to each region. Thus, a data point can be in region 7 with respect to one object and region 9 with respect to another (see Fig. 10). This clear division of the object space with respect to each point makes further analysis of possible movement and avoidance simple.

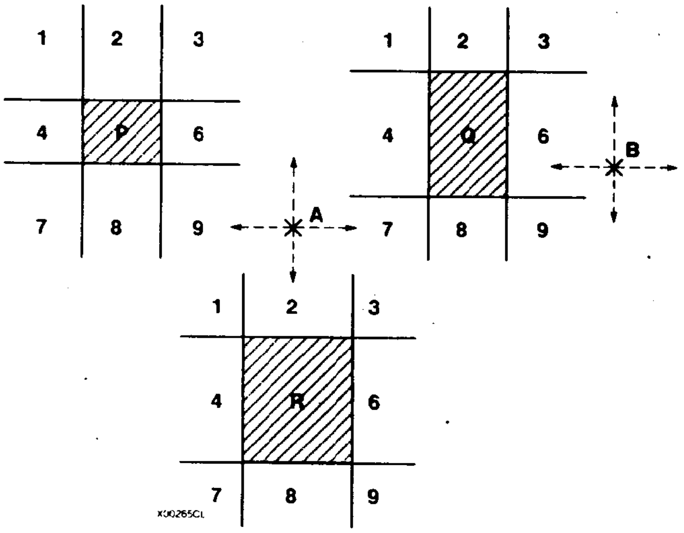

Fig. 10. Point A is in region 9 with respect to P, in region 7 with respect to Q, and in region 2 with respect to R. Point B is in region 6 with respect to P and Q, and in region 3 with respect to R. The configuration A-B is ambiguous around Q(6-7) and a MISS with respect to P(6-9) and R(2-3) (see Table 2).

5. PERFORMANCE

The TIGER architecture is optimized for performing real-time intersect analysis in three dimensions. As input, it requires that the Cartesian coordinates of the object's boundaries be analyzed. Given these boundaries, the TIGER determines whether a line characterized by two points (also in Cartesian coordinates) intersects two of the six surfaces defining any of the objects under consideration.

The TIGER architecture will significantly reduce the processing speed obtainable with current technology (500 ns/2-D ambiguity). Because the TIGER approach is optimized for geometrical analysis and intersect evaluation, three times as many objects can be analyzed with the same amount of hardware in the same amount of time as current technology. Due to TIGER's highly parallel and pipelined architecture, results will appear on the output bus at a rate of 40 to 120 ns apart. We believe that TIGER is the only hardware implementation of a 3-D geometric interceptor that can evaluate a minimum of 1 million intersects per second.

6. DATA REPRESENTATION

Consider as an example the case of planning and updating a military low-level penetration mission. The operation of the TIGER algorithm depends on a realistic underlying data representation of the associated mission data elements. The mission is based, among other things, upon location of known threats, terrain, and targets. This information will be available to the onboard path analyzer. If an unknown threat or a new target is detected and, therefore, deviation from the preplanned path becomes a possibility, the TIGER module will be activated and perform several levels of analysis. The result will be a path recommendation, different or identical to the preplanned path depending on the nature and location of the new data.

The information regarding an object occupying a specified volume in space can be made available in several ways. Three of these are:

1. A curve with a known or unknown parametric representation intersects a specified volume. An example of such knowledge is the elevation curves on a map. Here all the spatial information is present to generate a 2- and 3-D ROI image of the terrain.
2. The occupied volumes (ROIs) are identified by calculations based upon sensor knowledge of the source location. An example is the detection of a radar site and the subsequent calculation of its detection space, associated lethality, and corresponding range. This information is sufficient

此为试读，需要完整PDF请访问: www.ertongbook.com