PROCEEDINGS

Of SPIE-The International Society for Optical Engineering

Volume 848

Intelligent Robots and Computer Vision

Sixth in a Series

David P. Casasent, Ernest L. Hall
Chairs/Editors

Sponsored by
SPIE—The International Society for Optical Engineering

in association with
IEEE Industrial Electronics Society
Society of Instrument and Control Engineers of Japan

Cooperating Organizations
Center for Optical Data Processing/Carnegie Mellon University
Electro-Optics Technology Center/Tufts University
IEEE Robotics and Automation Council
Sira Ltd.—The Research Association for Instrumentation (UK)

2-6 November 1987
Cambridge, Massachusetts

Proceedings of SPIE—The International Society for Optical Engineering

Volume 848

Intelligent Robots and Computer Vision

Sixth in a Series

David P. Casasent, Ernest L. Hall
Chairs/Editors

Sponsored by
SPIE—The International Society for Optical Engineering
in association with
IEEE Industrial Electronics Society
Society of Instrument and Control Engineers of Japan
Cooperating Organizations
Center for Optical Data Processing/Carnegie Mellon University
Electro-Optics Technology Center/Tufts University
IEEE Robotics and Automation Council
Sira Ltd.—The Research Association for Instrumentation (UK)

2-6 November 1987
Cambridge, Massachusetts

Published by
SPIE—The International Society for Optical Engineering
P.O. Box 10, Bellingham, Washington 98227-0010 USA
Telephone 206/676-3290 (Pacific Time) • Telex 46-7053

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.

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," *Intelligent Robots and Computer Vision: Sixth in a Series,* David P. Casasent, Ernest L. Hall, Editors, Proc. SPIE 848, page numbers (1988).

Library of Congress Catalog Card No. 87-63194
ISBN 0-89252-883-4

Printed in the United States of America.

INTELLIGENT ROBOTS AND COMPUTER VISION
SIXTH IN A SERIES

Volume 848

INTRODUCTION

The Intelligent Robots and Computer Vision Conference was held November 2–6, 1987, in Cambridge, Massachusetts. This was the sixth conference in a continuing series that provides an international forum for new ideas and concepts for manipulators, vision and tactile sensor systems, control structures, algorithms, and paradigms of intelligent robots.

Eighty-one papers from eight countries are included in this proceedings. The fourteen sessions range from theoretical foundations in pattern recognition and image processing to applications of intelligent robots for textile fabric handling and automated assembly.

The sessions reflect the broad scientific base needed for the support and development of useful, intelligent machines. The first session, Pattern Recognition, includes new algorithms with an optical emphasis. The second session, Image Processing, contains a variety of new theoretical and experimental papers on imaging algorithms. The third session is on sensors and has a concentration of papers on exciting new uses of tactile sensors. The fourth session, Model-Based Object Recognition, considers the high level image understanding problems and contains excellent papers on the use of 3D models for image matching and tracking. The fifth session, on image understanding, continues the high level concentration with excellent papers on architectures and new machines. The sixth session is Artificial Neural Systems and expands the theme to adaptive networks that resemble neural networks in humans. The seventh session is on three-dimensional object recognition. Several new algorithms for 3D object measurement and recognition are presented. The next session, Multisensor Object Recognition, contains papers in the important area of fusing image data from several sensors. The ninth session, on stereo image processing, includes several new approaches to stereo vision understanding and applications. The session on optical flow contains papers on new approaches to motion estimation from images. The eleventh session is Intelligent Control and provides excellent papers on low and high level control methods for robots and intelligent machines. The twelfth session, Vision-Aided Automated Control Systems, contains papers on new expert control systems. The next session, Architectures and Software, contains papers on the new generation of machines such as the Hypercube, as well as interactive expert language and systems for imaging and robotics. The fourteenth and final session, Industrial Applications, contains some exciting papers, such as the application of mathematical morphology for vehicle detection.

The distribution of papers in the proceedings is interesting; approximately 8% of the papers were given by government researchers, about 60% of the papers were contributed by university-based researchers, and about 20% of the papers were contributed by industrial researchers. Finally, about 12% were contributed by university/industry or university/government teams. The number of cooperative papers has increased over the number presented in 1986. We believe this is a positive trend.

(continued)

This conference was held in conjunction with four other SPIE conferences: Automated Inspection and High Speed Vision Architectures (SPIE Vol. 849), Optics, Illumination, and Image Sensing for Machine Vision II (SPIE Vol. 850), Space Station Automation III (SPIE Vol. 851), and Mobile Robots II (SPIE Vol. 852). These proceedings should be of interest to readers with specific application interests. A similar conference arrangement is planned for 1988.

We gratefully acknowledge the assistance of Marlene Layton, the enthusiastic support of the conference committee, session chairs, and authors for making these contributions available.

David P. Casasent
Carnegie Mellon University

Ernest L. Hall
University of Cincinnati

INTELLIGENT ROBOTS AND COMPUTER VISION

SIXTH IN A SERIES

Volume 848

Contents

(continued)

INTELLIGENT ROBOTS AND COMPUTER VISION
SIXTH IN A SERIES

Volume 848

Conference Committee

Chairs
David P. Casasent, Carnegie Mellon University
Ernest L. Hall, University of Cincinnati

Program Committee
Bruce G. Batchelor, University of Wales (UK); **Mark F. Cullen,** Perkin-Elmer Corporation; **H. Steve Hou,** The Aerospace Corporation; **Ren C. Luo,** North Carolina State University; **Nelson Marquina,** RCA Advanced Technology Laboratories; **C. Anthony McPherson,** Avco Research Laboratory; **Richard Messner,** University of New Hampshire; **Mitchell Nathan,** University of Colorado/Boulder; **André Oosterlinck,** Catholic University of Leuven (Belgium); **Lewis J. Pinson,** University of Colorado/Colorado Springs; **Steven K. Rogers,** Air Force Institute of Technology; **Paul S. Schenker,** Jet Propulsion Laboratory; **Anne Schur,** Honeywell Inc.; **David M. Siegel,** Massachusetts Institute of Technology; **Susan Snell Solomon,** Datacube Inc.; **Donald J. Svetkoff,** Synthetic Vision Systems

Session Chairs
Session 1—Pattern Recognition, **David P. Casasent,** Carnegie Mellon University
Session 2—Image Processing, **Ernest L. Hall,** University of Cincinnati
Session 3—Sensors, **Ren C. Luo,** North Carolina State University
Session 4—Model-Based Object Recognition, **Steven K. Rogers,** Air Force Institute of Technology
Session 5—Image Understanding, **Mark F. Cullen,** Perkin-Elmer Corporation
Session 6—Artificial Neural Systems, **Richard A. Messner,** University of New Hampshire
Session 7—Three-Dimensional Object Recognition, **Mitchell Nathan,** University of Colorado/Boulder
Session 8—Multisensor Object Recognition, **Mitchell Nathan,** University of Colorado/Boulder
Session 9—Stereo Image Processing, **Bruce G. Batchelor,** University of Wales (UK)
Session 10—Optical Flow, **Bruce G. Batchelor,** University of Wales (UK)
Session 11—Intelligent Control, **Paul S. Schenker,** Jet Propulsion Laboratory
Session 12—Vision-Aided Automated Control Systems, **Paul S. Schenker,** Jet Propulsion Laboratory
Session 13—Architectures and Software, **H. Steve Hou,** The Aerospace Corporation
Session 14—Industrial Applications, **C. Anthony McPherson,** Avco Research Laboratory

Session 1

Pattern Recognition

Chair
David P. Casasent
Carnegie Mellon University

Rule-based String Code Processor

David Casasent and Sung-Il Chien

Carnegie Mellon University
Center for Excellence in Optical Data Processing
Department of Electrical and Computer Engineering
Pittsburgh, PA 15213

ABSTRACT

A new and efficient real time technique to produce a string code description of the contour of an object, such as an (angle, length) = (ϕ, s) feature space for the arcs describing the contour, is detailed. We demonstrate the use of such a description for an aircraft identification problem case study. Our (ϕ, s) feature space is modified to include a length string code and a convexity string code. This feature space allows both global and local feature extraction. The local feature extraction follows human techniques and is thus quite suitable for a rule-based processor (as we discuss and demonstrate). Aircraft have generic parts and thus are quite suitable for the model-based description.

1. INTRODUCTION

Aircraft recognition is a classic pattern recognition problem recently surveyed [1]. Many feature spaces have been suggested for such multiple degree of freedom pattern recognition problems. These include: moments [2,3] (which require large dynamic ranges and are noise sensitive when made distortion-invariant); Fourier descriptors [4,5] (which still require feature extraction, computationally intensive matching lists, and which do not lend themselves to use of local information or features); and various curvature features. Our proposed technique handles global and local features, includes feature extraction with in-plane distortion-invariance and avoids a large matching search.

We selected a string code description of the object. Other work with similar descriptions [6-9] has also been used and their VLSI realization discussed [10-12]. However, our string code description (ϕ, s) = (angle, length) of the arcs on the contour of an object is generated most efficiently and allows global and local feature space analysis. Global features are necessary for general problems and local features allow specific problems to be solved quite effectively. The local features we use correspond to specific object parts and thus allow rule-based analysis (since this is the manner in which humans achieve identification). Our edge description is different from the conventional chain code [9] and we do not convert the chain code to an (x, y) or other description as others [7] do early in the processing period. Our rule-based technique differs from syntactic [13] techniques. Our rule-base follows a forward chaining control flow as does SPAM [14]. As our model knowledge, we employ specific aircraft structural and part information.

Section 2 describes our case study, model base, and data base. Section 3 provides an introduction and overview of our processor and our feature space. Section 4 details our new efficient feature space generation technique and includes typical results. Section 5 briefly discusses our rule-based processor.

2. DATA BASE

The case study we consider is the identification and orientation estimation of 10 different aircraft. Fig.1 shows the top-down views of these aircraft grouped by the functional role of the aircraft. In our tests, all aircraft are 128 x 128 pixels in resolution. Our model base contains different polygon descriptions of all aircraft and their parts, from which any aspect view can be produced quite easily [15].

3. PREPROCESSOR OVERVIEW

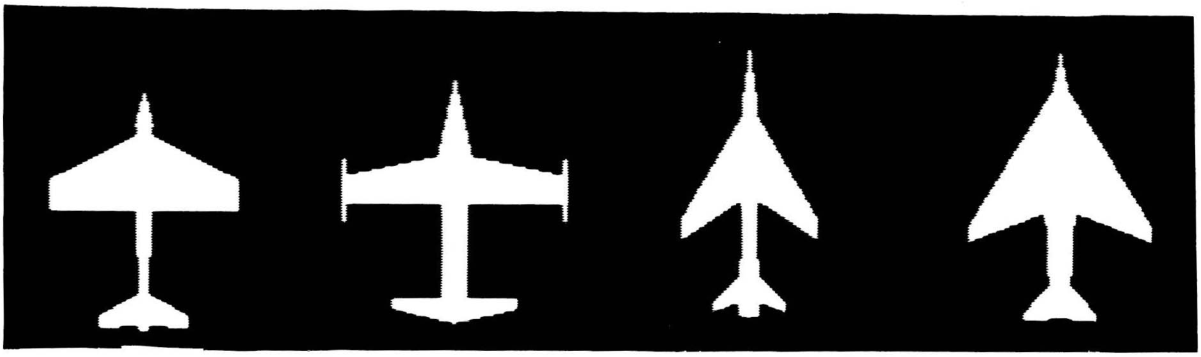
Our full processor contains five major sections as shown in Fig.2. The preprocessor performs edge enhancement (this is necessary to produce good peaks in the Hough transform space we will employ) and generates a clockwise ordered list of pixel coordinates for the contour or boundary of the object (using classic techniques [16,17]). The feature space produced is a (ϕ, s) description of the angle (ϕ) and the length (s) of all arcs clockwise in a string code connected object boundary or contour description. An aspect estimator unit determines if the aircraft is being viewed nearly top-down or if an out-of-plane distorted image is being investigated. A rule-based or an associative processor are used (depending upon the aircraft object's distortions). In this present paper, we discuss the rule-based processor. Thus, in this initial work, we will restrict attention to nearly top-down aircraft views.

4. EFFICIENT (ϕ, s) STRING CODE FEATURE SPACE GENERATION

The first step is to reduce the clockwise ordered contour pixel list to N (approximately 20-30) vertices. Fig.3 shows a DC10 (Fig.3.a) and its boundary description with the vertices noted (Fig.3.b). The N vertices define N arcs for the boundary, each with a length (s) and an internal angle (ϕ). Fig.3.c defines the angle ϕ. The result is a (ϕ, s) string code.

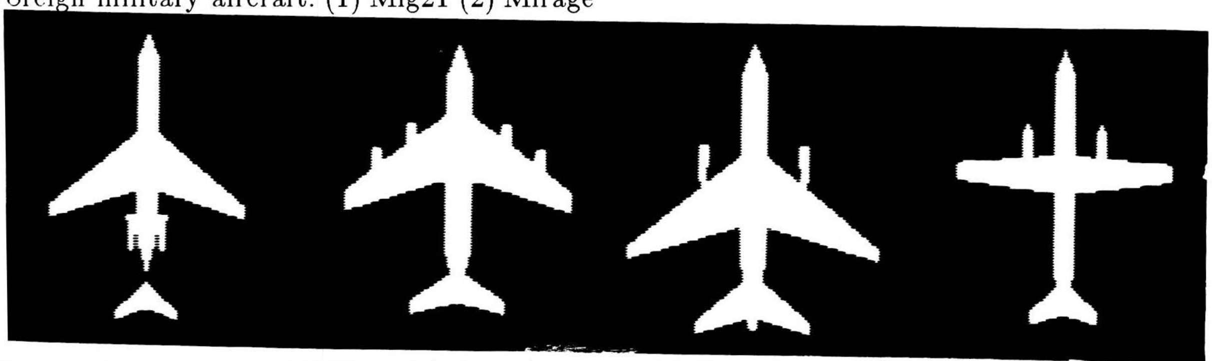
The block diagram of our efficient (ϕ, s) string-code generation system is shown in Fig.4. We use the clockwise-ordered contour list of the boundary pixels (x, y), form the Hough transform (HT) of the input from the original data, and locate the six major (and true) HT peaks and their (p, θ) values. We then Hough transform each contour pixel and check if it evokes a peak at one of the (p, θ) six major HT peak parameter locations. This assigns most contour points to the six major lines in the image and gives automatically (without time-consuming trigonometric operation) the angle ϕ and the length (s) of these lines. Only a small fraction of the pixel points in the contour list remain to be assigned ϕ and s values. Each of these is a connected set of pixels that lies in a gap between previously assigned points. We achieve the (ϕ, s) description of these pixels into lines by a conventional split-line fitting method [18,19]. This split-line technique is computationally expensive, but (with the six major lines and our HT technique) this needs only to be applied to a significantly reduced number of points in the contour list. Thus, this technique generates the full (ϕ, s) string code description quite efficiently.

A HT converts lines in the input into points in a (p, θ) parameter Hough space, i.e. at coordinates corresponding to the normal distance (p) and the angle (θ with respect to the x axis) of the normal of the line, with six peak heights proportional to the number of points on the line (or the length of the



(a) U.S.A. military aircraft: (1) B57 (2) F104 (3) F105 (4) Phantom

(b) Foreign military aircraft: (1) Mig21 (2) Mirage



(c) Commericial airliners: (1) B727 (2) B747 (3) DC10 (4) Swearingen

Figure 1: Image Data Base (128 x 128)

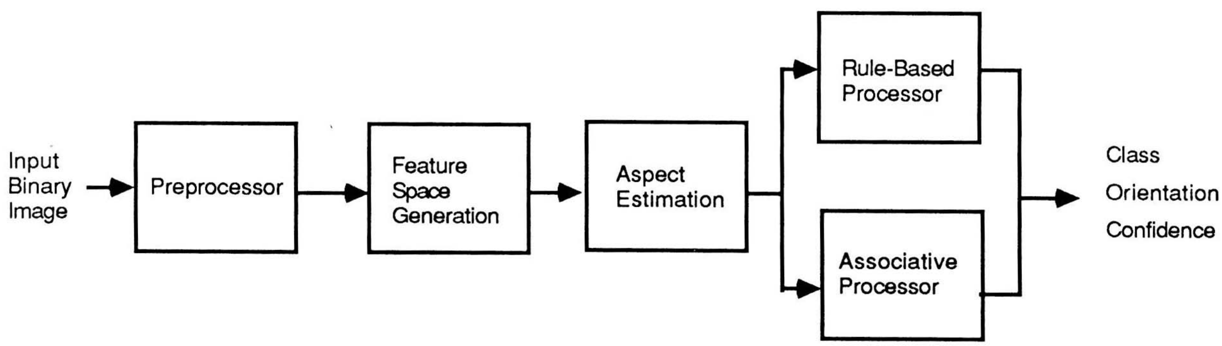
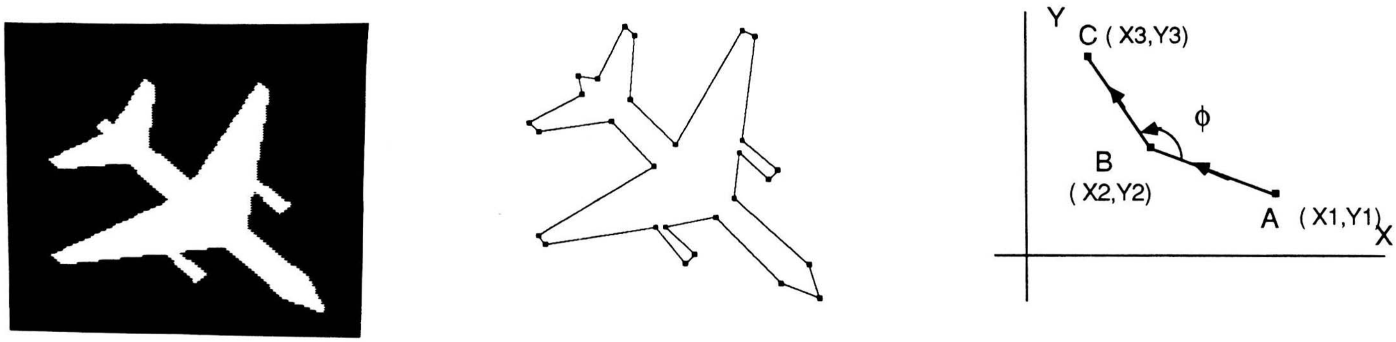


Figure 2: Overall Processor



(a) DC10 (b) DC10 vertices (c) Angle ϕ Definition

Figure 3: Example of vertices describing an object boundary (Fig.3 a and b) as arcs of length s and internal angle ϕ (ϕ is defined in Fig. 3.c)

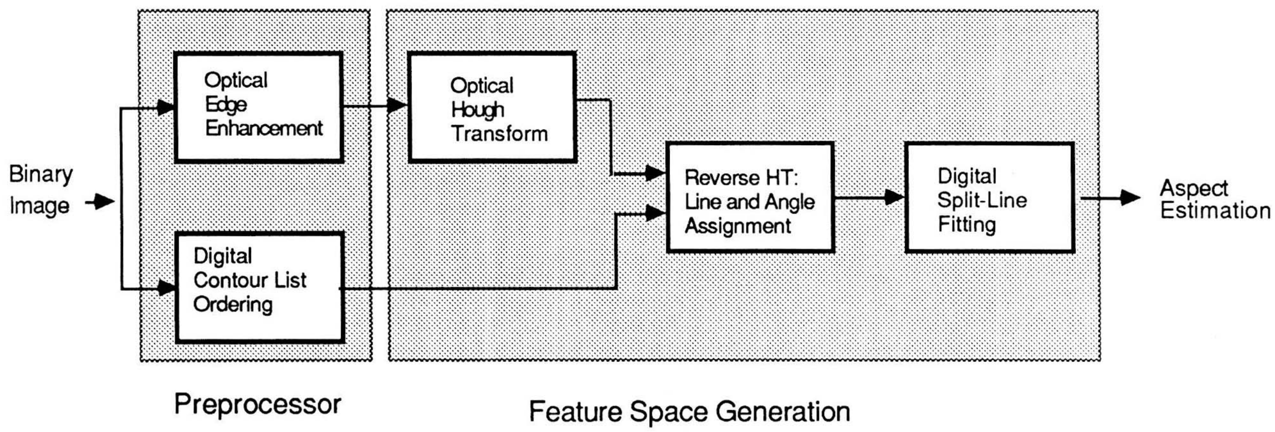


Figure 4: Block diagram of an efficient (ϕ, s) string code processor

line). Fig.5.a shows the HT for the DC10 with the nose vertical. Fig.5.b shows the HT for the DC10 with the nose horizontal. The two major peaks in Fig.5.a lie on the $\theta = 0°$ line and in Fig.5.b they lie on the $\theta = 90°$ line. These two major peaks denote the presence of the fuselage and its orientation. In Fig.5, we see six major peaks, however this does not always occur (when noise, quantization of the image resolution, and 3-D roll and pitch distortions occur). To demonstrate this and techniques to overcome these problems, we show (in Table 1) the 10 largest HT peaks obtained for the DC10 oriented at $120°$. This demonstrates specifically that the largest six HT peaks do not correspond to the major lines in the image, specifically HT peak 6 and 7 are false peaks that are larger than peak 8 (which is the next largest true peak). We note [20] that such false peaks occur close to the true peak (within three pixels for our aircraft data). Thus, we employ an algorithm that ignores HT space peaks that lie within four pixels of the large peak. Employing this rule, the six proper peaks corresponding to the six major lines in the aircraft image emerged (Table 2). Table 3 lists the six aircraft lines corresponding to the six major HT peaks and Fig.6.a shows the lines in the aircraft image itself. Fig.6.b shows the resultant final (ϕ, s) image with all vertices obtained (including those obtained by the split-line fitting technique).

An efficient technique to assign the θ and p parameters of the six HT peaks to point in the contour list is now detailed. To achieve this, we transform each pixel coordinates (x, y) in the clockwise contour list into a sinusoid. This sinusoid needs only be evaluated at the six θ values of the six dominant HT peaks and at the p coordinates within each. Thus, these HT operations on the contour list are easily achieved. Since we expect a number of successive pixels in the contour list (those for each arc) to correspond to the same HT peak point, the processor can be quite fast (and very efficient, compared to typical techniques involving extensive trigonometric calculation).

We now discuss the descriptions we employ of the string code representation of the object as a symbolic descriptor. We first consider the full (ϕ, s) string code with the exact analog values for all angles and lengths. Next, we consider a convexity string code. This lists only the convexity of the angles of the arcs in the boundary representation as convex V (if $\phi < 180^\circ$) or concave C (if $\phi > 180^\circ$). Last, we consider a length string code which lists only the length of each arc as : very short, short, medium, long, and very long. These are expressed in terms of maximum difference $\Delta = L_{max} - L_{min}$ in the length L of the arcs for the input image. Each length region is $\Delta/6$ except for the medium length region which is $\Delta/3$ in extent. These different symbolic string code descriptions of the object contour are found to be quite useful for global and local rule-based processing, as described in Section 5.

5. RULE-BASED PROCESSOR

Our rule-based system employs if-then rules, a context-limited and rule-ordered control strategy and forward chaining with five rule groups used as we now describe. The first rule group (starting rules) locates the fuselage.

The second rule group concerns substructure search rules. The purpose of this second rule group is to locate all separate regions of an object and to divide them into left (L) and right (R) regions with respect to the fuselage. We first extract the fuselage and all vertices corresponding to it. This separates the contour list into L and R regions. We group these into separate connected regions (closed polygon boundaries) corresponding to parts of the object. For each such region, we calculate its area, perimeter, compactness, and its position with respect to the fuselage. Various rules are used to determine the type of each region. Three representative examples are given below:

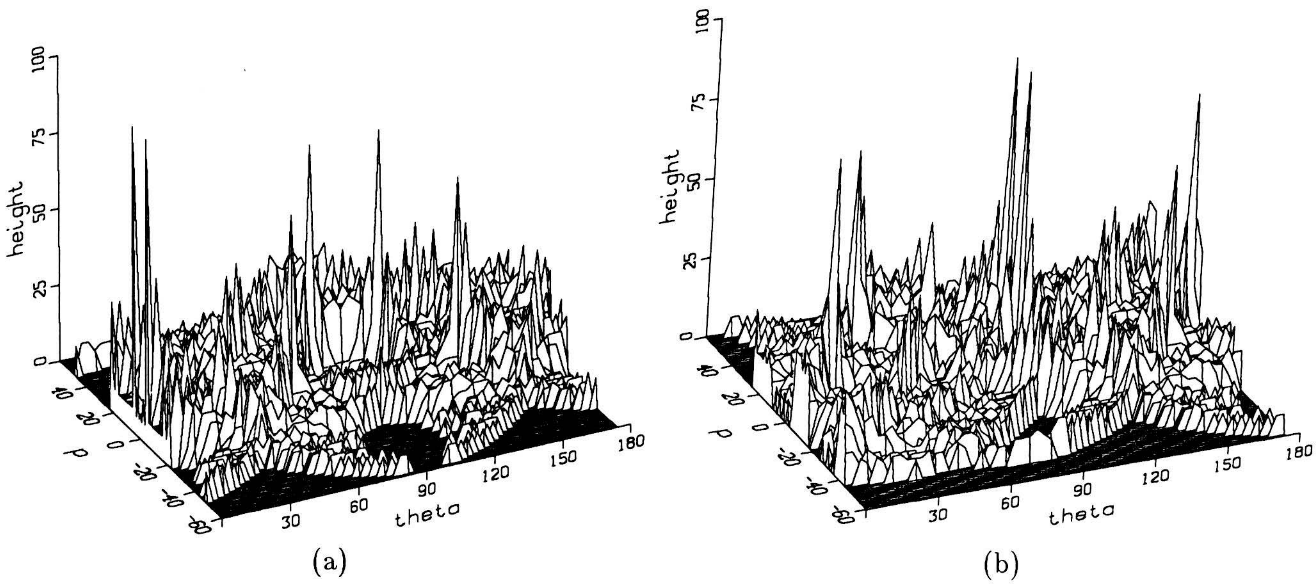


Figure 5: HT of DC10 with nose oriented vertical (a) and horizontal (b)

Hough Peak	p(pixel)	θ(degree)	Peak Height
1	3	165	100
2	-19	114	95
3	-5	60	92
4	19	6	92
5	5	60	87
6	-20	111	72
7	20	9	67
8	3	135	65
9	-7	63	55
10	-5	162	52

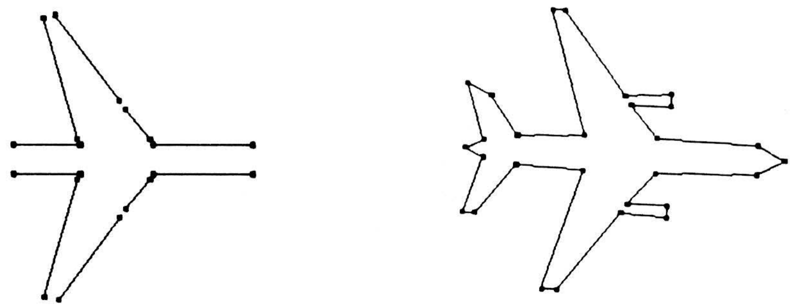
Table 1: Data on the 10 largest peaks for a DC10 with its nose at 120 °

Hough Peak	p(pixel)	θ(degree)	Peak Height
1	3	165	100
2	-19	114	95
3	-5	60	92
4	19	6	92
5	5	60	87
6	3	135	65

Table 2: Data on the six largest HT peaks using our false peak algorithm. The six peaks noted are the correct ones.

Corresponding Aircraft Part
Right Line on Fuselage
Left Line on Fuselage
Right Front Wing Line
Right Rear Wing Line
Left Front Wing Line
Left Rear Wing Line

Table 3: 6 major lines in an aircraft



(a) (b)

Figure 6: Aircraft Image with (a) only the six major arcs and (b) all arcs

Rule 1: Wings are the largest regions in L and R. They must have the proper spatial relationship to the fuselage.

Rule 2: If the convexity symbolic code for a region has all vertices convex, then this region is a wing with no engines etc on it.

Rule 3: If the convexity symbolic code for a region has two concave vertices out of four adjacent vertices and if this correspond to short arcs, then this region is a wing with an engine etc on it.

From the location of the concave vertices and arcs of short length, the position of the engine etc (refered to as a "blob") or small structure on the wing (or fuselage) can be determined. We discuss this further below. Fig.7 shows examples of a wing region with no engine (Fig.7.b) as detected from its convexity code (Fig.7.a). Fig.8 shows an analogous example when the convexity code (Fig.8.a) shows several C sections and hence indicates the presence of an engine in the image of Fig.8.b. Following such rules, we can segment the L and R regions into parts as shown in Fig.9 (wings, tails, and blobs).

The third rule group we use provides a check on the top-down orientation estimation (this is obtained from the number of regions in L and R, the areas of these regions, and the symmetry of the L and R sections), yaw estimates (these are obtained from the θ coordinate of the fuselage peak in the HT space), and roll estimates (from the symmetry or ratios of areas in regions L and R).

The fourth rule group concerns substructure rules. These are intended to identify the small or local features or object regions or parts. The best example of this concerns "blobs" on wings and specifically whether these are engines, missiles, or fuel tanks. For the image data base we considered, we note (from Fig.1) that if the blobs appear in the center of the wing, the blob is an engine (e.g. DC10); and if it appears on the tip of a wing, it is a missile (e.g. F104).

The fifth rule group contains classification rules. We note three examples below. There are approximately 40 rules used in total.The following are intended to be representative examples. Before