A Model-based Approach for Detection of Runways and Other Objects in Image Sequences Acquired Using an On-board Camera

Final Technical Report for NASA Grant NAG -1-1371
"Analysis of Image Sequences from Sensors for Restricted Visibility Operations"
Period of the Grant: January 24, 1992 to May 31, 1994

Submitted to
NASA Langley Research Center
Technical Officer: Randall L. Harris, Sr.
Flight Management Division, Mail Stop 152 E
Hampton, Virginia 23681

by

Rangachar Kasturi
Principal Investigator
Associate Professor of Computer Science and Engineering
The Pennsylvania State university
University Park, Pennsylvania 16802
Tel: (814) 863-4254 Fax: (814) 865-3176
E-Mail: kasturi@cse.psu.edu

Graduate Students:
Sadashiva Devadiga
Yuan-Liang Tang

August 18, 1994

A Model-based Approach for Detection of Runways and Other Objects in Image Sequences Acquired Using an On-board Camera

Final Technical Report for NASA Grant NAG -1-1371
"Analysis of Image Sequences from Sensors for Restricted Visibility Operations"
Period of the Grant: January 24, 1992 to May 31, 1994

Submitted to
NASA Langley Research Center
Technical Officer: Randall L. Harris, Sr.
Flight Management Division, Mail Stop 152 E
Hampton, Virginia 23681

by

Rangachar Kasturi
Principal Investigator
Associate Professor of Computer Science and Engineering
The Pennsylvania State university
University Park, Pennsylvania 16802
Tel: (814) 863-4254 Fax: (814) 865-3176
E-Mail: kasturi@cse.psu.edu

Graduate Students:
Sadashiva Devadiga
Yuan-Liang Tang

August 18, 1994

A Model-based Approach for Detection of Runways and Other Objects in Image Sequences Acquired Using an On-board Camera

Preface

This research was initiated as a part of the Advanced SenSor and Imaging System Technology (ASSIST) program at NASA Langley Research Center. The primary goal of this research is the development of image analysis algorithms for the detection of runways and other objects using an on-board camera. Initial effort was concentrated on images acquired using a passive millimeter wave (PMMW) sensor. The images obtained using PMMW sensors under poor visibility conditions due to atmospheric fog are characterized by very low spatial resolution but good image contrast compared to those images obtained using sensors operating in the visible spectrum. Algorithms developed for analyzing these images using a model of the runway and other objects are described in Part I of this report. Experimental verification of these algorithms was limited to a sequence of images simulated from a single frame of PMMW image. Subsequent development and evaluation of algorithms was done using video image sequences. These images have better spatial and temporal resolution compared to PMMW images. Algorithms for reliable recognition of runways and accurate estimation of spatial position of stationary objects on the ground have been developed and evaluated using several image sequences. These algorithms are described in Part II of this report. A list of all publications resulting from this work is also included.

Table of Contents

List of Publications Resulting from this Work

Kasturi, R. and R. L. Harris, "Analysis of Simulated Image Sequences from Sensors for Restricted Visibility Operations", presented at the *Tenth IEEE/AIAA Digital Avionics Systems Conference*, Los Angeles, October 1991.

Devadiga, S. and R. Kasturi, *Real-time Implementation of PMMW Image Sequence Processing: A Feasibility Study*, Report for NASA Grant NAG-1-1371, September 1993.

Devadiga, S., Tang, Y.-L., and R. Kasturi, *Sensor Positional Sensitivity Evaluation*, interim research report for NASA Grant NAG-1-1371, NASA Langley Research Center, July 1992.

Tang, Y.-L. and R. Kasturi, "Accurate Estimation of Object Location in an Image Sequence Acquired Using a Moving Camera," *NASA Goddard Conf. Space Applications of Artificial Intelligence*, May 1993, pp. 147–157.

Tang, Y.-L. and R. Kasturi, "Accurate Estimation of Object Location in an Image Sequence Acquired Using a Moving Camera," to appear *in Robotics and Computer Integrated Manufacturing* Journal 11(4), 1994.

Tang, Y.-L. and R. Kasturi, "Runway Detection in an Image Sequence" submitted to *SPIE Conference on Image and Video Processing* San Jose, February 1995.

Tang, Y.-L. and R. Kasturi, *An Airborne Vision System for Runway Recognition and Obstacle Detection*, Computer Science and Engineering Technical Report, CSE-94-045, August 1994.

Tang, Y.-L., S. Devadiga, and R. Kasturi, "Model-Based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images", *Image and Video Processing II, Proc. SPIE, Vol 2181*, San Jose, February 1994, pp. 320–330.

Tang, Y.-L., S. Devadiga, and R. Kasturi, *A Knowledge-Based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images*, Computer Engineering Technical Report, TR-93-118, February 1993.

Tang, Y.-L., S. Devadiga, and R. Kasturi and R. L. Harris, Sr., "A Knowledge-Based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images," *Proc. Augmented Visual Display (AVID) Research Workshop*, NASA Ames Research Center, March 1993, NASA Conference Publication 10128, pp. 313–328.

Part I

Model-based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images

I. Model-based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images

Abstract

In this part of the report we describe a knowledge-based vision system to assist the pilots in landing maneuvers under restricted visibiltiy conditions. The system has been designed to analyze image sequences obtained from Passive Millimeter Wave (PMMW) imaging system mounted on the aircraft to delineate runways/taxiways, buildings, and other objects on or near runways. PMMW sensors have good response in foggy atmosphere; but thier spatial resolution is very low. However, additional data such as airport model and approximate position and orientation of aircraft are available. We exploit these data to guide our knowledge-based system to locate objects in the low resolution image and generate warning signals to alert pilots. We also derive analytical expressions for the accuracy of the camera position estimate obtained by detecting the position of known objects in the image.

1. Introduction

Federal regulations specify the minimum visibility conditions under which airlines may take off and land. These minima are a function of the types of airplane and airport equipment. Therefore, there is a great deal of interest in imaging sensors which can *see through* fog and produce a *real world* display which, when combined with symbolic or pictorial guidance information, could provide the basis for a landing system with lower visual minimum capability than those presently being used (Hatfield & Parrish, 1990).

Since the energy attenuation in the visible spectrum due to fog is very large (Young et. al., 1990/1991), sensors are being designed to operate at lower frequencies (e.g. 94 GHz) where the attenuation is lower providing the ability to see through fog. NASA Langley Research Center, in cooperation with industry, is performing research on an on-board imaging system using a passive sensor operating at this frequency. Images from such sensors are of very low spatial resolution (Fig. 1.1). However, additional supporting information in the form of knowledge about the airport and the position, orientation and velocity of aircraft is generally available. Thus a model-based image analysis approach is feasible to segment the image and to detect and track objects on the ground. Information extracted from such an analysis is useful to generate warning signals to the pilot of any potential hazard. This part of the report describes such a model-based technique, which makes use of a priori information about the geometric model of the airport and camera

position and attitude data provided by the Global Positioning System (GPS) and other instruments.

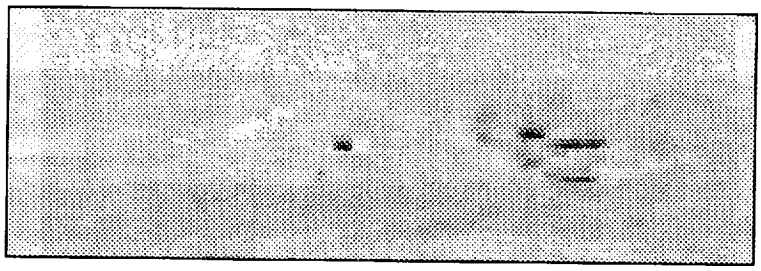


Fig. 1.1: The Passive Millimeter-Wave image.

The geometric model of the airport contains positions of the runways/taxiways and buildings, the navigation instruments provide the position of the aircraft, and on-board instruments provide the orientation of the aircraft (yaw, pitch and roll). We use this information to define regions of interest in the image where important features such as runways/taxiways, the horizon, etc. are likely to be present. Edges corresponding to these features of interest are detected within these regions. After delineating regions representing runways/taxiways, we look for objects inside and outside these regions.

The data from radio navigation instruments are known only upto a certain accuracy depending upon the type of radio navigation instruments. For example, GPS data is updated once every second and it is likely that a few such updates are missed making camera position data to be a few hundreds feet off. On-board instrument data is generally

useful to obtain more accurate camera position data than the GPS-based data. An alternative approach is to use the information about the location of detected objects in the images with known world coordinates (e.g. intersection of runways/taxiways, corners of buildings, etc.) to obtain an improved estimate of the camera position. This requires an analytical study of the relationships among the camera parameters, the resolution of the images, and the distances between the aircraft and objects.

Dickmanns (1988) described a computer vision system for flight vehicles. The main emphasis was to build a vision system which can achieve real-time performance with the microprocessors available at that time. In order to speed up the system, it is extremely important to avoid processing those regions which provide little information for scene understanding or navigation. Hence, having the runway geometric model is useful in locating regions of interest and concentrate processing only within those regions. Smith et. al. (1992) developed a vision system which assist the pilot during low-altitude flight to detect and locate obstacles near the helicopter's intended flight path. The system recursively estimates range using an extended Kalman filter with knowledge of the camera's motion, giving incremental update of obstacles found in the flight path. Sull and Ahuja (1994) proposed a system for recovering motion and structure parameters from a monocular flight image sequence. The system also uses intermediate results of the recovery process to synthesize an image sequence that depicts the motion and structure. The key feature of their approach is an integrated use of multiple image attributes or cues,

such as regions, point features, optical flow, texture gradient, and vanishing line, making the estimation more robust. Our approach makes use of a priori knowledge of the runway model which is useful in reducing unnecessary processing of images, in locating the runway in order for the aircraft to be heading in the right direction, and in detecting objects of interest on the ground. The system will report warning signals for objects which may jeopardize the landing.

In Section 2 we present a block diagram of the complete system. In Section 3 we describe the analytical model that establishes the relationship between the position, orientation and other physical parameters of the camera and the attributes of the captured images. This model is useful to calculate the accuracy of camera position estimation using image based features. In Section 4 we present the method for defining the regions of interest in the image using the camera parameters and airport model. Section 5 includes image processing steps that are used to find regions corresponding to major features in the image and to detect objects in these regions. Experimental results are presented in Section 6. We conclude the paper with a summary and a brief description of future work.

2. System Description

In this section, we describe the functions of various modules of the system shown in Fig. 2.1 and the interaction between them. The input model of the airport contains positions of the runways/taxiways, and buildings. The *model transformation* module will

take this model and the camera state information (position and orientation) as inputs to define the regions of interest in the image plane.

The image processing algorithms in the *feature detection* module operates within these regions of interest to detect the edges of the runway, horizon, etc. in the image. An edge is fitted to the edge pixels if enough edge pixels are found within the regions of interest. The module outputs parameters which define major regions in the input image.

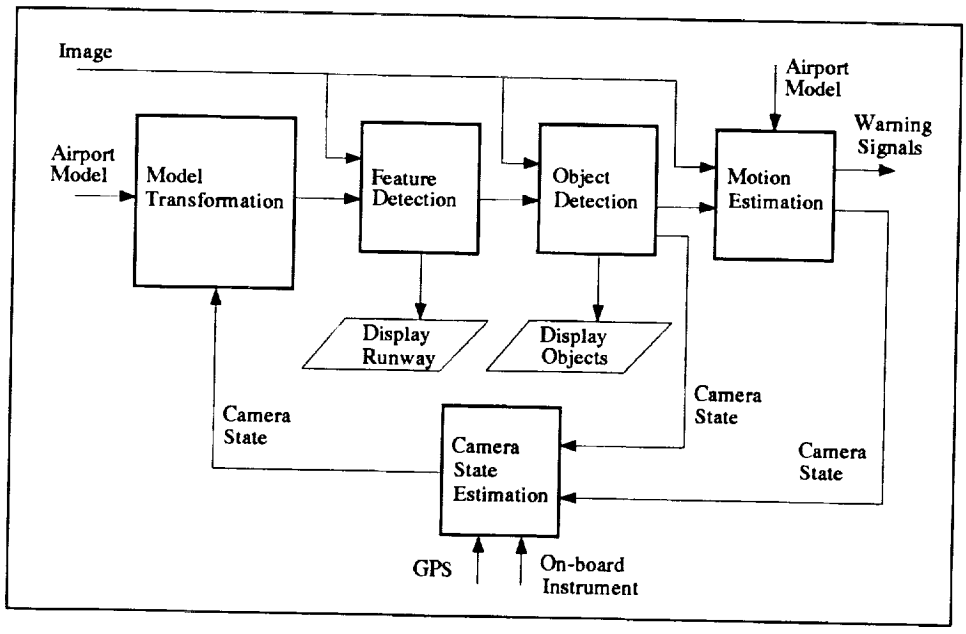


Fig. 2.1 System Block diagram

The *object detection* module detects objects in the image using different thresholds for each region. For example, since detection of objects on the runway is extremely

important, a lower threshold is used to flag every object even if the contrast is low whereas a higher threshold is used to detect objects which are outside the runway such as buildings, etc. Locations of detected objects with known world coordinates is useful to estimate camera state parameters.

The *motion estimation* module uses dynamic scene analysis methods to estimate camera state parameters as well as to detect velocities of objects on the ground. The outputs from this module will be useful to detect potential collisions and generate warning signals as appropriate.

The *camera state* estimation module integrates information obtained about the position and velocity of the aircraft from various sensors and modules and outputs necessary data to the *model transformation* module.

3. Accuracy of Camera State Estimation from Image-based Features

As we need to use the camera state estimated from locating features of known objects in the image during the period when the GPS is not updated, it is necessary to know the accuracy of such estimated positions and the factors that decide the accuracy. Hence, an analytical model that establishes the relationship between the camera parameters and the attributes of captured images is necessary for guiding the image analysis system. Sensor positional parameters include range (distance from the aircraft to the runway threshold),

cross range (distance from the aircraft to the runway center line), altitude, and pitch, roll and yaw angles. Sensor imaging attributes include the number of pixels in the image and the optical angular view measured in degrees. We derive the inter-relationships among these parameters. Using these relationships we calculate the accuracy of the estimate of camera position based on a minimum resolvable movement of features by one pixel in the image. We obtain these accuracies for three different types of cameras (PMMW, FLIR, HDTV) at six ranges.

3.1. Analysis

Throughout the analysis, for convenience, we assume that the sensor is located at the center of gravity of the airplane. Hence we can use the terms sensor position and aircraft position interchangeably. We also neglect the effect of curvature of the earth. The system of reference axis that forms the basis of system of notations used to describe the position of the sensor is shown in Fig. 3.1 The figure shows an airplane with three mutually perpendicular axes—pitch, roll and yaw—passing through the center of gravity of the airplane. The image plane is assumed to be perpendicular to the rolling axis with its vertical and horizontal axes coinciding with the yawing and the pitching axis of the airplane, respectively.

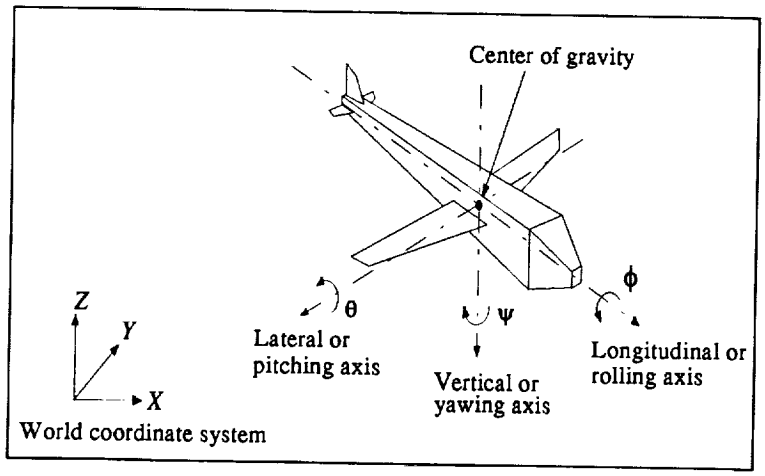


Fig. 3.1 Airplane body axes

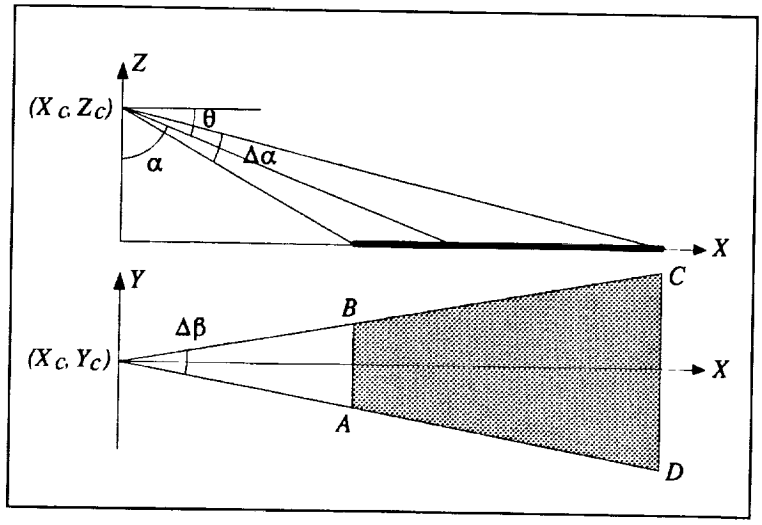


Fig. 3.2 Image obtained by the sensor projected towards the ground. Shaded area is the ground area covered by the sensor

Fig. 3.2 shows an imaging situation during landing where the aircraft is at (X_C, Y_C, Z_C), with pitching angle θ, zero yaw and zero roll angle. Let $\alpha = 90°-\theta$. The field of view of the camera is determined by two viewing angles: $\Delta\alpha$ defined in the same plane as θ and $\Delta\beta$ at right angles to $\Delta\alpha$ ($\Delta\alpha$ determines the vertical extent of the image and $\Delta\beta$ its horizontal extent). Even though the image obtained by the sensor is always a rectangle, the ground area captured by the sensor is a trapezoid *ABCD* whose side length and area depends on $\Delta\alpha$, $\Delta\beta$ and various other sensor parameters like position, orientation etc. Note that a pixel in the image plane corresponds to a patch on the ground plane. We refer to this as a pixel-patch (see Fig. 3.3).

Consider a point feature which has been detected at some pixel (p, q). Let the actual world coordinates of this feature be $(P, Q, 0)$. Since a pixel represents a patch on the ground, the camera could change in its position by certain amount while still retaining the image of the feature at the same pixel (p, q). Hence a camera pose estimation by passive triangulation will always give the same camera pose for nearby camera positions unless the change in camera position is large enough for the feature to be observed in the neighboring pixel. We define this minimum change in camera displacement as the sensitivity of the camera. Note that this is a measure of accuracy of camera position estimate and is a function of the camera, image size in number of pixels, angular resolution, and the pixel location (p, q) in the image plane.

Let N_x and N_y represent the number of pixels in the vertical and horizontal directions, respectively. The pixels are numbered $-N_x$, ..., 0, ..., $N_x/2-1$ in the vertical direction and $-N_y$, ..., 0, ..., $N_y/2-1$ in the horizontal direction. The rolling axis of the plane is assumed to pass through the bottom right corner of the patch on the ground plane which corresponds to the center pixel in the image plane. Other pixels are referenced in a similar manner. The coordinates of the reference corner of the ground area covered by a pixel (p, q) can be estimated by the following relations.

$$X = X_c + Z_c \tan(\alpha + p\frac{\Delta\alpha}{N_x}),$$
$$Y = Y_c + \frac{Z_c}{\cos(\alpha + p\frac{\Delta\alpha}{N_x})} \tan(q\frac{\Delta\beta}{N_y}). \qquad (3.1)$$

For a non zero rolling angle ϕ, the ground coordinates (X', Y') which corresponds to a pixel (p, q) in the image plane are obtained by replacing (p, q) in the above equation by (p', q'), where

$$p' = p\cos\phi - q\sin\phi, \text{ and}$$
$$q' = p\sin\phi + q\cos\phi. \qquad (3.2)$$

Since a pixel-patch is referenced by its bottom right corner of the pixel, the other three corners become the reference of its three neighboring pixels-patch as shown in Fig. 3.4. Thus, the four corners of this pixel-patch (X_i',Y_i'), i = 1, 2, 3, 4, are obtained by using Eq. (3.1), where (p, q) are replaced by (p_i',q_i'), where

$$p_i' = p_i \cos\phi - q_i \sin\phi,$$
$$q_i' = p_i \sin\phi + q_i \cos\phi, \qquad (3.3)$$

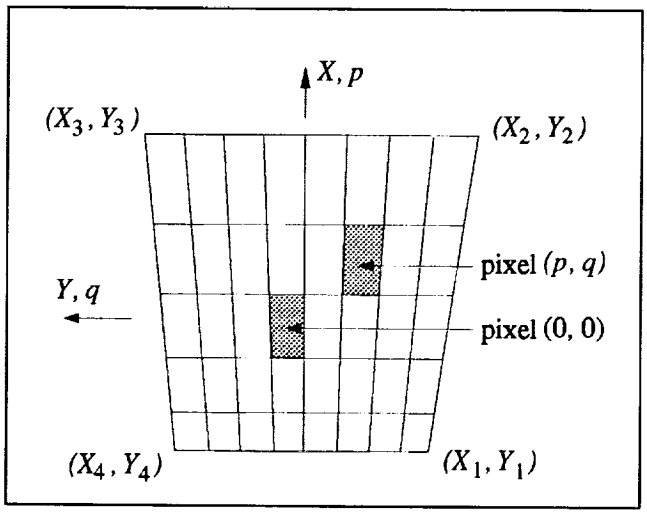


Fig. 3.3 Ground area covered by the senosr. Each small trapezoid corresponds to a pixel in the actual image

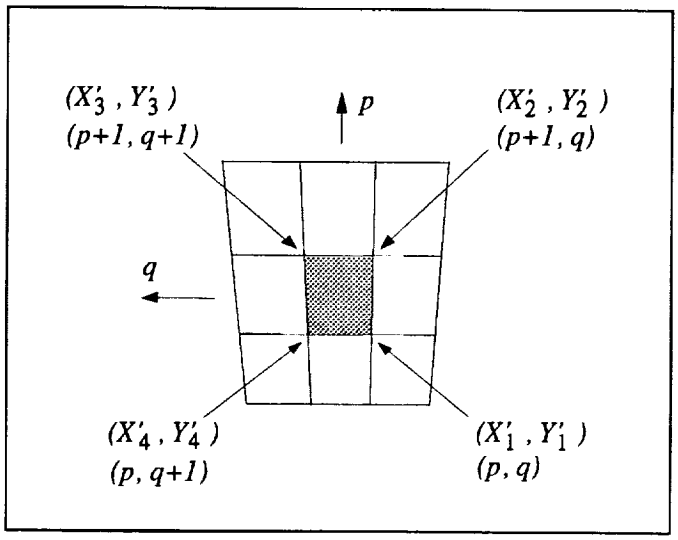


Fig. 3.4 A pixel (p, q) projected towards the ground

Eq. (3.1) explicitly gives the relationship between the camera parameters (X_C, Y_C, Z_C), θ, ϕ, and a ground point corresponding to a pixel (p, q). We are now interested in computing the sensitivity of the imagery sensor. This is defined as the minimum change in a camera parameter that would move a fixed ground point to the next pixel in the image plane. We obtain this by taking the partial derivative of X_1' and Y_1' with respect to the corresponding parameters. For example,

$$D_{X_c}^{X} = \frac{\partial X_1'}{\partial X_c}, \text{ and } D_{X_c}^{Y} = \frac{\partial Y_1'}{\partial X_c}. \tag{3.4}$$

This derivation is an approximation to the amount of change in X_1' for unit change in X_C. Thus we estimate that the amount of change in X_C in order to change X_1' to X_2', or Y_1' to Y_4' (which define the corners of adjacent pixels) as

$$S_{X_c}^{X} = \frac{(X_2' - X_1')}{D_{X_c}^{X}}, \text{ and } S_{X_c}^{Y} = \frac{(Y_4' - Y_1')}{D_{X_c}^{Y}}. \tag{3.5}$$

Note that $S_{X_c}^{Y} = \infty$, as expected. Sensitivity with reference to other parameters is defined in a similar manner. These are summarized in Table 3.1.

Sensor sensitivity is a function of various sensor parameters and sensor attitudes. Since the sensor plane is inclined to the ground plane, the sensitivity varies in the vertical and horizontal direction along the sensor plane and hence is a function of pixel number (p, q). Equivalently, the accuracy of estimation sensor position using ground truth data is a

Table 3.1: Sensor positional sensitivity equations.

SPP	Sensor Sensitivity at (p, q)		Sensor Sensitivity at (0,0) with ϕ=0
X_c	$S_{X_c}^X$	$\frac{2Z_c \sin(\cos\phi \cdot \Delta\alpha / N_x)}{\cos(2\alpha + \Delta\alpha / N_x (2p+1)\cos\phi - 2q\sin\phi)+1}$	$\frac{2Z_c \sin(\Delta\alpha / N_x)}{\cos(2\alpha + \Delta\alpha / N_x)+1}$
	$S_{X_c}^Y$	∞	∞
Y_c	$S_{Y_c}^X$	∞	∞
	$S_{Y_c}^Y$	$\{Z_c \tan(q_4'\Delta\beta / N_y) / \cos(\alpha + p_4'\Delta\alpha / N_x)\} - \{Z_c \tan(q_1'\Delta\beta / N_y) / \cos(\alpha + p_1'\Delta\alpha / N_x)\}$	$\frac{2Z_c \sin(\Delta\beta/N_y)}{\cos\alpha\cos(\Delta\beta/N_y)+1}$
Z_c	$S_{Z_c}^X$	$S_{X_c}^X / \tan(\alpha+p_1'\Delta\alpha/N_x)$	$2Z_c \sin(\Delta\alpha / N_x) / \sin(2\alpha / N_x)$
	$S_{Z_c}^Y$	$S_{Y_c}^Y \cos(\alpha+p_1'\Delta\alpha/N_x)/\tan(q_1'\Delta\beta/N_y)$	∞
θ	S_θ^X	$S_{X_c}^X \cos^2(\alpha+p_1'\Delta\alpha/N_x)/Z_c$	$\sin(\Delta\alpha / N_x) / \{\cos\alpha / \cos(\alpha + \Delta\alpha / N_x)\}$
	S_θ^Y	$\frac{S_{Y_c}^Y \cos^2(\alpha+p_1'\Delta\alpha/N_x)}{Z_c \tan(q_1'\Delta\beta/N_y)\sin(\alpha+p_1'\Delta\alpha/N_x)}$	∞
ϕ	S_ϕ^X	$\frac{2Z_c \sin(\cos\phi \cdot \Delta\alpha / N_x)}{\cos(2\alpha + \Delta\alpha / N_x (2p+1)\cos\phi - 2q\sin\phi)+1}$	∞
	S_ϕ^Y	$S_{Y_c}^Y / Zc(A\delta B\delta\phi + B\delta A / \delta\phi)$	∞

$A = 1/\cos(\alpha + p_1'\Delta\alpha / N_x)$; $B = \tan(q_1'\Delta\beta / N_y)$; $\delta B / \delta\phi = (p\cos\phi - q\sin\phi)(\Delta\beta / N_y)\cos^2(q_1'\Delta\beta / N_y)$;
$\delta A/\delta\phi = \tan(\alpha+p_1'\Delta\alpha/N_y)(-p\sin\phi - q\cos\phi)(\Delta\alpha/N_x)\cos(\alpha+p_1'\Delta\alpha/N_x)$; $\alpha = 90° + \theta$;
$(p,q_1)=(p,q)$; $(p_4,q_4)=(p,q+1)$; $p_1' = p_1\cos\phi - q_1\sin\phi$; $p_4' = p_4\cos\phi - q_4\sin\phi$; $q_1' = p_1\sin\phi + q_1\cos\phi$;
$q_4' = p_4\sin\phi + q_4\cos\phi$;

SPP: Sensor Positional Parameters

(X_c, Y_c, Z_c)	Sensor position
θ	Pitch angle
ϕ	Roll angle

	Sensor Characteristics	
	Vertical	Horizontal
Field of view	$\Delta\alpha$	$\Delta\beta$
Number of pixels	N_x	N_y

Sensitivity: Minimum change in the sensor positional parameters (X_c, Y_c, Z_c, θ, ϕ) that will make the object to appear in the next pixel either in the vertical (X: hence called as sensitivity in x direction) or in the horizontal (Y: hence called as sensitivity in y direction) direction of the sensor plane. S_i^j: Sensitivity in the direction 'j' due to the sensor positional parameter 'i' computed at pixel (p, q) in the image plane.

function of pixel position as well as other parameters. For a given range, the estimation using features that are observed at the top half of the sensor are less accurate because of the large ground area represented by these pixels. Also for a given p, the accuracy decreases as we move towards the border of the sensor in the horizontal direction. In summary, the accuracy of estimation is a function of sensor characteristic and the ratio of the sensor view angle to the number of pixels in the image.

3.2. Quantitative Results and Discussions

The sensitivity analysis described in the previous section was applied to three different sensors (Table 3.2) at six different positions (Table 3.3). Sensitivities $S_{X_c}^X$, $S_{Y_c}^Y$, and $S_{Z_c}^X$ at the aim point (i.e., p=0, q=0) for various sensor positions (Table 3.3) are plotted in Figs. 3.5, 3.6 and 3.7 respectively. In all plots sensitivity to range $S_{X_c}^X$ was scaled down by a factor of 10. Note that $S_{Z_c}^Y$ is larger than $S_{Z_c}^X$ at (0, 0) and hence a feature would move to the next horizontal pixel before it moves to the next vertical pixel. Thus only $S_{Z_c}^X$ is important.

As expected, the sensitivity is the best for the sensor with the highest pixel resolution. Sensitivity also improves as the sensor is moved closer to the ground. It becomes poor for the features that are located at the far end of the vertical axis (top of the sensor), i.e., for the objects that are located at the far end of the runway. Thus, as expected, the position

and velocity of the aircraft can be computed to a better accuracy by knowing the position of stationary objects on the ground that are closer to the aircraft.

Table 3.2

Sensor Characteristics		
Sensor type	Pixel (H×V)	Field of view (H× V) deg.
HDTV	1920×1035	30×24
FLIR	512×512	28×21
PMMW	80×64	27×22

Table 3.3

Sensor Positions		
(pitch = −3.0°, Roll = 0°, Cross range = 0 ft.)		
Location	Range in ft.	Altitude in ft
Threshold	0.0	50.0
CAT II-DH	908.1	100.0
CAT I-DH	2816.2	200.0
Middle Marker	4500.0	288.2
1000' Altitude	18081.1	1000.0
Outer Marker	29040.0	1574.3

The above results indicate that the accuracy of camera state estimation would be no better than the GPS data unless a high resolution sensor is employed. Note that these results do not consider potential improvements that can be obtained by motion stereo techniques using a large number of image frames. We are presently investigating the possibility of improving the accuracy of the computed sensor positional parameters by extending our analysis using this method.

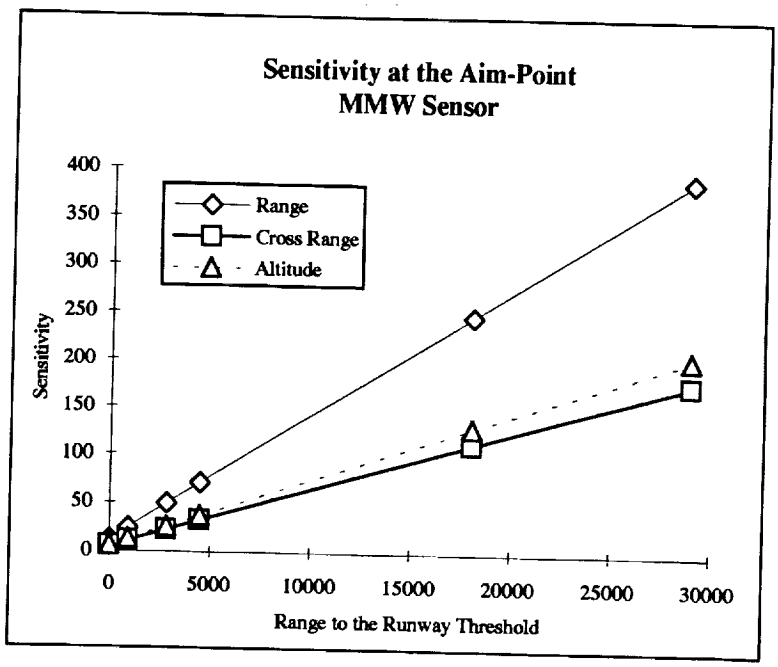


Fig. 3.5

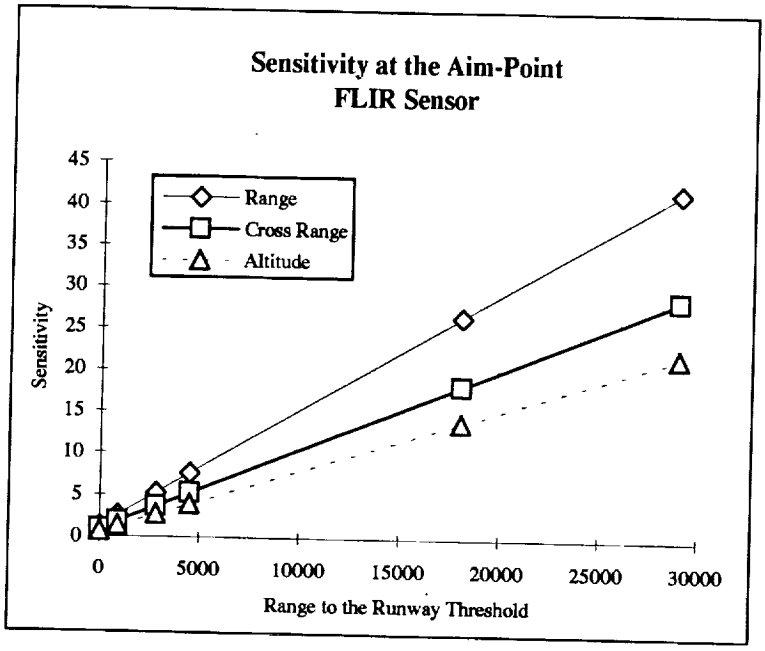

Sensitivity at the Aim-Point
FLIR Sensor
Range
Cross Range
Altitude
Sensitivity
45
40
35
30
25
20
15
10
5
0
0
5000
10000
15000
20000
25000
30000
Range to the Runway Threshold

Fig. 3.6

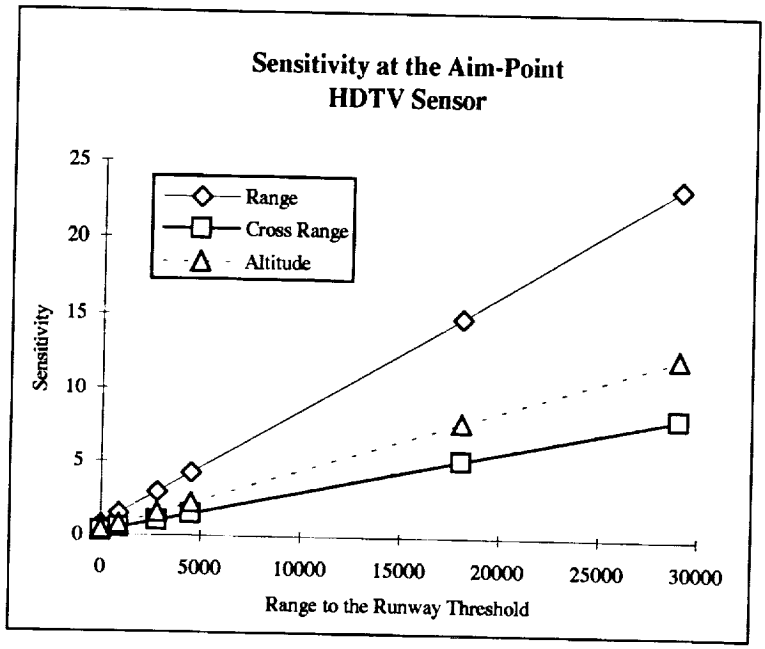

Sensitivity at the Aim-Point
HDTV Sensor
Range
Cross Range
Altitude
Sensitivity
25
20
15
10
5
0
0
5000
10000
15000
20000
25000
30000
Range to the Runway Threshold

Fig. 3.7

4. Model Transformation

As noted earlier, the PMMW images are low contrast-low resolution images. Simple edge detection techniques on these images generate many noisy edge pixels in addition to those belonging to the true edges such as runways, sky etc. This problem is alleviated by defining regions of interest on the ground plane for each feature in the model and to perform 3D to 2D transformation. It also defines a region in the image plane where the horizon line should occur.

4.1. Defining Regions of Interest for Runway Edges

The error in the expected location of a feature and its actual position in the image depends on several factors, most notably the accuracy of the camera position parameters used by the *model transformation* module. Furthermore, it is evident from our earlier analysis (Fig. 3.4) that the ground area covered by a pixel is a function of the position of the pixel in the image. Thus it is not reasonable to define the search space for each feature as a fixed number of pixels centered around the expected location in the image plane. Hence we define the region of interest in the 3D space and then apply transformation to get the corresponding region of interest in the image. The extent of the search space in the 3D space is determined by the estimated error in camera positional parameters (which are based on GPS and on-board instrument data).

The geometric model of the airport contains a sequence of 3D coordinates of the vertices of the runway/taxiways, which forms a polygon with n vertices:

$$runway = \{P_i\},\ i = 1, 2, ..., n,$$

where $P_i = (X_i, Y_i, Z_i)^T$ is one of the vertices of the polygon. Note that $Z_i \cong 0$. P_iP_{i+1} specifies an edge of the polygon. The region of interest is defined as a rectangle on the ground which encloses the edge. Therefore, each edge P_iP_{i+1} of the polygon is associated with the region of interest defined by four points $b_i = (X_j, Y_j, Z_j)$, $j = 1, ..., 4$, and $Z_j = Z_i$.

The width of the region of interest is defined as a function of the width of the runway/taxiways, w, accuracy of the GPS data, g ($0 \le g \le 1$), and the accuracy of the on-board instrument, d ($0 \le d \le 1$). Note that g and d are determined by the specification and characteristics of these instruments. This relationship is given by

$$width(w, g, d) = \frac{0.2w}{gd}. \qquad (4.1)$$

Note that the minimum width is $0.2w$ when $g=d=1$, which corresponds to ±10% potential displacement of runway edge feature. To limit the search area from being a large fraction of the runway width we limit the search width to $0.4w$ even if $gd<0.5$.

After defining the region of interest for each edge, 3D to 2D coordinate transformation is performed using the following homogeneous equation (Smith 1990):

$$\begin{bmatrix} \lambda p \\ \lambda q \\ \lambda r \\ \lambda \end{bmatrix} = PRT \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{4.2}$$

where

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{f} & 0 & 0 & 1 \end{bmatrix}, \tag{4.3}$$

$$R = \begin{bmatrix} -\cos\psi\cos\theta & -\sin\psi\cos\theta & -\sin\theta & 0 \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\sin\phi & -\cos\theta\sin\phi & 0 \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi & -\cos\theta\cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.4}$$

and

$$T = \begin{bmatrix} 1 & 0 & 0 & -X_c \\ 0 & 1 & 0 & -Y_c \\ 0 & 0 & 1 & -Z_c \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

are the perspective projection, rotation and translation transformation matrices, respectively, and f is the focal length. After perspective projection, we need to consider the following special cases:

A. the region of interest degenerates to a line in the image plane because the region is too far from the camera,

B. the region of interest in the image plane becomes very large because the edge is very close to the camera.

For case A, a minimum width in the image plane is assigned in order to provide some search space for the feature detector. For case B, a maximum width in image space is defined to further restrict the region. In our experiment, for the aforementioned extreme cases, the minimum and maximum width of a region of interest are set to be 10 and 20 pixels, respectively.

4.2. Defining Search Space for Horizon Line

When the vertical angular field of view is larger than 2θ, then a horizon line appears in the image (Fig. 4.1). The horizon is an important clue in estimating the camera orientation since it gives the roll angle information directly. Search space in the image plane is defined to locate this line.

Without loss of generality, consider the situation when the aircraft is heading towards the X axis of the world coordinate system. Assume the camera is located at point D (see Fig. 4.1) with pitch angle θ, and zero yaw and roll angles. Point A and B are on the top and bottom edge of the image, respectively. The horizon will then appear horizontally in

the image plane as shown. The distance between this line and the center line of the image is given by $\overline{HC} = f \tan\theta$. Since in the above analysis roll angle has been assumed to be zero, the horizon appears parallel to the horizontal axis of the image plane. For any non zero roll angle, a simple roll transformation on this line will give the horizon in the image. The associated region of interest is defined to be 10 pixels centered around the expected horizontal line in the image.

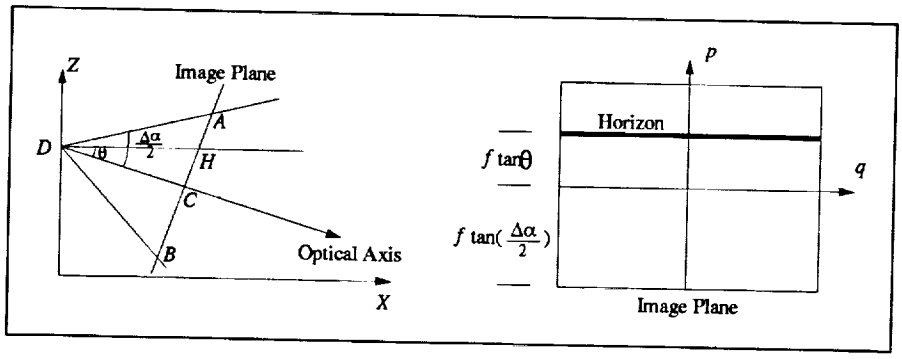


Fig. 4.1: Horizon line in the image.

It is possible for the projection of the region of interest onto the image plane to be partially outside the image boundary. In such cases, wee need to clip these regions so that the search space always remains within the confines of the image. This is done using the *polygon clip and fill* algorithm (Foley et. al., 1990). The regions of interest for both the runway and the horizon of the image sequence used in these experiment are shown in Fig. 4.2.

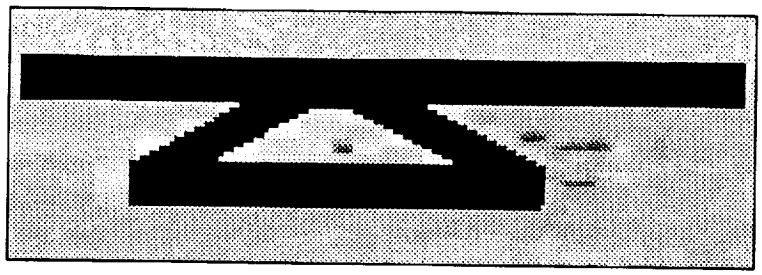


Fig. 4.2: Regions of interest.

5. Runway Localization and Object Detection

5.1. Runway Localization

In this part, we search for the expected features within the region of interest defined by previous module. This will significantly reduce the search time and also avoid spurious responses which are likely in such a low resolution input image. An accurate localization of the feature is necessary for estimation of motion parameters and camera pose.

A Sobel edge detector is applied to the sensor image. We then select one of the four scanning directions (–45°, 0°, 45°, 90°) which is approximately orthogonal to the direction of the expected edge. Along each scan line we locate pixels with greatest edge strength. As the runway edge is supposed to be a straight line we fit a best line to these pixels. We also associate a measure of confidence for these detected edges based on the number of edge pixels detected along the line.

5.2. Object Detection

In this part, the region inside and outside the runway/taxiways are separately checked for the existence of any stationary or moving objects. The image has three homogeneous regions, namely the sky, the runway/taxiways and the region outside the runway/taxiways. Any objects on or outside the runway/taxiways are expected to have some deviation in graylevel from their respective homogeneous background. Hence, we use histogram-based thresholding for object detection. The thresholds which determine this deviation are set to be different for different regions.

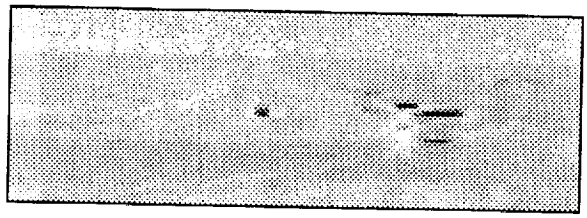
We generate a mask image which represents three homogeneous regions. Using this mask image, we generate the histogram and compute its standard deviation for each region separately (except for the sky region). The threshold value is determined as a function of the mean and the standard deviation, and any area which has graylevel lower than the threshold is considered as object regions. An object is assumed to have a reasonable size. This size restriction on the object can be used to ignore spurious responses resulting from the thresholding. Each object is then labeled based on 4-connectivity.

6. Experimental Results

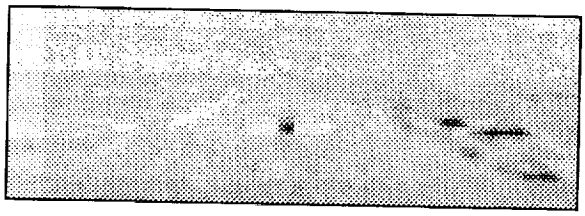
We have tested our algorithm on a test image provided by the TRW. This image was obtained using a single pixel camera located at a fixed point in space (a camera with an array of pixels is under development). The camera was mechanically scanned to obtain a 50×150 pixel image. This is the image shown in Fig. 1.1. We were also provided with the model of the runway giving the 3D world coordinates of the runway corners, locations of the buildings etc. Using these data and the single image, we created a sequence of 30 frames to simulate the images from a moving camera. Frames 1 (original), 8, 16, and 24 from this sequence are shown in Fig. 6.1(a). Edge enhanced images corresponding to these frames are shown in Fig. 6.1(b). The regions of interest defined on these frames are shown in Fig. 6.1(c). Delineated features superimposed on the images are shown in Fig. 6.1(d). Although all the edges are detected accurately in the example, it is likely that one or more edges of a polygon are not detected. To handle such situations we associate a degree of importance for each edge. For example, runway edges which are closer to the camera must be detected in the image whereas those corresponding to the far end of the runway are usually very short and may or may not be detected. And overall confidence measure is associated with each detected region.

Objects detected on the runway in Frame 1 and those outside the runway are shown in Fig. 6.2. Warning signals are generated for each object on or near runway. Algorithms to

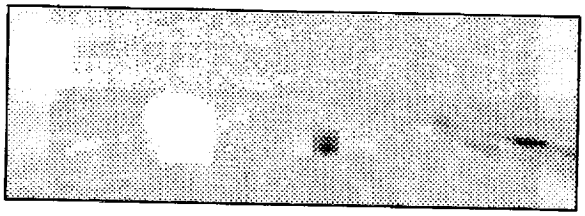
track these in successive frames and estimate camera state using motion stereo are under development.



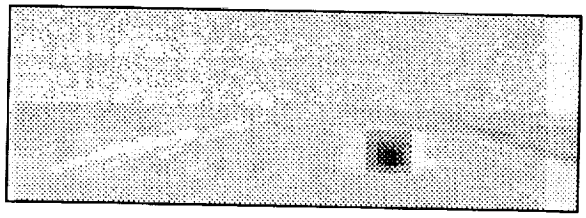
Frame 1



Frame 8



Frame 16



Frame 24

Fig. 6.1(a) The input images

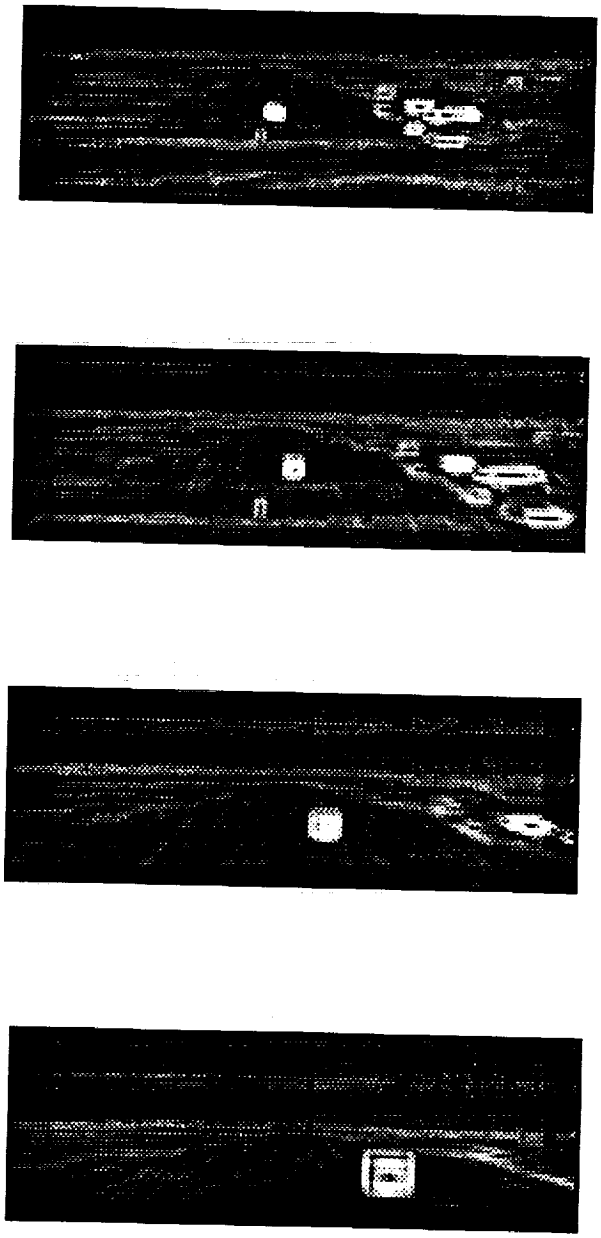


Fig. 6.1(b) The edge images

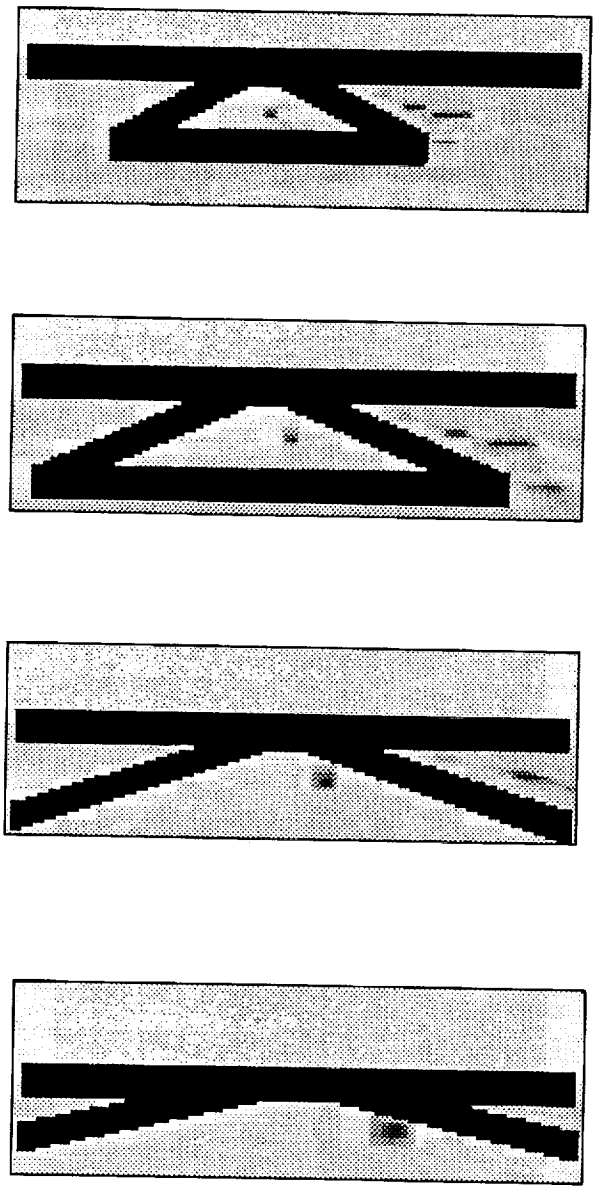


Fig. 6.1(c) Regions of interest

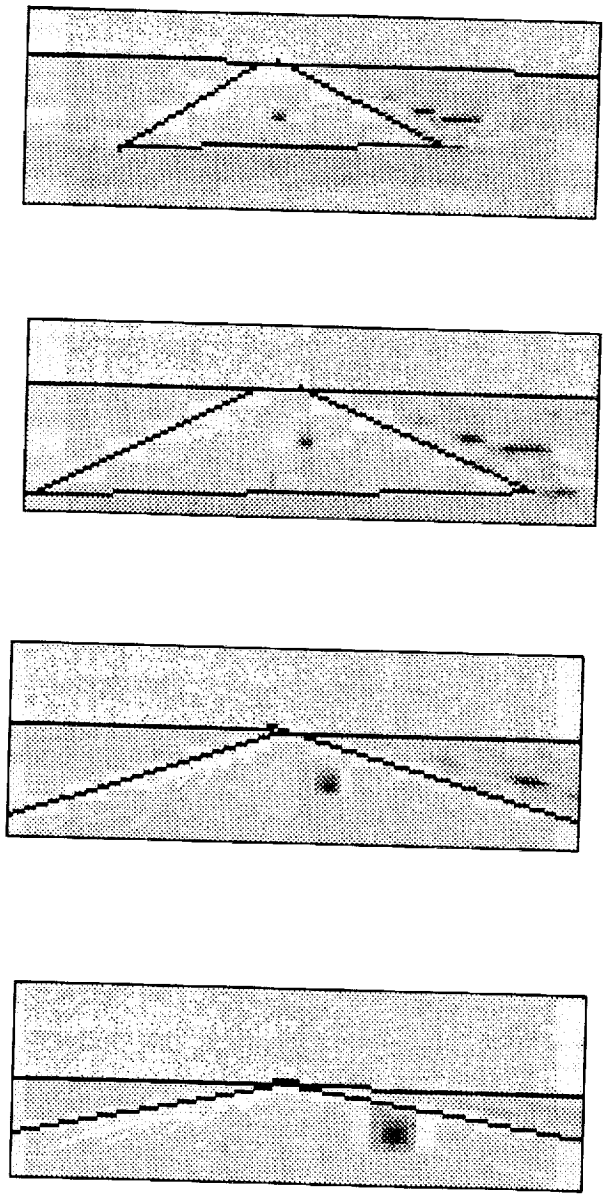


Fig. 6.1(d) Detected features superimposed on the original images

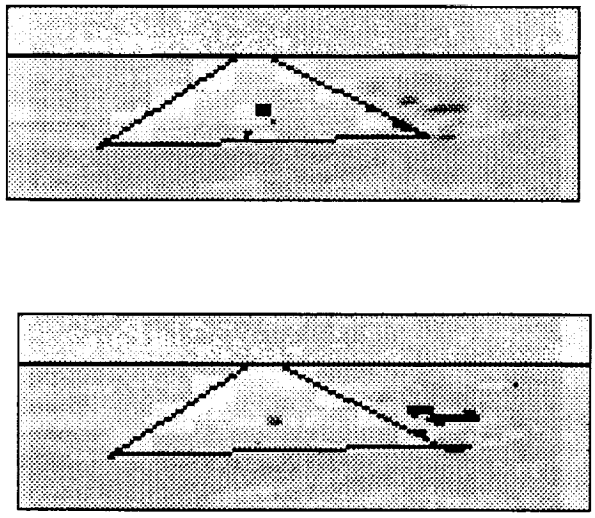


Fig. 6.2: Detected objects inside and outside the runway.

7. Future Work and Conclusions

In this part of the report, we have described a vision-based system to assist pilots during landing under restricted visibility conditions. The images obtained by a passive sensor is processed to detect major regions such as runways and objects inside and outside these regions. The image resolution is very poor, however, additional information in the form of airport geometric model, and camera position parameters are available to guide the segmentation algorithms. Objects are detected in each of these regions using thresholds computed separately for each region. Our results show that the model-based feature detection approach is quite accurate and the homogeneity assumption on regions

for object detection is reasonable. The success of this model-based approach clearly depends upon the accuracy of the camera position parameters used to define search regions in the image. One of the methods for updating camera position information is triangulation using known objects. We have derived the accuracy of such an update as a function of camera characteristics and image parameters.

At this stage, our system is able to detect the runway/taxiways and the objects inside and outside the runway/taxiways in each frame and to report their positions in the image. Since we have a moving camera, moving object situation, even the stationary objects appear to be moving in the image. Work is in progress to estimate the egomotion of the camera, to distinguish moving objects from stationary ones and to estimate the velocities of the moving objects. There is also potential to obtain more accurate camera state estimation using motion stereo from image sequences compared to using GPS data alone.

REFERENCES

Dickmanns, E.D., "Computer Vision for Flight Vehicles," *Zeitschrift für Flugwissenschaften und Wetraumforsch (Journal of Flight Science and Space Research)*, vol. 12, pp. 71–79, 1988.

Foley, J.D., A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics—Principles and Practice*, Addison-Wesley Publishing Co., 2nd ed., 1990.

Hatfield, J.J. and R.V. Parrish, "Advanced Cockpit Technology for Future Civil Transport Aircraft," *Proc. 11th Annual IEEE/AESS Dayton Chapter Symposium*, 1990.

Smith, P.N. "NASA Image Data Base User's Guide," NASA Ames Research Center, Moffett Field, CA., Version 1.0, 1990.

Smith, P.N., B. Sridhar, and B. Hussien, "Vision-Based Range Estimation Using Helicopter Flight Data," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 202–208, 1992.

Sull, S. and N. Ahuja, "Integrated 3D Recovery and visualization of Flight Image Sequences," *Proceedings of the Image Understanding Workshop*, pp. 473, 1992.

Young, S.K., R.A. Davidheiser, B. Hauss, P.S.C. Lee, M. Mussetto, M.M. Shoucri, and L. Yujiri, "Passive Millimeter-Wave Imaging," *TRW Space & Defense—Quest*, Vol. 13, No. 2, pp. 3–20, 1990/1991.

Part II

Runway Recognition and Position Estimation for Stationary Objects

Runway Recognition and Position Estimation for Stationary Objects

Abstract

A computer vision system to assist pilots during low-altitude flight and landing maneuvers has been developed in this research study. During this critical section of flight, a system which can recognize the runway in the image and detect various objects on the ground will be very useful to enhance the safety of navigation. Such tasks can generally be automated by computer vision-based methods, which provide the ability for object recognition and obstacle detection. In this work several algorithms have been developed to accomplish important tasks such as runway recognition, obstacle detection etc. These algorithms were tested on image sequences captured by a single camera mounted on an aircraft or a rotorcraft.

This research makes use of a priori knowledge about the runway geometric model and the aircraft/rotorcraft motion information to improve the performance of dynamic scene analysis. First, The system recognizes the runway in the images by using the runway model and the camera position information. An outline of the detected runway can thus be projected onto the image, which serves as a verification that the aircraft is heading in the correct direction. Creating such graphics also renders a better visualization for the pilots. Second, stationry objects in the image are detected and thier 3D positions are estimated. By knowing the static environment, the system is able not only to verify expected objects in the scene, such as the runway/taxiway, equipment, and the buildings, but also to find unexpected objects near the flightpath. The aircraft/rotorcraft can use such information to modify the nominal trajectory and thus avoid possible collision.

1. Introduction

Because of the heavy workload demands that are imposed upon pilots and crew during low-altitude flight, there is a significant need for an automatic obstacle detection system onboard an aircraft or a rotorcraft. Such a system can relieve the pilots from tiring, monotonous flight control tasks so that they can concentrate more on flight planning. In addition, despite the enforced control of airport ground personnel and vehicles, runway incursion is still a serious problem which jeopardizes the safety of aircraft landing. A computer vision system would thus provide a general approach to assist the pilots in recognizing objects and detecting obstacles. There is also the same need for a detection system for high performance unmanned vehicles. For example, a spacecraft exploring a remote planet needs to have a vision system in order to land on the rough terrain of the planet. Helicopters carrying cameras would be useful for inspecting contaminated or dangerous areas. Although such vision-based navigation is very important, there has been very little work done to develop such a system, mainly because of the difficulty in obtaining sufficient test data and the immaturity of vision algorithms. Since it is obvious that vision-based autonomous navigation will become more and more important in the future, we feel it is necessary to perform research on this topic and develop techniques which would benefit future applications. In this research, the vision system is designed mainly for low-altitude flying aircraft or rotorcraft. Therefore, it is, in

fact, a general case of other types of navigation, such as Autonomous Land Vehicles (ALV) and robotics.

Obstacle detection for a moving observer has been an active research topic for years. A critical issue is to estimate the range or the position of an obstacle relative to the moving observer. This technique has been extensively studied in ALV and other ground-based robotic applications, where road following is the key guidance function. However, in the case of low-altitude flying rotorcraft, the ability to maneuver around obstacles is the challenge for guidance systems. The rotorcraft flight at low-altitude has several distinct characteristics as opposed to the ALV case: (1) due to the curvilinear motion of the rotorcraft, a large class of passive ranging algorithms designed for linear motion are not directly applicable; (2) the outdoor scene is generally not known in advance; hence model-based algorithms, such as the road following method, will not be possible; (3) the sensor motion parameters are available from the Inertial Navigation System (INS). Also, this research differs from general motion analysis methods in that the rotorcraft's motion parameters are not estimated. They are assumed to be computed using an onboard navigation system. The main idea is to demonstrate that accurate position estimates and moving object tracking are achievable with the knowledge of camera motion parameters.

1.1. Objectives

This research is concerned with the problem of estimating and visualizing the structure of a stationary scene and the velocities of the independently moving objects as seen by a moving observer. There are four main objectives: first, to recognize the runway in the image from a priori knowledge of the runway geometric model and camera motion; second, to detect moving objects in the images; third, to estimate the velocities of the moving objects, and finally, to estimate the 3D positions of the stationary objects. The key to meeting these objectives is the information of the camera motion. As modern aircraft are usually equipped with onboard inertial navigation systems, the aircraft/camera state (i.e., position and orientation) is continuously available. With the help of the camera state information, the runway in the image can be reliably recognized and the quality of the motion analysis will be significantly improved.

Among the four objectives, two of which are addressed in this part of the report, each addresses important and difficult issues in the current research trends as described in the following. Locating the runway presents a problem of model-based recognition. Hypothesis of the runway is first generated on the image plane by defining the regions of interest on the image plane. And then the existence of the runway is verified by feature extraction and model matching. After this stage, the image can be roughly divided into several regions, such as the sky, the runway/taxiways, and other areas. Different image analysis techniques can thus be performed on each area. This heterogeneous processing is

important for a real-time system in that computationally expensive processing on the *whole* image can be avoided.

As is well known, moving object detection using visual information alone is quite difficult, particularly when the observer is also moving. The reason is that different motion scenarios among the environment, objects, and the camera may result in a very similar image sequence. This ambiguity can be resolved by the knowledge of the camera motion information. For stationary object position estimation, also known as *structure from motion* techniques, most algorithms suffer from the noise caused by image digitization and the algorithms of feature extraction and feature matching. A method is presented to improve the performance of feature matching and hence to obtain accurate position estimates.

This research presents rather challenging problems in computer vision. Related research topics include image processing, model-based recognition, estimation theory, and dynamic scene analysis. The algorithms developed will be tested on three sets of image sequences. One of them, namely the *runway sequence*, was obtained by digitizing a video tape provided by NASA Langley Research Center, Hampton, Virginia. The tape was recorded during the landing of an aircraft. The other two, namely the *line* and the *arc sequences* of the helicopter images, were provided by NASA Ames Research Center, Moffett Field, California. These two sequences were captured from a helicopter

conducting low-altitude flight. The nature of these sequences will be further explained later.

1.2. Organization

In this section a brief introduction to the problem addressed in this resarch has been provided, followed by the main objectives. Section 2 gives an overview of the vision analysis system. In that section, the coordinate systems of the imaging system are first defined. This includes the relationships between the coordinate systems and the camera motion parameters. A system block diagram is then shown which describes the functions of various modules of the vision system and the interaction among them. And finally, the first stage, i.e., the calibration stage of the system is described.

A model-based method to recognize the runway as the aircraft approaches the airport is described in section 3. With the availability of the runway model and the camera pose, the runway in the image can be quickly and reliably recognized. The success of quick recognition relies on the fact that blind tree search can be totally avoided if the approximate camera pose is known. The quality of recognition, however, depends upon the accuracy of camera pose information and the line detection algorithm.

The method to estimate the positions of stationary objects is described in section 3. This technique belongs to the class of *structure from motion* methods. Such methods generally suffer from noise caused by image digitization, feature detection algorithms, and

feature tracking/matching algorithms. A technique utilizing the epipolar lines is presented to significantly improve the performance of feature tracking, and thus to obtain accurate position estimates. Section 4 gives the summary and conclusions of this part of theresearch work. Future research directions are also addressed.

2. system overview

Piloting an aircraft is a rather demanding task and engineers have long been considering ways to alleviate some of the pilot's work by using partial automation. Automatic control could free the pilot from tiring, monotonous control tasks. Instead, the pilot could concentrate on mission planning and supervision, adjusting *autopilot* parameters now and then while taking over control only when more complicated navigation is needed. In addition, during the critical period of take-off and landing there is a need to have a system which can assist the pilot in detecting the runway and potential hazards more accurately and reliably. Although remote piloting can satisfy this need, it requires a steady high bandwidth data link which is extremely difficult to maintain, especially in a hostile environment. Also, remote piloting is not able to alert the pilot of any unexpected obstacles that enter the runway area. The key to the success of human piloting relies on the human vision system, which provides the pilot with the most useful information, especially for interactions with the environment close to the ground surface. A computer vision system could make the informaiton more complete and accurate, however, and therefore make flight safer. The imaging system is depicted in Fig. 2.1,

where the helicopter is equipped with the Inertial Navigation System which, in cooperation with the ground equipment, constantly provides the position and orientation of the helicopter. A camera is mounted in front of the helicopter to capture the images. Also shown in the figure are the relationships among different coordinate systems which are described in the following sections.

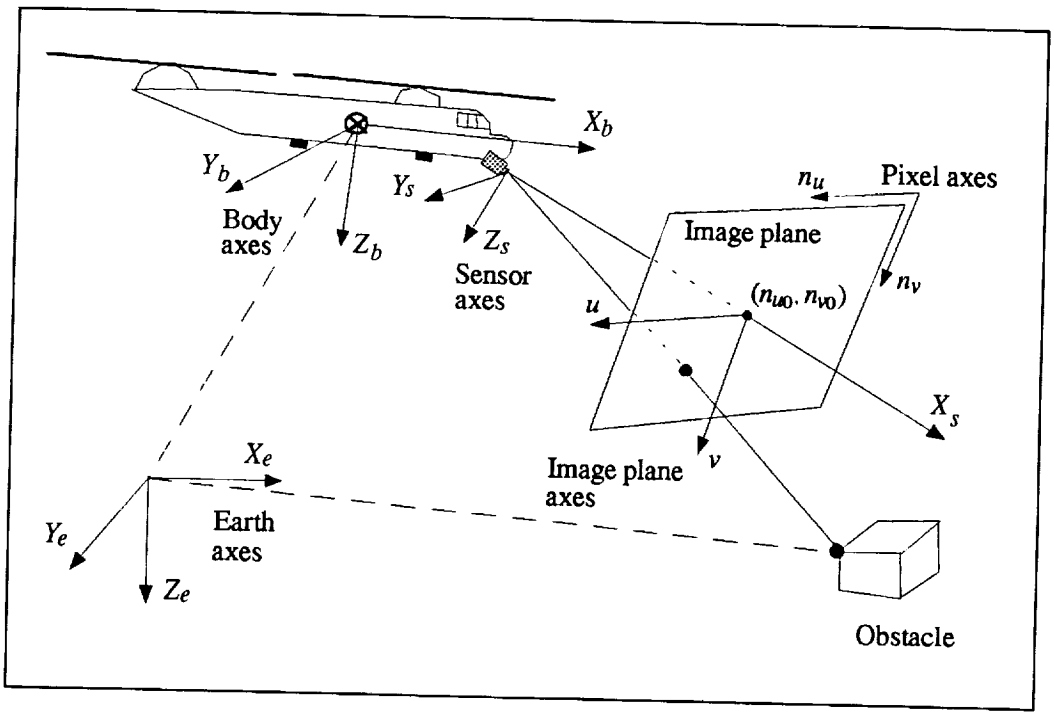


Fig. 2.1: Geometry for the imaging system.

2.1. Coordinate Systems

As shown in Fig. 2.1, the coordinate systems of the vision system include the Earth frame, the helicopter body frame, the sensor frame, the image plane axes, and the pixel axes (Smith, 1990):

1. Earth (world) frame — The Earth frame is rigidly affixed to the Earth with three axes, X_e, Y_e, and Z_e. The origin of the Earth frame is an arbitrarily selected point on a runway at the test flight facility.
2. Helicopter body frame — the helicopter body axes frame (or body frame) is assumed to be fixed relative to the helicopter with the X_b axis pointing forward out the helicopter nose, the Y_b axis pointing out the right hand side of the helicopter, and the Z_b axis pointing downward relative to the helicopter's geometry. The origin of the body frame is the helicopters nominal center of gravity.
3. Sensor frame — The sensor frame is rigidly attached to the camera and originates at the lens focal point. The Y_S and Z_S axes are parallel to the image plane axes u and v, respectively. The X_S axis points along the optical axis. The camera is rigidly mounted to the helicopter.
4. Image Plane Axes — The image plane axes are oriented along the rows and columns of the sensor array. The u axis points to the right along rows and the v axis points downward along the columns.

5. Pixel axes — The pixel axes, n_u and n_v, are attached to the camera's image plane and point along the rows and columns of the sensor array as do the image plane axes; however, the pixel axes originate at the upper left-hand corner of the sensor array rather than at the image center. The upper left-hand pixel has coordinate (0, 0).

2.1.1. Coordinate Transformation

The coordinate transformation from one frame to another can be expressed by

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R(\psi,\theta,\phi)\begin{bmatrix} X - X_t \\ Y - Y_t \\ Z - Z_t \end{bmatrix} \tag{2.1}$$

where $[X_t, Y_t, Z_t]^{\mathrm{T}}$ is the translation matrix and R is the rotation matrix. The rotation matrix is defined in terms of Euler angles. The Euler angles are yaw angle ψ, pitch angle θ, and roll angle ϕ. The rotation sequence is R_ψ about Z, R_θ about Y, and R_ϕ about X axis, where all rotations are positive in the right-hand sense. The rotation matrix resulting from this sequence of rotation is given below:

$$R(\psi,\theta,\phi) = R_\phi R_\theta R\psi = \begin{bmatrix} c\psi c\theta & s\psi c\theta & -s\theta \\ c\psi s\theta s\phi - s\psi c\phi & s\psi s\theta s\phi + c\psi c\phi & c\theta s\phi \\ c\psi s\theta c\phi + s\psi s\phi & s\psi s\theta c\phi - c\psi s\phi & c\theta c\phi \end{bmatrix} \tag{2.2}$$

where c is cosine and s is sine functions, respectively.

The rotation matrix premultiplies a column vector to express that vector in another coordinate frame. The rotation matrices from the Earth frame to the body frame, from the body frame to the sensor frame, and from the Earth frame to the sensor frame are given by the following equations:

$$\begin{aligned} R_{be} &= R(\psi_{be}, \theta_{be}, \phi_{be}) \\ R_{sb} &= R(\psi_{sb}, \theta_{sb}, \phi_{sb}) \\ R_{se} &= R_{sb} R_{be} = R(\psi_{se}, \theta_{se}, \phi_{se}) \end{aligned} \tag{2.3}$$

where R_{be} is the rotation matrix from the Earth frame to the body frame, ψ_{be} is the yaw angle from the Earth frame to the body frame, etc.

2.1.2. Perspective Projection Equations

The perspective projection equation which map points from the sensor axes system to a pixel location in the image array are

$$\begin{aligned} n_u &= n_{u0} + AR\, f_e (y_s / x_s) \\ n_v &= n_{v0} + f_e (z_s / x_s) \end{aligned} \tag{2.4}$$

where (x_s, y_s, z_s) is the location of a point in sensor axes and (n_u, n_v) is its projected location on the image plane, (n_{u0}, n_{v0}) is the image center, f_e is the effective focal length, and AR is the aspect ratio defined as $AR = \delta v/\delta u$. δu and δv are the horizontal and vertical pixel spacing, respectively.

2.2. System Description

Fig. 2.2 gives an overview of the vision system. The first module, *calibration*, performs the calibration task. This is done only on the runway sequence as the field of view of the camera used is very large, producing distorted images. For the helicopter sequences, the images are well calibrated. The calibration method will be described in the following section. After this stage, the corrected images are fed to other modules. The runway in the images are first detected by the *runway recognition* module. This module takes the runway geometric model and the camera state information as inputs to define the regions of interest on the image plane. The runway model contains the 3D position information of the runway features. After defining the regions of interest, an hypothesis of the runway can be made in the image and further verified by model-based object recognition. An outline of the runway can be superimposed on the cockpit screen which provides better visualization for the pilots. Detailed algorithms for runway recognition will be described in section 2.3.

The purpose of the *motion detection* module is to detect and segment independently moving objects in the image. An advantage of separating the moving objects is that different analysis techniques can be applied on different portions of the image. Computationally expensive algorithms applying on the whole image such as moving object tracking can be avoided. The module also takes the camera state information as inputs. With such information, optical flow due to independently moving objects can be identified

because such flow usually violates a certain constraint. A method proposed by Lucas and Kanade (1981) is first applied to compute the optical flow in the image, followed by *constraint ray filtering* (Nelson, 1991) to detect optical flow produced by moving objects. After this stage, moving objects in the image can be segmented from other stationary objects. The segmented image portions will be fed into the *tracking* module and the rest of the image will be the input of the *position estimation* module.

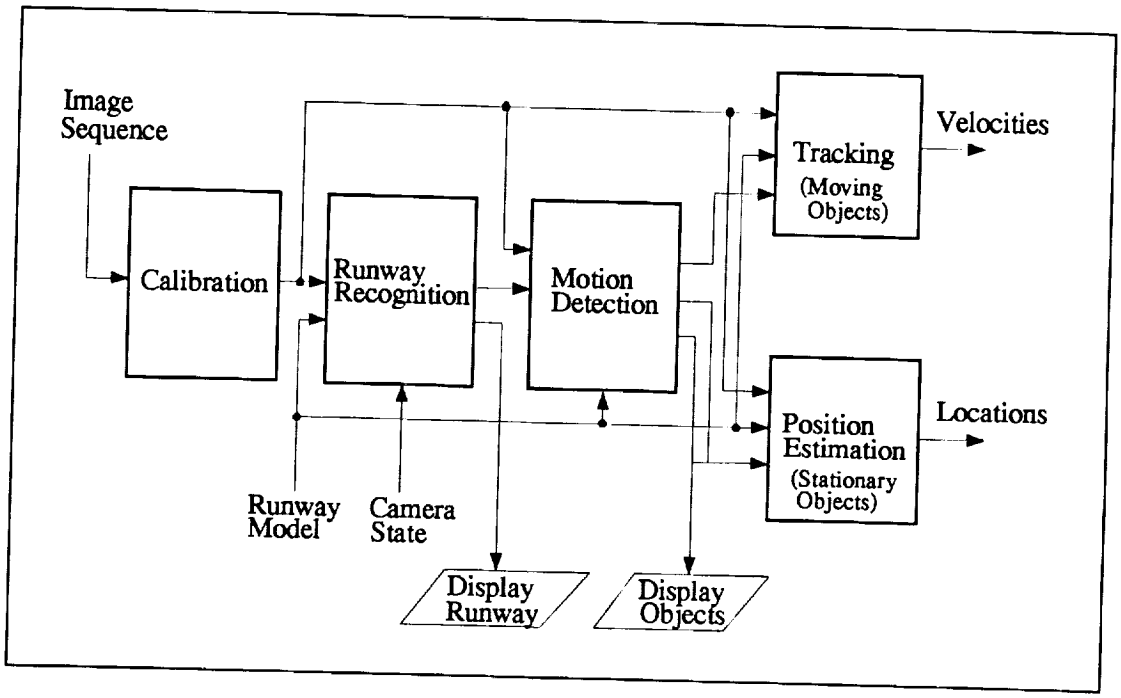


Fig. 2.2: System block diagram.

In the *tracking* module, moving objects in the scene are tracked in the image sequence. The tracking is done by matching the 2D *shape* of an object in the image. The *shape* of the object is created directly from the image; hence, no a priori information of the object is required. To estimate the object motion, the extended Kalman filtering technique is used. Using the Kalman filter, the estimation can be performed recursively and the state covariance matrix indicates the goodness of the estimates. The outputs of this module are the 3D positions and the velocities of the moving objects.

In the *position estimation* module, an incremental estimation method is used to estimate the 3D positions of the stationary objects. Since the motion of the camera is known, a pencil of epipolar planes and epipolar lines (Baker & Bolles, 1989) can be defined on the image plane. As is well known, the motion of image features, i.e., the optical flow, will follow the directions of the epipolar lines. This provides a strong constraint, namely the *epipolar constraint*, in feature tracking on the image plane. It will be shown in section 4 that such successful tracking makes the estimates of object positions very accurate. The outputs of this module are the 3D position estimates of stationary objects.

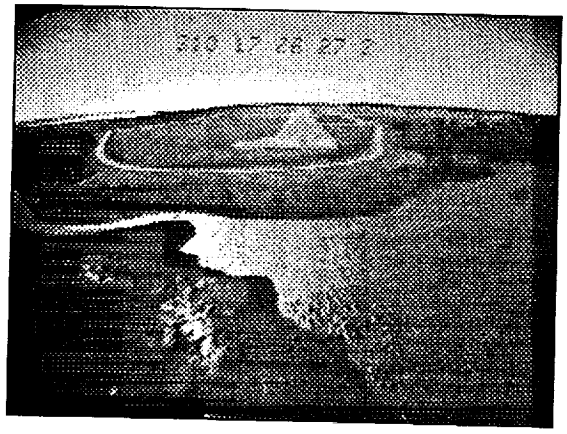
2.3. Image Sequences for Experiments

The algorithms developed in the research work were tested on three sequences of images. The first is the runway image sequence digitized from a video tape provided by NASA Langley Research Center, Hampton, Virginia. The other two, i.e., the *line* and the *arc* sequences of the helicopter images were provided by NASA Ames Research Center, Moffett Field, California. The image sequences are described as follows.

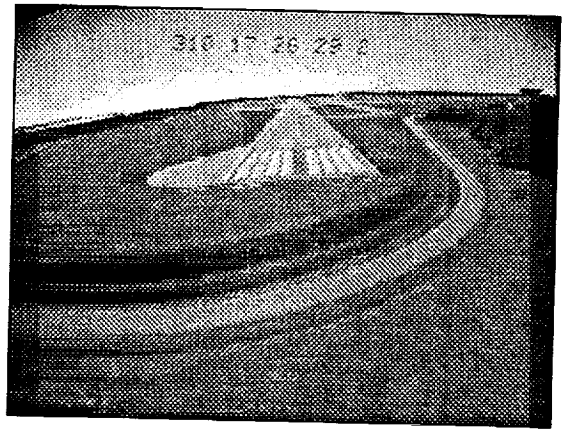
2.3.1. Runway Image Sequence and Calibration

The runway images are obtained by digitizing a video tape recorded when an aircraft approaches the runway for landing. The image resolution is 480x640 and the frame grabbing rate is 2 frames/sec. The information of the aircraft body positions and orientations are also provided, together with the camera attitude relative to the aircraft body. Fig. 2.3 shows a number of frames of the sequence. Section 3 provides a detailed description about the characteristics of the image sequence. Due to the wide field of view of the camera (81x64 degrees), the images exhibit considerable distortion called the *radial lens distortion*. This is obvious by noticing that the horizon line and the runway edges in the image are curved. Line detection in such images may produce noisy outputs and camera calibration is thus necessary. The calibration program developed by Sommer (1994) is used to correct the images. In the calibration procedures, only the internal

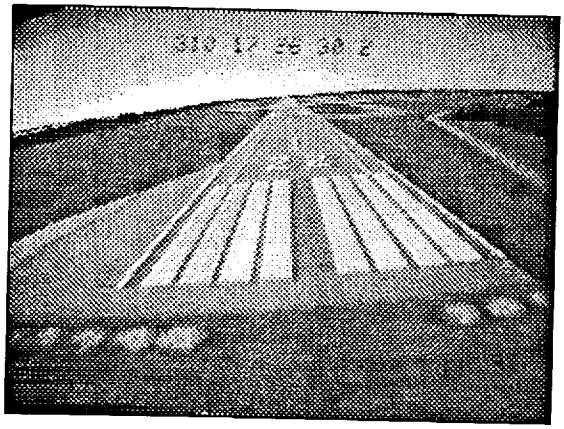
parameters such as the effective focal length and the distortion coefficient of the camera need to be estimated if the radial lens distortion is assumed.



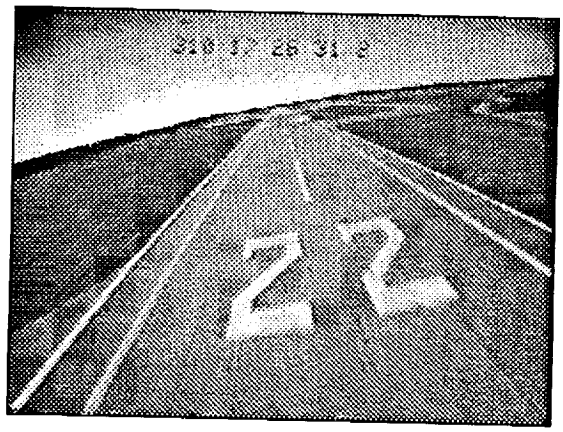
Frame 54



Frame 58



Frame 62



Frame 66

Fig. 2.3: The sequence of the runway images.

To estimate the internal parameters of the camera, a projection model has to be established. If a scene point P is known at location (X_c, Y_c, Z_c) relative to the camera coordinate system, the image plane is parallel to the camera's YZ plane, and the image axes y and z are parallel to camera axes Y and Z, respectively, then the perspective projection equations can be written as (see Fig. 2.4):

$$\begin{cases} y_u = Y_c \dfrac{f}{X_c} + y_0 \\ z_u = Z_c \dfrac{f}{X_c} + z_0 \end{cases} \tag{2.5}$$

where (y_u, z_u) is the undistorted image position, f is the focal length, and (y_0, z_0) is the image center. The radial lens distortion is modeled as

$$\begin{cases} y_d = y_u / \left(1 + kr^2\right) \\ z_d = z_u / \left(1 + kr^2\right) \end{cases} \tag{2.6}$$

where (y_d, z_d) is the distorted image position, k is the distortion coefficient, and $r^2 = y_u^2 + z_u^2$. In the runway images, a set of scene points $P_i = (X_{ci}, Y_{ci}, Z_{ci})$, $i = 1, \ldots, n$, is available from the specifications of the locations of the painted markings, and their corresponding image positions (y_{di}, z_{di}) can be measured manually. Thus a set of simultaneous nonlinear equations with unknowns f and k can be created. The equations are solved by using the Marquadt method (Press et al., 1992). After solving for f and k,

the image is corrected by the inverse function of Eq. (2.6). Fig. 2.5 shows the original image and the corrected image.

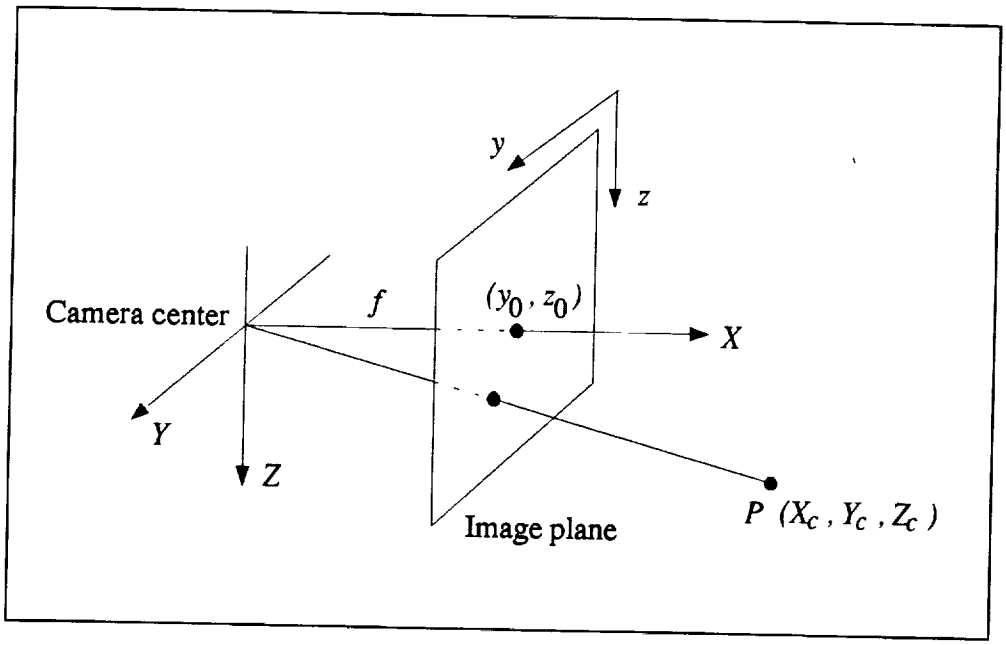


Fig. 2.4: The perspective projection.

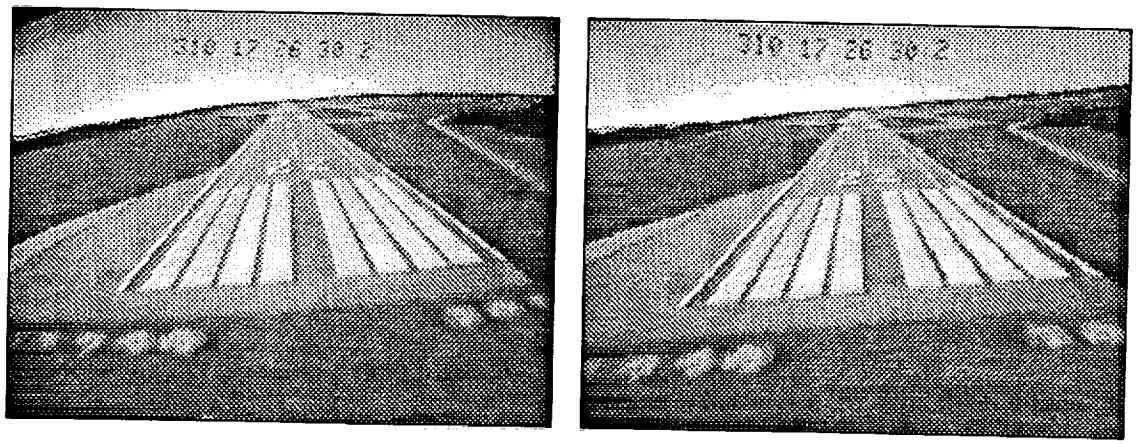


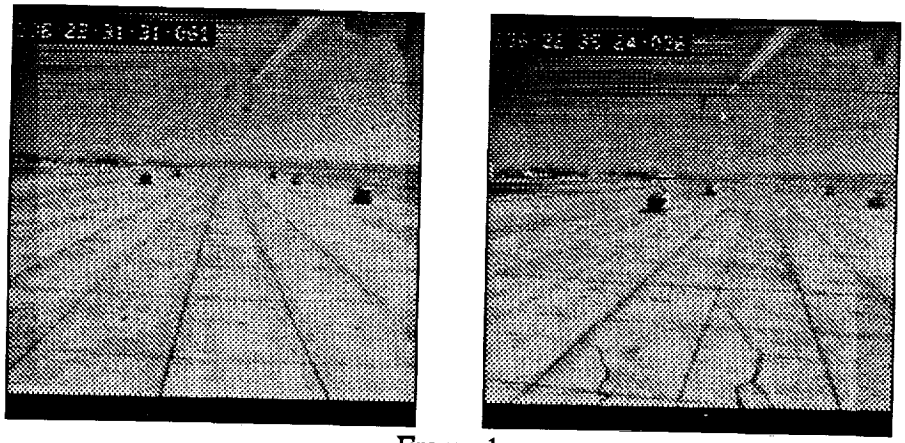
Fig. 2.5: The distorted image (left) and the corrected image (right).

2.3.2. Helicopter Image Sequences

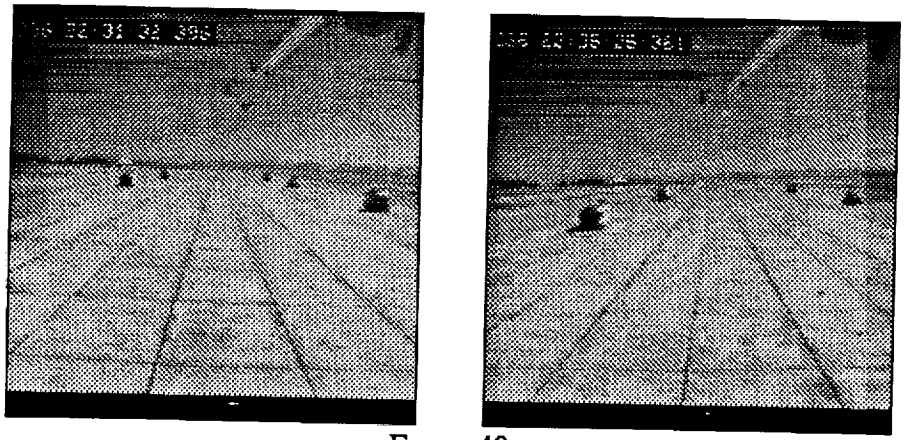
There are two sequences of helicopter image sequences, namely the *line* and the *arc* sequences, which were provided by NASA Ames Research Center, Moffett, California. Each sequence consists of 90 image frames with size 512×512 pixels and each frame contains a header information which records the helicopter body and camera positions and orientations, body and camera motion parameters, camera parameters, etc. Time stamps are projected directly on the image frames. Table 2.1 gives an example of the image header information. Fig. 2.6 shows several frames of the two sequences. For the *line* sequence, the helicopter's flightpath is approximately a straight line and there are five trucks in the scene during the whole sequence. For the *arc* sequence, the helicopter is making a turning flight and Truck 1 is not visible in all frames. The trucks are labeled in terms of their range (X) values. Thus, Truck 1 is the nearest and Truck 5 is the farthest relative to the camera. Ground truths for the locations of each trucks are also provided, as listed in Table 2.2.

Table 2.1: Sample Image Header Data.

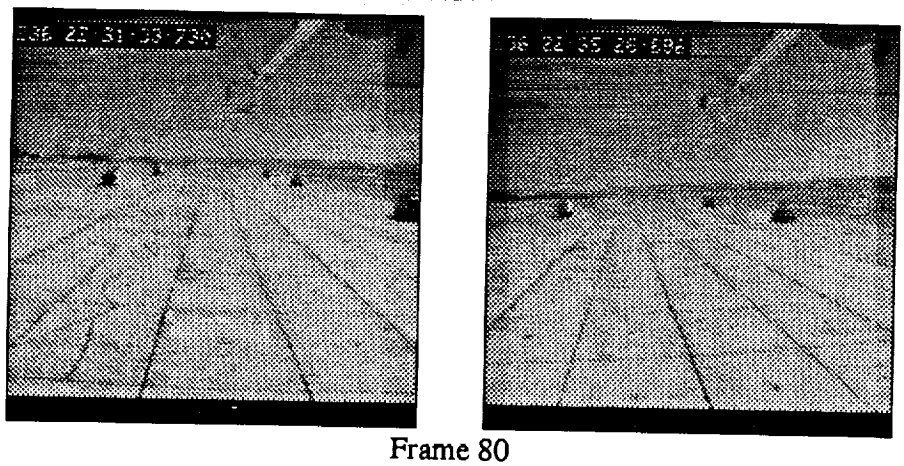
Measurement Name	Value	Accuracy	Units
SENSOR_POSITION_X_WORLD	734.218469		feet
SENSOR_POSITION_Y_WORLD	520.450862		feet
SENSOR_POSITION_Z_WORLD	-10.706197		feet
BODY_POSITION_X_WORLD	757.336792	2.0	feet
BODY_POSITION_Y_WORLD	517.029711	2.0	feet
BODY_POSITION_Z_WORLD	-16.165920	2.0	feet
SENSOR_VELOCITY_X_SENSOR	30.175145		feet/sec
SENSOR_VELOCITY_Y_SENSOR	0.186516		feet/sec
SENSOR_VELOCITY_Z_SENSOR	-1.921346		feet/sec
SENSOR_ANGULAR_RATE_X_SENSOR	0.023833		rad/sec
SENSOR_ANGULAR_RATE_Y_SENSOR	0.011290		rad/sec
SENSOR_ANGULAR_RATE_Z_SENSOR	0.012405		rad/sec
BODY_VELOCITY_X_BODY	30.052409	0.3	feet/sec
BODY_VELOCITY_Y_BODY	0.134803	0.3	feet/sec
BODY_VELOCITY_Z_BODY	2.572170	0.6	feet/sec
BODY_ANGULAR_RATE_X_BODY	0.021827	0.0045	rad/sec
BODY_ANGULAR_RATE_Y_BODY	0.011502	0.0045	rad/sec
BODY_ANGULAR_RATE_Z_BODY	0.015511	0.0025	rad/sec
SENSOR_POSITION_X_BODY	22.950401	0.042	feet
SENSOR_POSITION_Y_BODY	-1.043942	0.017	feet
SENSOR_POSITION_Z_BODY	6.939767	0.017	feet
ANGLE_PSI_WORLD_TO_BODY	3.034820	0.0123	radians
ANGLE_THETA_WORLD_TO_BODY	0.064561	0.0021	radians
ANGLE_PHI_WORLD_TO_BODY	-0.015253	0.0042	radians
ANGLE_PSI_BODY_TO_SENSOR	0.005538	0.0035	radians
ANGLE_THETA_BODY_TO_SENSOR	-0.139288	0.0035	radians
ANGLE_PHI_BODY_TO_SENSOR	-0.007376	0.0017	radians
ANGLE_PSI_WORLD_TO_SENSOR	3.042443		radians
ANGLE_THETA_WORLD_TO_SENSOR	-0.074628		radians
ANGLE_PHI_WORLD_TO_SENSOR	-0.022281		radians
ASPECT_RATIO	1.005400	0.001	non-dimensional
FOCAL_LENGTH	621.399231	2.6	pixels
U_CENTER	253.255096	2.4	pixels
V_CENTER	238.301407	1.6	pixels
STAMP_TIME	236:22:31:31.061		seconds
GLOBAL_TIME	81091.061000		seconds
DELTA_TIME	0.033333		seconds
FRAMEID	0		non-dimensional



Frame 1



Frame 40



Frame 80

Fig. 2.6: The line (left) and the arc (right) sequences of the helicopter images.

Table 2.2: The ground truths of the locations of the trucks in the scene.

Truck	X	Y	Z
1 (north east ground level)	479.3	470.6	4.9
1 (south east top corner)	461.5	472.3	−3.0
2 (north east ground level)	368.7	614.1	4.5
2 (south east top corner)	348.7	614.5	−3.1
3 (north east ground level)	231.2	490.2	3.9
3 (south east top corner)	209.0	491.9	−0.2
4 (north east ground level)	118.3	633.1	3.6
4 (south east top corner)	98.6	634.8	−7.4
5 (north east ground level)	−17.7	510.6	3.0
5 (south east top corner)	−37.6	511.5	−7.7

3. RUNWAY RECOGNITION

When a pilot is trying to land an aircraft, the first thing he/she needs to do is to recognize and localize the runway. From the view point of the vision system, the pilot has an image on his retina, which corresponds to the *sensor data*. He/she also has in mind what the runway should look like, which corresponds to the *object model*. The task of the vision system is then to find occurrences of the object in the sensor data. In order to recognize an occurrence of an object in a scene, we must have some notion of what the object looks like. This information is usually provided by a geometric model of the object stored in the computer. This object model usually is a set of *model features* which the object is composed of and a set of relationships or constraints among the features. Thus, the recognition process is essentially a matching problem where we seek to find the best correspondence between some set of sensor features and the same types of known model features. Hence, this type of recognition strategy is called *model-based recognition*. A wide range of applications are suitable for model-based recognition, such as object manipulation, navigation, and inspection. A more complete discussion of the strategies can be found in the review papers by Binford (1982), Besl and Jain (1985), Brady et al. (1989), and Chin and Dyer (1986).

For the application of runway recognition, the only object to be recognized is the runway and there should be only one instance of the runway in the image. This is different from most other object recognition systems, where usually a class of different objects is

stored in the database, and a number of instances of objects may exist in the image. In such a case, the recognition process has to not only find all occurrences of objects but also distinguish among different objects. In this chapter, several aspects of the subject of the *runway recognition* are addressed, including the following:

- what model features are used to represent the object?
- what image features are used to recognize the object?
- what methods are used to establish a correspondence between image features and model features of the object?
- how do we deduce the existence of an object from image features?

The answers to these questions will be provided in the following sections.

3.1. Runway Model and Line Detection

The regulations set by the Federal Aviation Administration specify various runway specifications (Advisory Circular, 1980) including the geometry, pavement, painted markings, etc. Fig. 3.1 shows the particular markings at the beginning of a runway, and Fig. 3.2 shows what the markings look like in the input image. These markings are painted white and appear prominently when seen from an approaching aircraft. Hence they are considered as important features for runway recognition. Among the features,

the eight short stripes at the beginning of the runway and the two side stripes are more important since they are more easily seen and are relatively easy to detect in the image. This can be seen from Fig. 3.1 where the edge image clearly shows the edges of these features. We thus define a number of edges of the markings as the model features to be matched in the image. A runway model is defined as a set of model features:

$$runway = \{M_i\},\ i = 1, .., m, \tag{3.1}$$

where $M_i = (M_i^1, M_i^2)$ is one of the edges of the marking with $M_i^1 = (X_i^1, Y_i^1, Z_i^1)$ and $M_i^2 = (X_i^2, Y_i^2, Z_i^2)$ specifying, respectively, the 3D position of the two end points of the edge and m is the number of total model features selected. All positions are relative to an arbitrary reference point in the airport. Fig. 3.4 shows the selected 36 features as they are more likely to appear in the image.

To perform runway recognition, three stages of processing are conducted, namely, edge detection, line detection, and model-based recognition. Canny edge detector (Canny, 1986) is first applied to the image. The result is shown in Fig. 3.3. To detect straight lines given the edge image, the software package *Object Recognition Toolkit* (*ORT*) developed by Etemadi et al. (Etemadi, 1993; Etemadi et al., 1993) at the University of Surrey, England, is used. This software takes an edge image as the input and reports the detected straight line segments, circular arcs, and various junction between lines, such as the T and Y junctions. In our application, we ae interested in straight line segments and

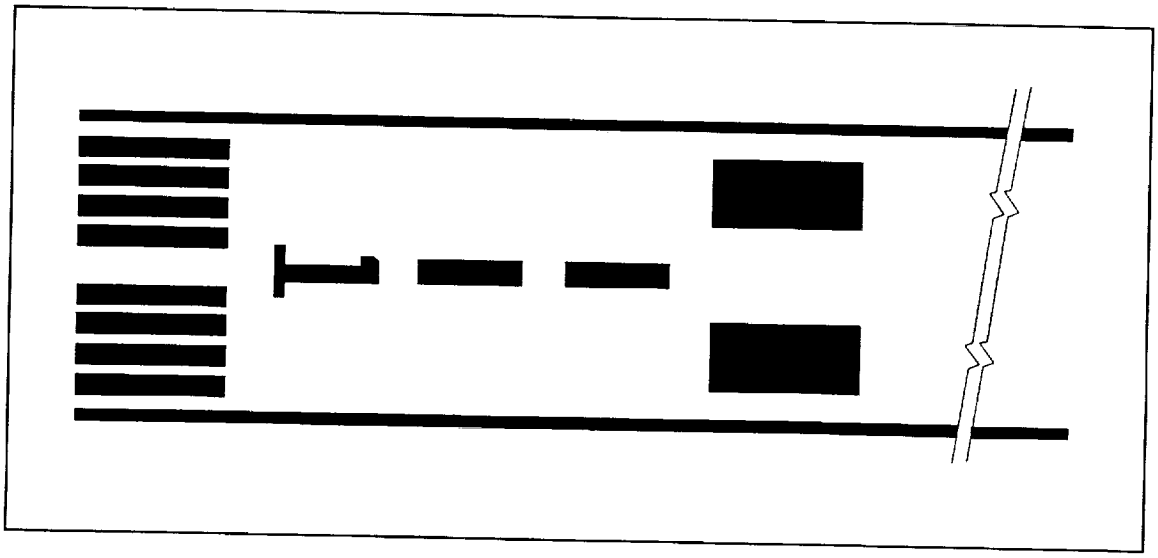


Fig. 3.1: The painted markings of a runway.

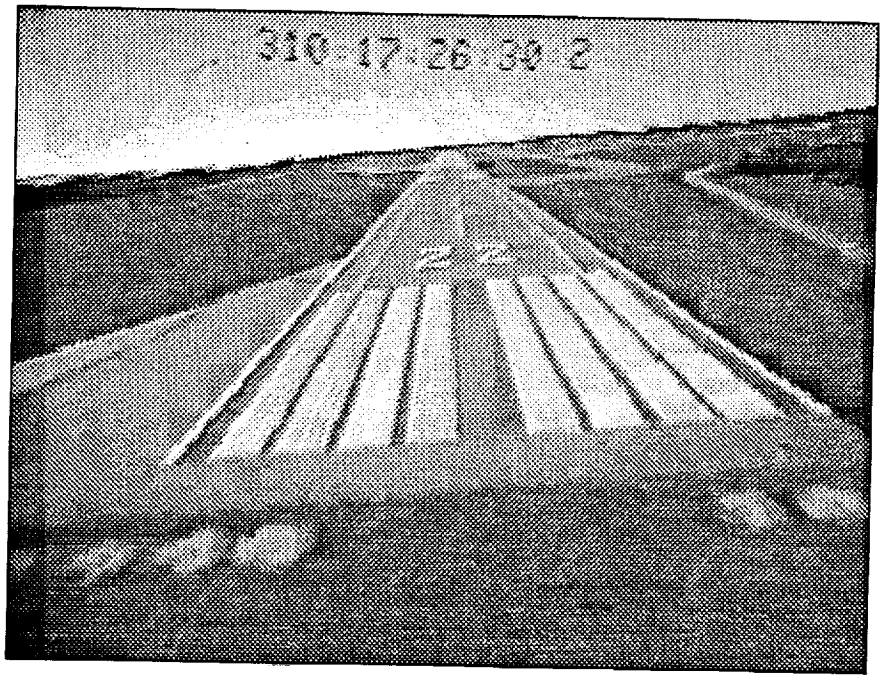


Fig. 3.2: The runway image (Frame 60).

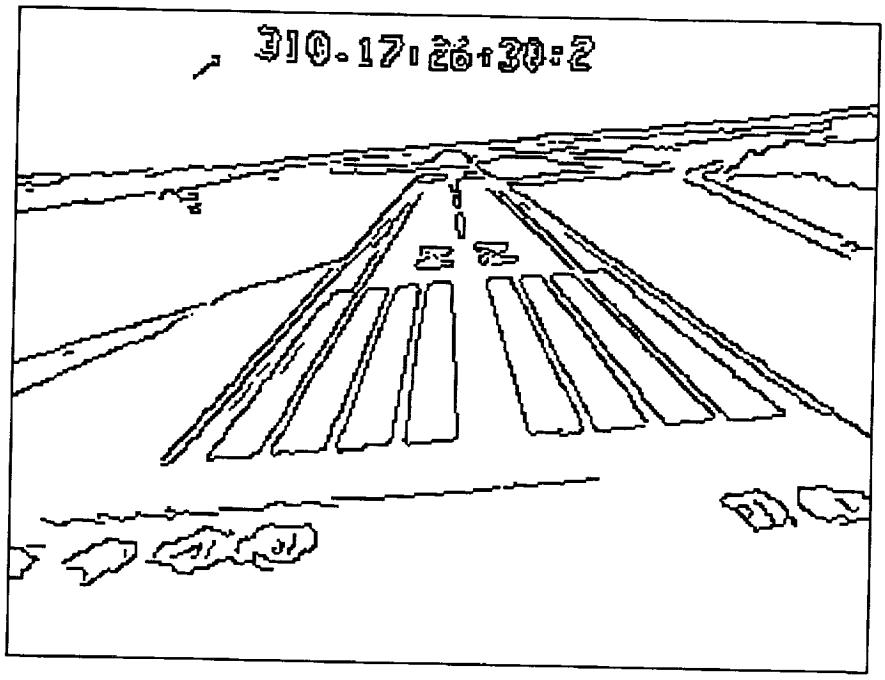


Fig. 3.3: The edge image.

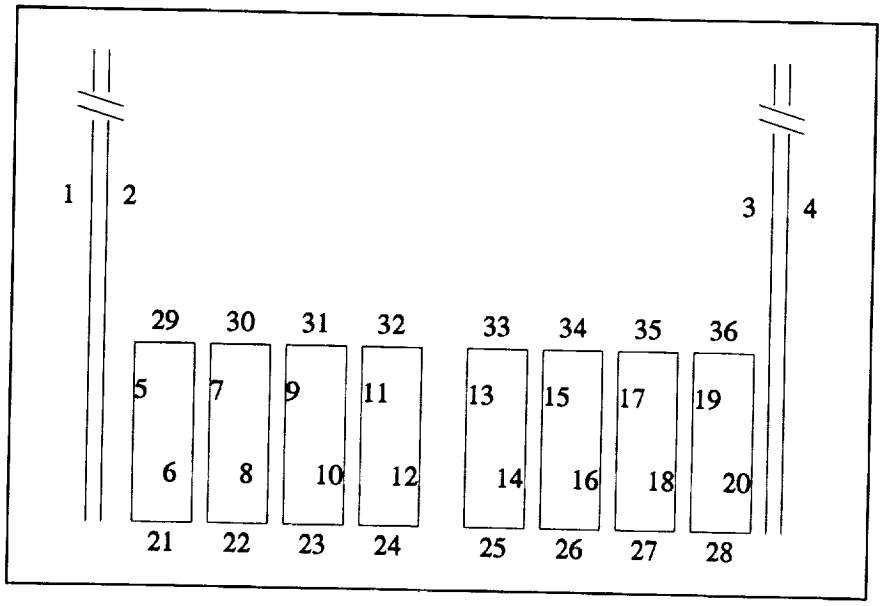


Fig. 3.4: Selected runway features.

these line segments are the image features. Each line segment is represented by several parameters including the two end points, mid point, length, orientation, the variance in length and in orientation, etc. Fig. 3.5 shows the detected line segments. The next step is to perform the runway recognition.

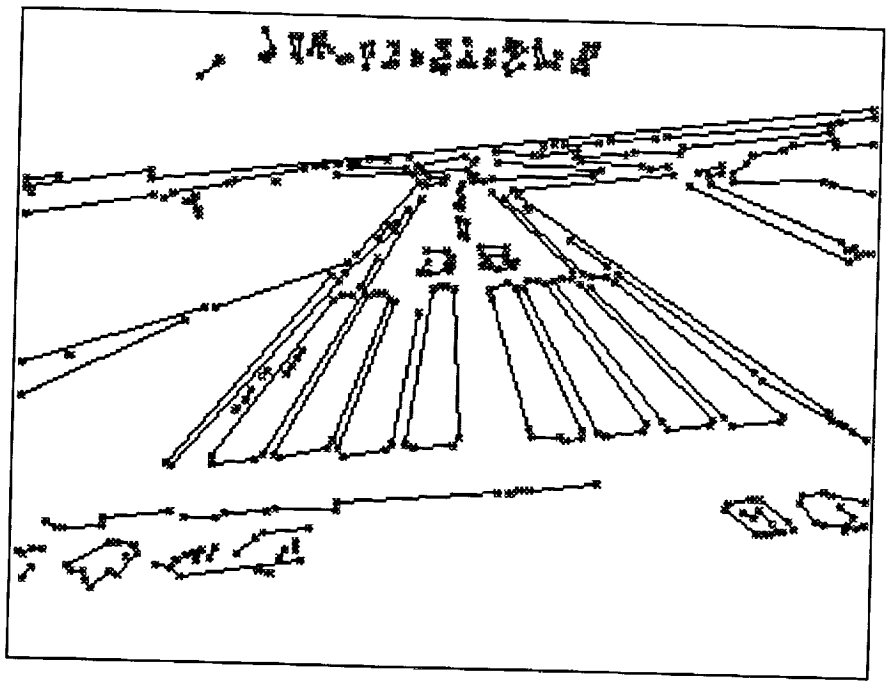


Fig. 3.5: The detected line segments.

3.2. Model-Based Runway Recognition

In the model-based object recognition, the stored geometric model is matched against features extracted from an image. An interpretation of an image consists of a set of model and image feature pairs so that there exists a particular type of transformation

that maps each model feature onto its corresponding image feature. Thus, the recognition process can be considered as a sequential matching process where the image features are matched against the model features. An image feature is said to match a model feature if it satisfies certain constraints defined together with the model features. For line segments, the constraints may be the length, the orientation, and the geometric relationships with other features. Another important product of the matching process is the transformation matrix from model to image coordinate frames. This matrix determines the location and orientation of the object relative to the camera. Since the search space is usually far too large for the approach to be practical, especially when occlusions, noise, and spurious data are present, efforts have made been to reduce the space. Several matching techniques have been developed, including *geometric constraints* (Grimson, 1990), *alignment* (Ayache & Faugeras, 1986; Dhome et al., 1989; Huttenlocher & Ullman, 1987), *geometric hashing* (Lamdan et al., 1988; Lamdan & Wolfson 1988; Wolfson, 1990), *generalized Hough transform* (Silbergberg et al., 1984; Thompson & Mundy, 1987; Turney et al., 1985), etc.

For the runway recognition problem, several aspects that differ from ordinary recognition problems are noticeable. First, the object to be matched has a simple geometric shape which consists of a group of approximately *coplanar* straight lines. Second, with the knowledge of the camera pose, the approximate two-dimensional (2D) projection of the object on the image is predictable, making the computation for the

transformation matrix unnecessary. These characteristics considerably reduce the complexity of the matching process. Third, there is little occlusion problem. The only problems to deal with are the missing features and spurious data. This is obvious from Fig. 3.5, where several model features are missed by the line detector and there are cases where multiple line reports correspond to a single model feature. And finally, the scaling problem has to be explicitly dealt with. As the aircraft may be very far away from the runway, some small features may not be visible at all in the image. Therefore, special care should be taken to rule out the model features which may not be easy to detect in the image. The next section describes the runway recognition method.

3.2.1. Regions of Interest

For an aircraft equipped with Inertial Navigation Systems, the information about the aircraft/camera state is continuously available as the aircraft moves. This information is very important to reduce the complexity of the recognition algorithm and the matching result is significantly improved as well. If the camera state information is exact, the 2D shape and the 2D model features of the runway are readily obtainable simply by performing 3D to 2D transformation and projection using the known transformation matrix. Therefore, the 3D matching problem becomes a 2D problem — just to match the 2D shape of the object using the sensor data. Since the camera state information is inexact, a method based on defining the *regions of interest* (ROIs) on the image plane is

developed. An ROI is an area in the image where a certain feature is expected to appear given a range of different camera states. Fig. 3.6 shows an ROI for a model feature. The more accurate the camera state information, the smaller the area. By defining such an area, searching for a match of a model feature in the image is performed only within that area and nowhere else. The use of the ROI is a very strong constraint which reduces the search space significantly. The geometric constraints for a single feature are also defined by an ROI. Fig. 3.7 shows the constraints for matching a model feature, where the 2D model feature has a length of d and an orientation of α. The candidate image features should have a length not exceeding d and an orientation different from α by no more than β. β is the angle between the two lines joining, respectively, the two pairs of opposite vertices of the ROI. Only those image features satisfying these two constraints can be considered as potential matches. Defining the ROI also implicitly preserves the geometric properties or constraints among the features since the set of ROIs clearly maintains the original 2D shape of the object. As can be seen, the ROIs not only provide geometric constraints in matching, but also speeds up the matching process. This is essential in designing an efficient and reliable matching method. In our previous work, using ROIs has proved that the runway can be reliably detected in a sequence of Passive Millimeter-Wave images (Tang et al., 1993, 1994).

To define the ROIs in the image, the camera state information is used to determine the extent of the ROIs. The camera state relative to the world coordinate frame is

represented by six parameters (X_C, Y_C, Z_C, ψ, θ, ϕ) described in section 2.2. For the particular INS used in the experiments, the inaccuracies are (X_C±2.0, Y_C±2.0, Z_C±2.0) for the position and (ψ±0.0158, θ±0.0056, ϕ±0.0059) for the orientation, where the unit of the position is feet and that of the orientation is radians (Smith, 1990). If the two extreme values are considered for each variable, there can be 64 (2^6) different camera states. Accordingly, each model feature can have 64 positions in the image with one position corresponding to a particular camera state. The situation is described in Fig. 3.8, where the two end points of a model feature each have 64 possible positions in the image. If we define a bounding box which encloses all these points, we have defined a ROI for the feature. As a result, an ROI fully specifies the possible locations and orientations of the feature to appear in the image, and the feature matcher seeks to find the best matching image feature only within the corresponding ROI for that model feature. A significant amount of computation can thus be saved, and the matching is more accurate.

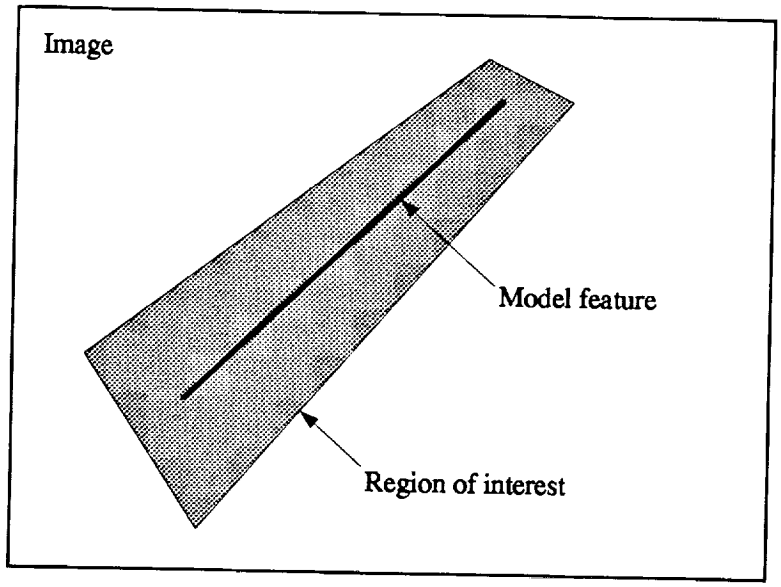


Fig. 3.6: An ROI is a bounding box enclosing the model feature.

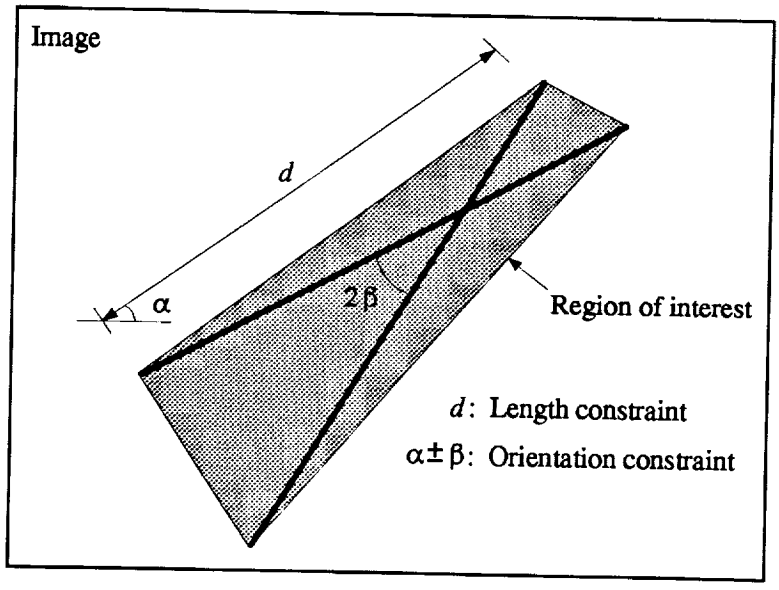


Fig. 3.7: The constraints associated with the ROI.

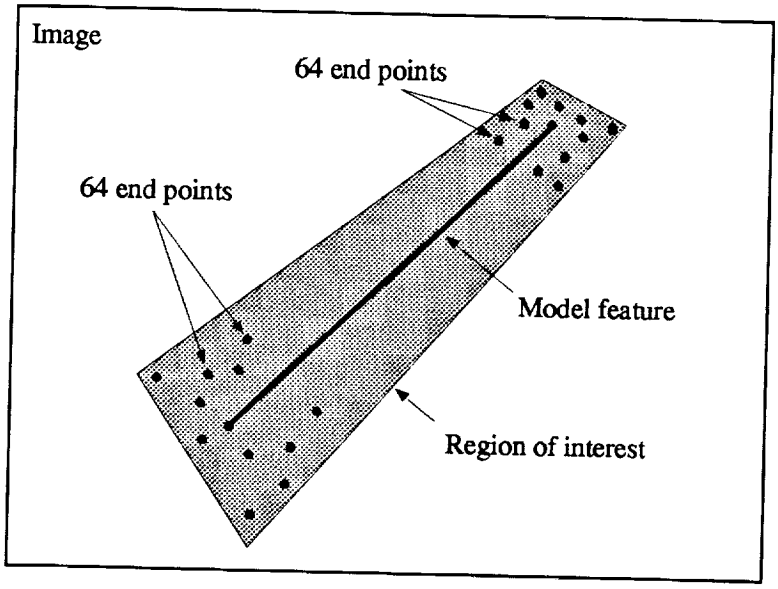


Fig. 3.8: The ROI is defined as a bounding box which encloses all possible end points of the feature.

The horizon line is an important clue in verifying the camera orientation since it gives the roll angle information directly. In addition, the horizon line, if appearing in the image, is very salient and is relatively easy to detect. Hence, a ROI is also defined for the horizon line. When the vertical angular field of view is larger than 2θ, a horizon line will appear in the image (see Fig. 3.9). Without loss of generality, consider the situation when the aircraft is heading towards the X axis of the world coordinate system. Assume the camera is located at point D (see Fig. 3.9) with pitch angle θ, and zero yaw and roll angles. Points A and B are the top and bottom edge of the image, respectively. The horizon will then appear horizontally in the image plane as shown. The distance between this line and the center line of the image is given by $\overline{HC} = f \tan\theta$. Since in the above analysis, roll

angle has been assumed to be zero, the horizon appears parallel to the horizontal axis of the image plane. For any non-zero roll angle, a simple rotation of this line will give the horizon in the image. The associated ROI is defined to be 20 pixels wide (on the 240x320 image) centered around the expected horizon line in the image. Fig. 3.10 shows all the ROIs defined for the horizon line and the runway model features given a camera state and its inaccuracies.

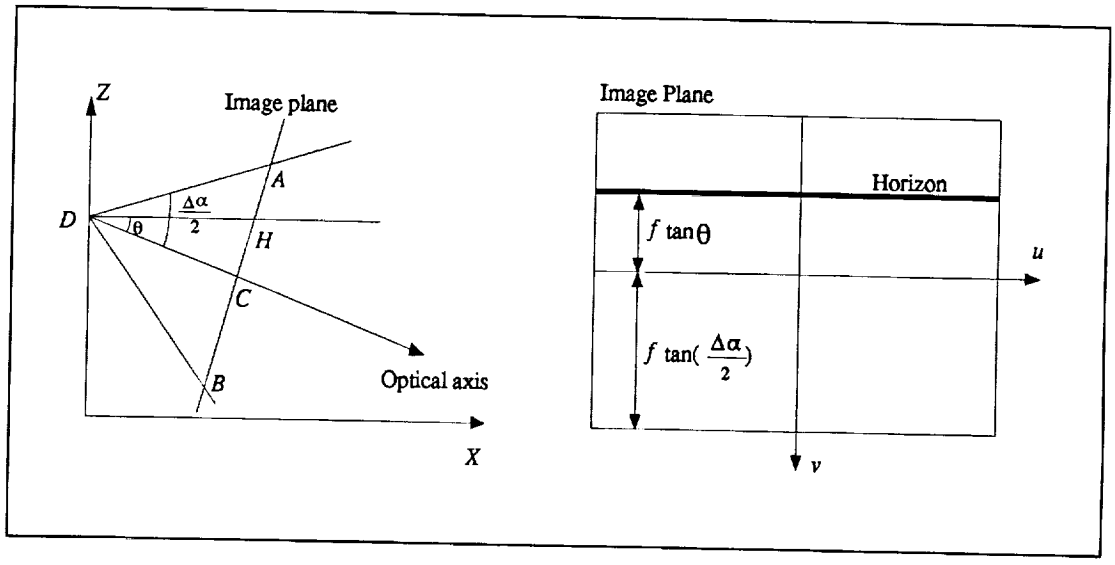


Fig. 3.9: Horizon line in the image (zero roll angle).

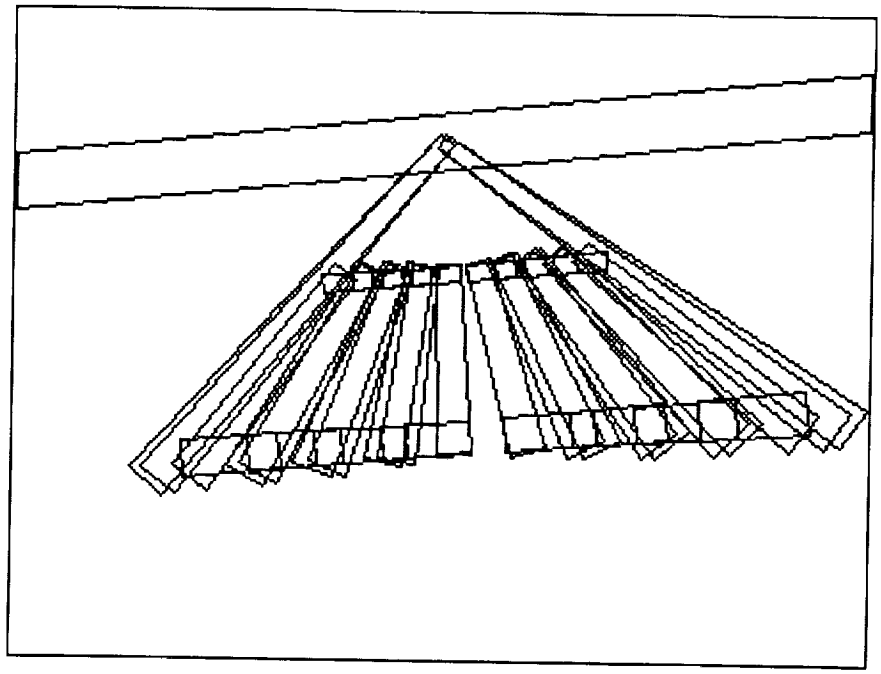


Fig. 3.10: The ROIs defined for the horizon and the runway in Frame 60.

3.2.2. Creating the 2D Model and the Regions of Interest

Before conducting the matching process, we have to obtain the 2D image version of the 3D object. Each 3D end point of a model feature is first transformed to obtain 64 positions on the image plane by using the known 64 transformation matrices mentioned earlier. And then, a bounding box can be computed on the image plane which encloses all the 128 points (two end points) for a 2D model feature. Polygon and line clipping (Foley et al., 1990) is necessary if only a part of the bounding box or the model feature is inside the image. Now a set of ROIs is created for the 2D model features. Next, for each 2D feature, which may have been clipped, compute its length in the image. The summation of

all the feature lengths is also computed which gives the total image length of the 2D model. The total length is used for measuring the goodness of the matching result to be described later. The feature list is then sorted according to the feature length. Having an ordered set of features is advantageous because longer features tend to be detected more likely and reliably, so they should have higher priority to be matched first. As the runway may appear different depending on the location and orientation of the camera, more work needs to be done to discard non-detectable 2D model features. For example, the 8 stripes will appear too small to be detected if the aircraft is too far away from the runway. Also, if the aircraft is not heading in the same direction as the runway direction, two adjacent features may appear so close that they will merge into one or they may not be visible at all. Therefore, only those features with a length greater than a certain threshold are retained. Features with a small distance to its neighbors are discarded also. After all these processes, we have an ordered 2D model feature list; each feature is associated with an ROI. And the geometric constraints, i.e., the length and the orientation constraints, can be computed for each ROI.

3.2.3. Feature Matching

In matching a model feature, some factors have to be taken into consideration. First, usually there are several image features in the ROI (see Fig. 3.11). Some image features may be missed or broken by the detector and there may be multiple reports within an ROI

due to either noise or overlapping of ROIs. If ROIs overlap, some image features in the ROI would actually correspond to other model features. Second, the line segments may be broken. Hence, several collinear image features should match a single model feature (see Fig. 3.12). With these observations, the following four-step algorithm is designed to perform the matching task for each model feature sequentially.

Step 1: Extract all the image features that fall completely within the defined ROI. That is, both end points of an image feature have to be inside the ROI in order to satisfy the length constraint. However, small image features (< 3 pixels in length) are not included since they tend to be due to noise.

Step 2: Discard those image features whose orientations do not satisfy the orientation constraint.

Step 3: Group features if they are collinear. This is to join the broken but collinear image features. The total length of the collinear line segments is then computed.

Step 4: Sort the feature groups according to the total length and choose the group with the greatest length as the best match. Since the model features have been sorted, the group with the greatest total length should be assigned first.

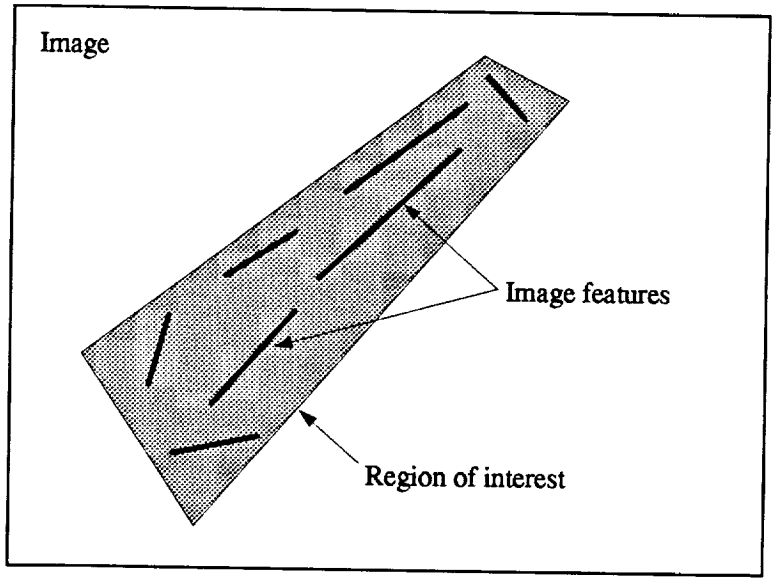


Fig. 3.11: Multiple image feature reports within an ROI.

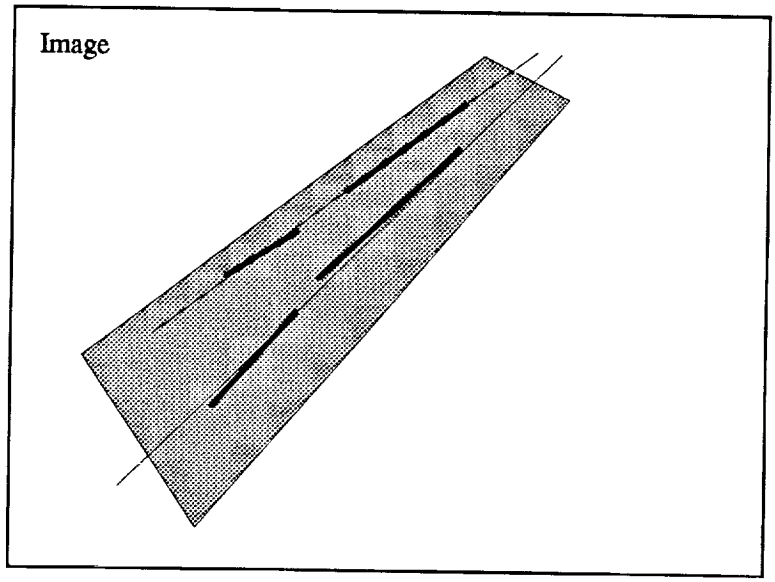


Fig. 3.12: Two collinear lines detected from image features which also satisfy the geometric constraints.

As mentioned before, the geometric constraints (relationships) *among* model features are implied by the definition of ROI since the set of ROIs still preserves the 2D shape of the object model. Therefore, matching against model features can be done sequentially without checking with other already matched results. This can save a tremendous amount of computation time.

3.3. Measure of Goodness

A measure of the goodness of matching is computed to evaluate the result of runway recognition. Let the length of the model feature f_i be D_i, the total length of the model be L, the image features matched to m_i be $\{s_j^i\}_{j=1...n_i}$ and the length of each s_j^i be e_j^i. We have

$$L = \sum_{i=1}^{m} D_i \quad \text{and} \quad d_i = \sum_{j=1}^{n_i} e_j^i . \tag{3.2}$$

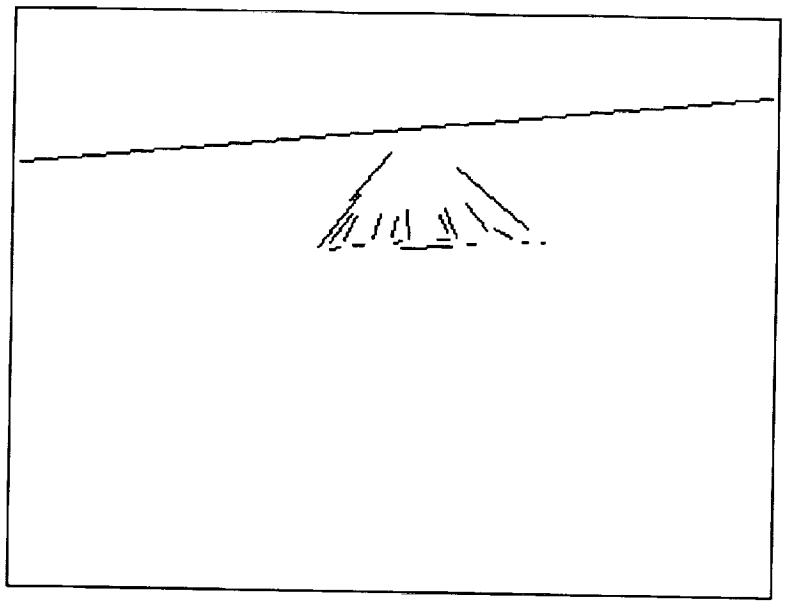
The measure of goodness Ω is defined as

$$\Omega = \sum_{i=1}^{m} w_i \cdot r_i = \sum_{i=1}^{m} \frac{D_i}{L} \cdot \frac{d_i}{D_i} = \frac{1}{L} \sum_{i=1}^{m} \sum_{j=1}^{n_i} e_j^i . \tag{3.3}$$

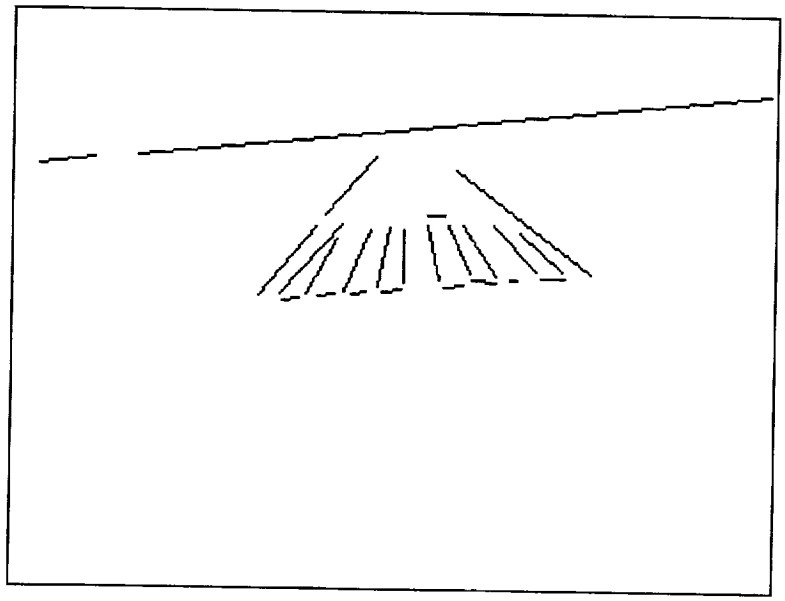
where w_i ($=D_i/L$) represents the weight of the model feature f_i in the object model, i.e., the importance of the detection of this model feature to the complete detection of the

object. It is measured as the ratio of the length of the model feature to the total model length. Features with greater length are more important to be detected in order to recognize the whole object. Hence, they should have more weight in the goodness measure. Another measure, r_i ($=d_i/D_i$), represents the result of matching for the model feature f_i, i.e., how much of the model feature has been detected. It is defined as the ratio of the total length of the matched image features to the expected length of the model feature.

As a result, the measure of goodness Ω is the ratio of the total length of *all* matched image features to the total length of the object model. If Ω is larger than a certain threshold, the object is recognized in the image. In the experiments, Ω is set to 0.6. Fig. 3.13 shows the recognition results of a number of runway image frames.

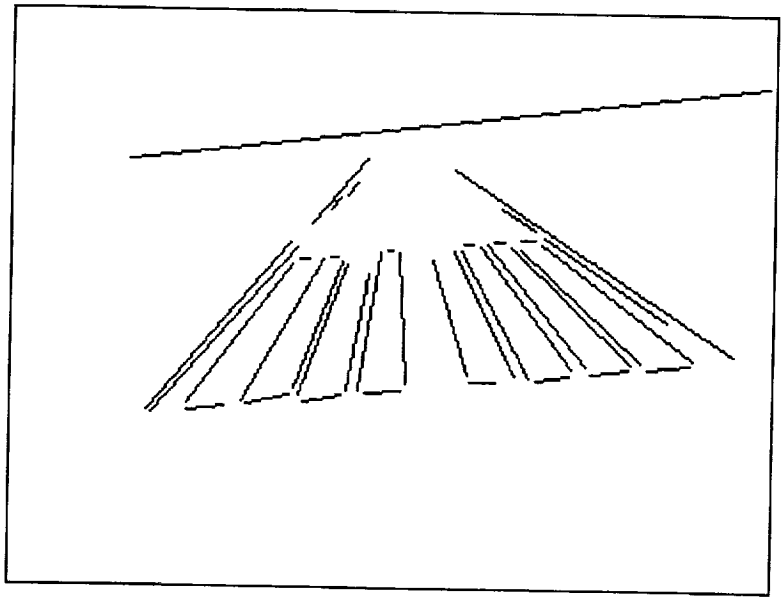


(a) Frame 58 (Ω=62%)

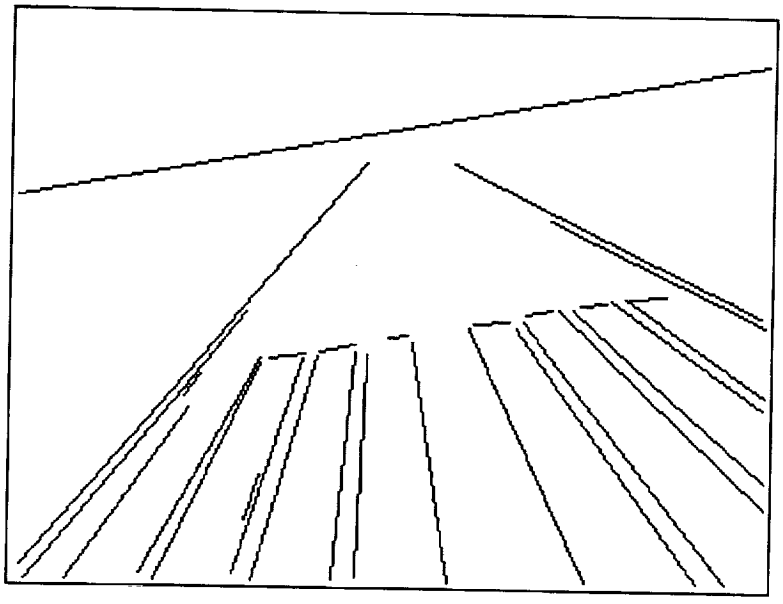


(b) Frame 59 (Ω=71%)

Fig. 3.13: The detected horizon and runway in Frames 58 to 61 (a to d).



(c) Frame 60 (Ω=78%)



(d) Frame 61 (Ω=82%)
Fig. 3.13: (continued)

3.4. Analysis

The algorithm described in this chapter is mainly designed for model-based recognition with the knowledge of approximate camera location and orientation. Features are defined as line segments. Hence, this algorithm depends on the accuracy of the camera state information and the quality of line detection. Loosely speaking, the computation complexity of the algorithm is $O(mn)$, where m is the number of model features and n is the number of image features. Strictly speaking, however, since considerable heuristics are used to rule out image features which cannot possibly be matching candidates, the computation complexity is really $O(m)$. That is, by defining the ROI and using the geometric constraints, the number of match candidates to be processed for a single model feature is almost a constant in the order of 10 or less depending upon the accuracy of the camera state information. Fast recognition is thus achievable. Due to the large size of the runway, there is little occlusion problem. Only the noise from the line detection algorithm has to be dealt with.

It is demonstrated in this chapter that, with the knowledge of the approximate camera state information, computation complexity and recognition quality can be greatly improved. The recognition of the runway has several advantages. First, detection of the runway verifies that the aircraft is heading in the correct direction, and hence, increases the landing safety. Second, it is possible to use the recognized runway to refine the camera state information, resulting in a more robust multi-sensor system. And finally, the image

can be divided into several regions, such as the sky, the runway, and the regions outside the runway. Different image analysis techniques can thus be performed in different areas, which makes a speedup and better processing possible.

4. Position estimation for stationary objects

In autonomous navigation, it is essential to obtain a three-dimensional description of the static environment in which the vehicle is traveling. For rotorcraft conducting low-altitude flight, this description is particularly useful to detect and avoid obstacles in the intended flightpath. This technique is generally referred to as *structure from motion* (Longuet-Higgins, 1981; Spetsakis & Aloimonos, 1987; Tsai & Huang, 1984; Tseng & Sood, 1989; Weng et al., 1992), where the 3D structure of the static scene is estimated, using more than two image frames captured at different camera locations and/or orientations. Many approaches have used line or point correspondences among two to four images to compute the camera motion and the scene structure (McIntosh & Mutch, 1988; Mitiche et al., 1989; Liu & Huang, 1988). Results showed that the solutions from these methods are very noisy. Other approaches overcome this problem by integrating information from a long sequence of images and Kalman filtering is a common choice to obtain a smoothed estimate. The Kalman filtering technique is popular because of its elegant way of handling uncertainty and providing incremental processing. Broida and

Chellappa (1986) used Kalman filtering as a recursive means to estimate object motion parameters. Matthies et al. (1989) built a framework which gives depth estimates for every pixel in the image. In their experiments, the side-viewing camera is assumed and the camera motion is only translational in the vertical direction. Under such conditions, feature tracks will follow the vertical image scan lines, and feature matching becomes much simpler.

Baker and Bolles (1989) used *Epipolar Plane Image (EPI) Analysis* for static scene analysis. In their approach, the camera moving path is known and linear. Therefore, each image frame can be decomposed into a set of *epipolar lines*. Supposing the environment is static and the camera is moving in a straight line, the epipolar lines are the paths the motion of the features *will* follow between image frames. This phenomenon is called the *epipolar constraint* since image features are constrained to move only along the epipolar lines. If a sufficient number of images are accumulated to form a solid block referred to as the *spatiotemporal* (ST) *data*, an epipolar plane image can be created by collecting corresponding epipolar lines in each image frame. Furthermore, in their experiments the camera is moving only laterally and the camera viewing direction is always orthogonal to the motion path. In such a simple case, an EPI will be a horizontal slice of the spatiotemporal data, and the apparent motion track of a feature on the EPI will be a straight line. The feature motion analysis thus becomes merely a line fitting process on the EPI, and the 3D location of the tracked feature can be determined by the parameters of the

fitted line. In the case of forward linear camera motion, however, the feature tracks will be hyperbolas and curve fitting becomes necessary. Sawhney et al. (1993) reported that curve fitting is much more difficult and noisy, making this approach less robust. In another paper by Bolles et al. (1987), the Kalman filtering technique is used to estimate the range in the case of forward linear camera motion. Also dealt with was the nonlinear camera motion case where the nonlinear motion path is restricted to be on a horizontal plane and where only *one* of the EPIs in the spatiotemporal data, parallel to the motion plane, can be analyzed. This restriction makes their approach less useful in practice.

In the course of navigation, the robot or the vehicle has to estimate the range from an obstacle to the camera in order to avoid it by changing its nominal path. Several methods for range estimation have been investigated at NASA Ames Research Center (Bhanu et al., 1989b; Cheng & Sridhar, 1991; Roberts et al., 1991; Smith et al., 1992; Sridhar et al., 1989; Sridhar & Phatak, 1988). Their main approach is also to use Kalman filtering to recursively refine the estimated range. In this chapter, an approach (Tang & Kasturi, 1994) is developed for stationary object position estimation using known camera state parameters. The epipolar constraint is the main criterion in tracking image features. Since the camera state information is continuously available from the onboard navigation system, it can be used to facilitate the process of scene reconstruction. For example, the location of the focus of expansion (FOE) in the image plane can be readily determined. In addition, if the image acquisition rate is high enough, an image feature will not move by

more than a few pixels in the next image frame. The correspondence problem between successive images can thus be minimized. The camera state information and high image acquisition rate are the two main requirements for the proposed method. As is well known, depth estimation under the forward moving camera situation is difficult and noisy because the optical flow in the image is generally small compared to lateral motion cases, especially when objects in the scene are far away from the camera. This problem is overcome in the proposed algorithm by integrating information over a long sequence of images. As image frames are accumulated, the range estimates can be refined by taking more data into account. Hence, estimates will be more accurate . In our approach, the problem of general 3D camera motion is dealt with. This is handled by breaking the camera motion path into piece-wise linear segments. Through this process, the camera path determined by two successive camera locations is approximated as a straight line. Epipolar planes can thus be set up for each pair of images, and motion analysis is repeatedly performed on each pair of image frames. The developed algorithms were tested on the helicopter image sequences.

4.1. Constructing the Epipolar Lines

As is well known, if the scene is static and the camera is moving in a straight line, the motion path of image features between successive images will follow a line termed the *epipolar line*. This constraint is thus called the *epipolar constraint*. This constraint gives

a strong restriction in confining the apparent motion directions of image features. If the epipolar lines are available in the image, the problem of feature tracking can be significantly simplified. In fact, the epipolar lines are determined by the *epipolar planes* as shown in Fig. 4.1, where all image frames share a common set of epipolar planes. As can be seen in the figure, no matter how the camera changes its location and orientation, as long as the motion path is linear, all the lines-of-sight (projection lines) from the camera centers to a particular static world point always lie on the same epipolar plane which is determined by the 3D location of the scene point and the camera path. The intersections of the epipolar planes and an image plane defines the epipolar lines. Any image feature will then drift between images only along the corresponding epipolar line. Therefore, tracking of an image feature is simple — just along the epipolar line.

Since we are dealing with general 3D camera motion, the moving path of the camera is not a straight line, and its orientation is not constant during motion. Fig. 4.2 illustrates such a situation, where the camera's path is an arc. Even if the camera is fixed on the helicopter, its orientation is still changing because the helicopter body orientation is changing during nonlinear motion. The location of the FOE on the image plane also changes significantly. In this case, there is no common set of epipolar planes for all the image frames. Hence, it is impossible to perform analysis using the epipolar constraint. This problem is overcome in the proposed method by using the piece-wise linear approximation for the camera path. Since the image acquisition rate is very high (30

frames/sec), between two consecutive image frames, the camera path can be approximated as linear and hence a pencil of epipolar planes can be created. Analysis using the epipolar constraint is thus possible. In the following, the construction of the epipolar planes and lines is described. The appendix gives a more detailed description.

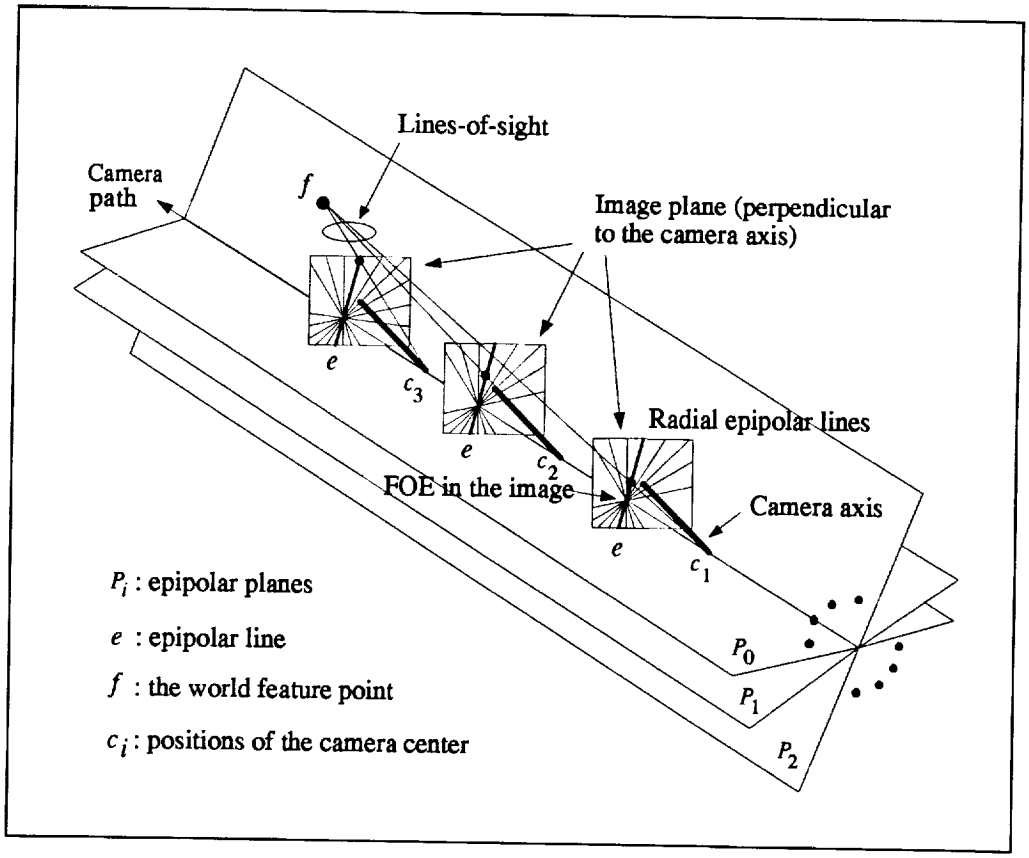


Fig. 4.1: Geometric relationships between the camera path, the epipolar planes, and the epipolar lines. The camera is moving in a straight line and its orientation is constant.

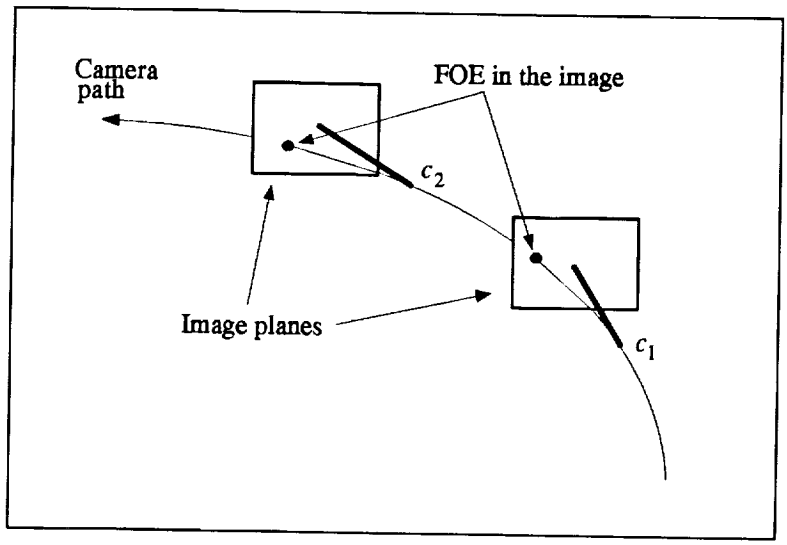
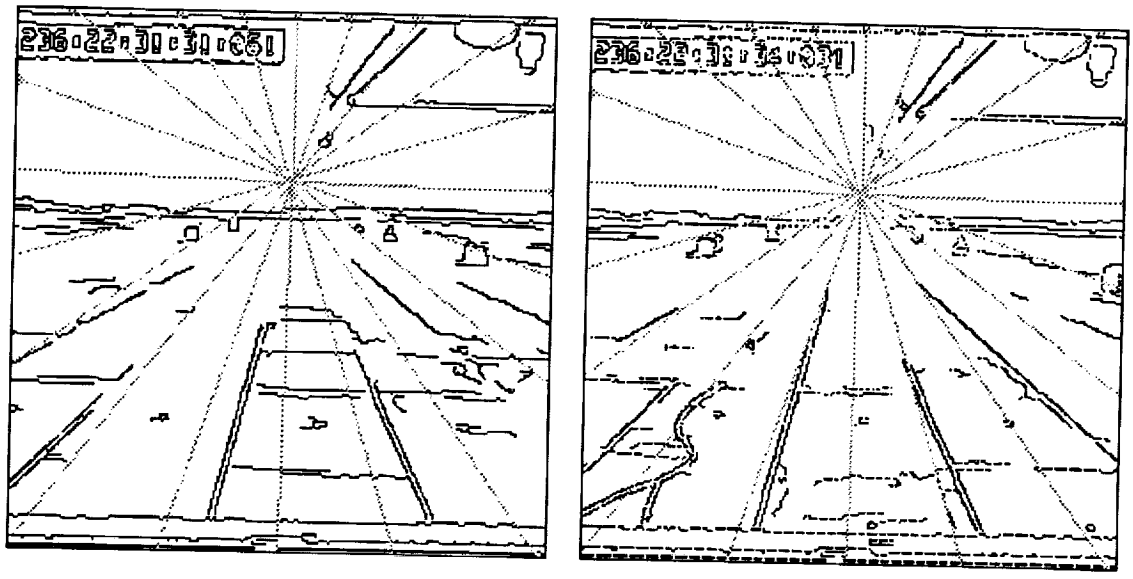


Fig. 4.2: There is no common set of epipolar planes if the camera path is not linear. The locations of the FOE in the images also varies.

For each pair of image frames, the camera path parameters are first computed from the input camera state data. A pencil of Q epipolar planes, $\{P_j\}_{j=0...Q-1}$, is then defined, which all intersect at the camera path as shown in Fig. 4.1. Q determines the resolution of the 3D space (Q is set to 100 in all the experiments). The angle between two adjacent epipolar planes P_j and P_{j+1} is equal to π/Q. The result of this process is the construction of a pencil of epipolar planes equally spaced in terms of their angular orientations; they all intersect at the camera path. After creation of the epipolar planes, epipolar lines on each image plane can thus be determined by intersecting the image plane with the epipolar planes. The process of creating the epipolar lines is repeatedly performed on each pair of image frames in the sequence. Fig. 4.3 shows the epipolar lines superimposed on the

edge-detected images of the *line* sequence. The intersection of all the epipolar lines shows the location of the FOE in the image. Even for the *line* sequence in which the FOE is expected to remain at the same pixel location in the image, it changes by about 25 pixels, both horizontally and vertically. For the *arc* sequence, the FOE location varies by about 70 pixels horizontally and 30 pixels vertically.



(a) (b)

Fig. 4.3: The epipolar lines superimposed on the edge detected images (every tenth line is shown). (a), (b): the first and the last frames of the *line* sequence.

4.2. Feature Detection and Tracking

The purpose of constructing epipolar lines is twofold: to detect features and to facilitate feature tracking. Features in an image are defined to be the intersecting points between the epipolar lines and the edge pixels detected by Canny's edge detector (Canny, 1986). The features are extracted from the first image by tracing the pencil of epipolar lines, $\{l_j\}_{j=0...Q-1}$. The result will be Q sets of feature points. Feature detection and tracking on the following frames are completely independent among these sets. To obtain good localization of detected features, the edges in the image should be nearly perpendicular to the epipolar lines. It is obvious that since Q directly determines the number of features to be processed, we can easily increase or decrease the number of features by adjusting the value of Q.

In dealing with general camera motion, the camera orientations for two frames are likely to be different. Traditional feature matchers try to search target features within the *neighborhood* (i.e., the 2D search window) surrounding the source feature to be matched. With the information of the camera state, we argue that this method of positioning the search window is inappropriate because the term *neighborhood* is improperly defined. The following statement is one of the implicit assumptions in defining the *neighborhood* to be a 2D window centered around the source feature:

> "Since the image acquisition rate is high enough that the camera will not move for a long distance between two consecutive frames, the source feature will not move by more than a few pixels in the next image plane."

We find this statement is sustained only if the camera orientation keeps approximately constant during the camera motion. As is well known, even a small amount of camera rotation can actually create large image feature motion in the image plane. Hence, the camera rotation has much more influence on image feature motion compared to camera translation, especially when the distance from the camera to a world feature is very large compared to the distance the camera travels between two consecutive frames. This can be best illustrated with the 2D world shown in Fig. 4.4, where the camera moves from c_1 to c_2 with orientation changes. The projections of a far-away world feature onto the two frames are p_1 and p_2, respectively, and p_1' specifies the same image location as p_1 in the second frame. Due to large camera rotation, p_2 is not, however, at the neighborhood of p_1'. Under such conditions, the search window should be positioned around p_2 instead of p_1'. This is exactly our situation where the velocity of the camera is about 1 foot/frame and the major features of interest in the image are hundreds of feet away. Also, for the *arc* sequence, the instantaneous orientation of the camera is continuously changing since the flight path is nonlinear. Experiments showed that if we perform tracking on the *arc* sequence by searching the neighborhood of the source feature, the correct corresponding feature may be completely out of the search window.

From the argument above, the term *neighborhood* can be redefined as follows. It is observed that the only thing guaranteed by high image acquisition rate is that the vector of the lines-of-sight to a far-away world feature will not change drastically between two consecutive frames, i.e., the vectors $\vec{v}_1$ and $\vec{v}_2$ in Fig. 4.4 should be approximately the same. Hence, the *neighborhood* of feature p_1 is redefined as "the region surrounding the intersection between the image plane of frame 2 and the vector through c_2 and parallel to $\vec{v}_1$." And the feature tracker will search within this newly defined *neighborhood* for a match for feature p_1. This is the main reason for the success of the proposed feature tracker.

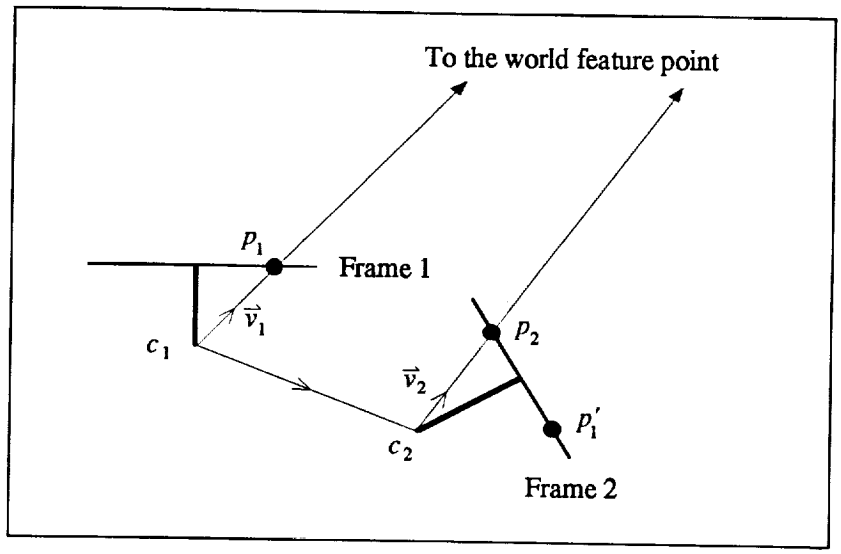


Fig. 4.4: The orientation of the lines-of-sight to a far-away feature remains approximately the same between images. However, the orientation of the camera may change drastically.

The feature tracker is implemented as follows. For each pair of image frames, the locations of the FOE on each image plane are first computed. Let subscripts 1 and 2 denote the time instance of the first and the second frame, respectively. For each image feature at location p_1 in the first frame, I_1, we first compute $\vec{v}_1$, which is the 3D vector from the camera center c_1 to p_1. And then, the hypothetical location p_2 is obtained by intersecting I_2 with the vector $\vec{v}_1$ passing through camera center c_2. Incorporating the epipolar constraint, instead of searching within a neighborhood centered around p_2, the feature tracker searches along the direction of the epipolar line, which is determined by the FOE and p_2. This neighborhood is actually a one-dimensional (1D) search window oriented in the direction of the corresponding epipolar line as shown in Fig. 4.5 (in the experiment, seven pixels are used as the window size). Feature detection and tracking are then performed within this window on the second image. Note that, in order to reduce the amount of computation, the feature detection and tracking are performed based on the edge pixels only. For more robust feature tracking, the intensity distribution around a feature should be considered.

Within a small 1D *neighborhood*, there are a number of source features in the first frame and a number of target features in the second frame. They are to be matched. Depending on the matching results, a feature can be labeled as **matched**, **new**, **no match**, and **multiple matches** described as follows:

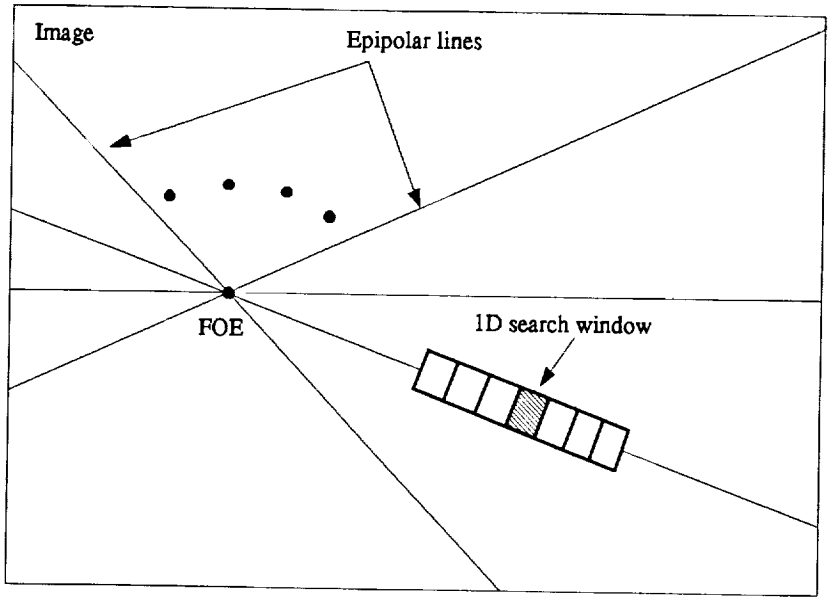


Fig. 4.5: The 1D search window

1. Matched: There is only one target feature. If there are several source features that are competing for the target (i.e., the occlusion case), the target will be matched to the source feature that has a *stable* position estimate. The position estimate of a feature is considered stable at time t if its estimated position remains approximately the same through several frames:

$$\frac{1}{r}\sum_{k=t-r}^{t-1}\left\{(\hat{X}_k-\hat{X}_t)^2+(\hat{Y}_k-\hat{Y}_t)^2+(\hat{Z}_k-\hat{Z}_t)^2\right\}<T, \qquad (4.1)$$

where $(\hat{X}_k,\hat{Y}_k,\hat{Z}_k)$ is the estimated position at time k, T is a threshold and r is the number of frames considered (r is set to 10 in the experiments). If two or more source features

have a stable position estimate, the local slope of their tracks is compared to determine the best match result. The comparison is based on the fact that the image feature corresponding to a more distant world feature will create less optical flow in the image. Fig. 4.6(a) illustrates such a situation. Supposing source *A* and *B* in the first frame (frame 3) can both match to target *C* in the second frame (frame 4), the feature track with a steeper slope (solid squares), i.e., the one which creates less optical flow along the epipolar line, should correspond to the feature which is more distant from the camera. Since only the near feature can occlude the farther one, target *C* should be matched to feature *A*. For the matched source feature, its location in the image will be updated and its 3D position estimated (described in the next section). Feature *B* will be thereafter labeled as **no match** and handled as described later.

2. **New**: When a target feature has no source feature to match, it is classified as a new feature and a feature node will be created in the database.

3. **No match**: Here, the source feature does not match any target feature. This may be due to the failure of feature detector, occlusion, or feature moving out of the image. For the last case, which can be easily detected, the source feature is simply removed. For the other two cases, if the source feature already has a stable 3D position estimate, the feature tracker will make a hypothesis about its image position based on its 3D position estimate. No estimation will be performed on these features except for updating their 2D positions,

using the hypotheses. Features with no stable position estimates, since we have no reliable information about their position, remain in their current state awaiting possible matches in the future. A maximum number of consecutive no-matches is defined to remove those features being occluded or missed by the detector for a long time.

4. **Multiple matches:** In this case, more than one target feature appears within the neighborhood. This may result from feature disocclusion or be due to the noise from the feature detector. The goal here is to choose the best match. Cox (1993) reviewed some of the approaches: *nearest-neighbor* (Crowley et al., 1988), *Mahalanobis* distance (Therrien, 1989), *track-splitting filter* (Smith & Beuchler, 1975), *joint-likelihood* (Morefield, 1977), etc. In our problem, since the epipolar constraint already gives an effective means to improve the tracking process, simple techniques are used to resolve this confusion and at the same time reduce the algorithm complexity. This idea is also supported by three observations. First, if the feature already has a stable 3D position estimate, the best match can be easily found by projecting its position estimate onto the second image and hence the target feature near the projection will be chosen as its match. Second, according to the epipolar constraint, actually only *one* direction, away from FOE when the camera is moving forward, is possible for the feature motion under noise-free circumstances. Hence, the feature motion conforming to this constraint should be favored. And finally, the size of the search window is small, giving only a small number of multiple matches. With these observations, three criteria are used for choosing a best

match. They are listed in the following according to their priority: (1) the one that satisfies the position estimate; (2) the one that is in the direction away from the FOE and is nearest; and (3) the one that is in the direction towards the FOE and is nearest. These three criteria also determine the weights in the position estimation. The weight for each criterion is w_p, w_e, and w_n, respectively, where $0 < w_n < w_e < w_p \leq 1$. The main reason to include (3) as a legal match is to compensate for feature detection noise. Fig. 4.6(b) demonstrates how the feature tracker performs the matching task under a complicated situation. Source feature A in Frame 2 is searching for a best match among the target features in Frame 3. Correct track is $AEFG$, but feature E (the blank circle) is missed by the detector. There are also a disocclusion, feature B, and a noise, feature N. Several scenarios and consequences are possible in the tracking process as listed below:

Scenario 1:

Condition: Feature A is stable, feature B is chosen as the best match, and feature F is in B's search window in Frame 4.

Consequence: Since B does not satisfy the estimated location of A, it will be lightly weighted in the position estimation and has little contribution to the estimate. However, according to the estimated position, the tracker will choose the correct match, feature F, as the best match in frame 4.

Scenario 2:

Condition: Feature A is stable, feature B is chosen as the best match, and feature F is not in B's search window or is missed by the detector.

Consequence: The tracker will follow the track ABC, which is incorrect for feature A. Features B and C will contribute lightly to the position estimate. The tracker, however, may still make correction on its tracking if it is possible to match feature G to feature C in Frame 5. For such a case to happen, feature G has to be in the search window of feature C; otherwise the tracker will follow the path of the triangles thereafter, and the position estimates will gradually become unstable. Such a feature can be recognized by noting that its position estimate is still unstable after lengthy tracking. We then have to reset their estimates and start a new estimate similar to that for newly appeared features.

Scenario 3:

Condition: Feature A is stable, feature N is chosen as the best match.

Consequence: This is similar to the above two cases. Feature N has little contribution to the position estimate, and if feature F is in the search window of feature N, the tracker may be able to correct its tracking at Frame 4. If feature F is missed or is not in N's search window, feature C will be chosen in Frame 4. The feature tracker may still find the correct match, feature G, in Frame 5 if G is in C's search window; otherwise, the tracker will follow the triangle path as described in Scenario 2.

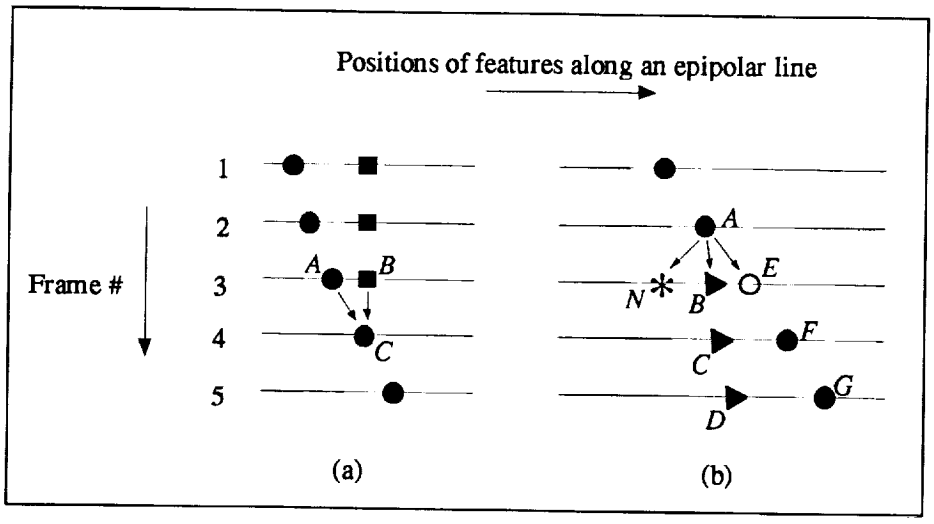


Fig. 4.6: Two matching cases: (a) Occlusion; (b) Disocclusion, missed features, and noise.

Scenario 4:

Condition: Feature *A* is not stable.

Consequence: Either feature *B* or *N* may be chosen to be the best match in Frame 3, and the tracker will follow either the path of triangles (*ABCD* or *ANCD*) if feature *C* is the best match in frame 4 or the path of circles (*ABFG* or *ANFG*) if feature *F* is the best match instead. This does not matter too much since no reliable information has been accumulated and these matches are lightly weighted in the position estimation.

Feature matching is difficult and may contribute most of the error to the estimation. From the analysis above, we can see that the epipolar constraint helps to simplify and

improve the matching quality. The feature tracker may be further improved by incorporating more clues, such as the intensity distribution, edge strength, and edge orientation.

4.3. Position Estimation and Experimental Results

The *incremental weighted least squares* is used for estimating the 3D position of the features being tracked. The main reason for that is its simplicity. Kalman filtering is another choice as described in (Matthies et al., 1989). However, it is desired to demonstrate that using the proposed feature tracker, a simple estimation method can still achieve accurate estimates. As is known, the line-of-sight determined by the camera center and the image feature position will also pass through the 3D position of the world feature. As time elapses, for a certain feature, a set of lines-of-sight can be obtained, which, theoretically, will intersect at its true 3D position. However, due to the finite pixel size, inaccuracies both in the camera parameters and in the feature's image plane estimates, these lines do not exactly intersect at a world point. This suggests that the *point fitting* technique can be a good means to perform the estimation. Let (a_k, b_k, c_k) be the normalized direction vector of the line-of-sight and (X_{ck}, Y_{ck}, Z_{ck}) be the position of the camera center at time k. The distance d_k between the 3D feature position (X, Y, Z) and the line-of-sight can be computed as:

$$d_k = \left\{ \left[c_k(Y - Y_{ck}) - b_k(Z - Z_{ck}) \right]^2 + \left[a_k(Z - Z_{ck}) - c_k(X - X_{ck}) \right]^2 + \left[b_k(X - X_{ck}) - a_k(Y - Y_{ck}) \right]^2 \right\}. \tag{4.2}$$

The objective function J can be defined as the sum of the weighted squares of the distances, i.e.,

$$J = \sum_{k=1}^{t} w_k d_k^2, \tag{4.3}$$

where w_k is the weight and t is the current time. J is minimized to obtain a weighted least squares estimate. This results in a linear estimation problem after differentiating J with respect to X, Y, and Z. The summation is performed incrementally, and as more data are acquired, the estimate is expected to converge to its truth value. The weight is determined by the result of matching, as described in the previous section. It is defined as:

$$w_k = \begin{cases} w_p, \text{ first crition is satisfied (stable position estimate), else} \\ w_e, \text{ second crition is satisfied (epipolar constraint), else} \\ w_n, \text{ third crition is satisfied (nearest),} \end{cases}$$

where $0 < w_n < w_e < w_p \leq 1$ (the values are chosen based on experimental heuristics: $w_p = 0.9$, $w_e = 0.8$, $w_n = 0.7$).

For the *line* sequence, the tracking for the five trucks is successful. Fig. 4.7 shows the position estimates and the relative estimate errors of Truck 1 (right most) as a function

of number of frames processed. The relative error is defined as the difference between the estimated value and the ground truth divided by the Euclidean distance between the camera center and the ground truth. We can see from the figures that the estimation converges after 10 frames and that the relative errors of the estimates for the range, cross range, and height after convergence are within 5%, 1%, and 1%, respectively. As the frame acquisition rate is 30 frames/sec, the estimate converges in less than one second. The reason for this fast convergence is that Truck 1 is the nearest, and the average number of pixels moved between two consecutive frames is about 0.7. The fast moving image features provide better optical flow information for motion estimation. The more accurate estimates are the cross range (Y) and the height (Z) estimates, since there is little lateral or vertical motion. The least accurate is the range (X) estimate, since the helicopter is moving forward. Due to this fact, in all the experiments, the cross range and the height estimates are always more accurate than the range estimates, which have relative errors of about 1%. Table 4.1 summarizes the range estimates for all the five trucks in the *line* sequence. It is shown that the number of frames needed for the range estimates to converge within 10% error increases as the distance to the truck increases. Truck 5 needs more than 90 frames to obtain a more accurate estimate because it is farthest, and the average number of pixels of image feature movement is only 0.1 pixels. Experiments are also performed on the *arc* sequence; Table 4.2 summarizes the estimates. The tracking of Truck 4 is not quite successful. The reasons are that during the initial several frames, the

detected edge is approximately parallel to the corresponding epipolar line, making the tracking more difficult. However, its position estimate converges to within a 25% error after tracking through 90 frames. The estimate would converge to the correct value, provided that more image frames were accumulated.

The experimental results are also compared with those reported by Smith et al. (1992). Table 4.3 summarizes the comparison. As can be seen in the table, the range estimates are considerably better than theirs except for Truck 1. However, as stated in their paper, the length of each truck is 20 feet in length. Range errors less than about 10 feet are unimportant because they indicate range estimates which lie between the front and back of a truck.

4.4. Analysis

In this chapter, a motion estimation system is described that is capable of accurately estimating the 3D object positions in the scene. The piece-wise linear approximation for the flight path is used to facilitate the construction of the epipolar planes and lines. This is very suitable for rotorcraft having an onboard inertial navigation system, where the camera states are always available. The piece-wise linear approximation turns out to be a good method to solve the difficult motion problem resulting from general 3D camera motion. An efficient feature tracker is also developed which makes use of the epipolar constraint to achieve good feature matching results. The weighted incremental least squares estimator

then performs the position estimation. The final relative error of the range estimate for the object approximately 176 feet away (Truck 1 in the *line* sequence) is less than 3%, which corresponds to only 5 feet absolute error. In addition, the range estimation for Truck 1 takes less than 10 frames to converge within 10% error. Since the speed of the helicopter is 35 feet/sec in the *line* sequence, this gives the pilot about 8 seconds to avoid the obstacle. For the estimation on other trucks, the pilot actually has ample time to make a decision about the flightpath. Currently, the algorithms run at the rate of two frames per second on a DECstation 5000/240 for tracking several hundreds of features on image sequences (this excludes the edge detection; the image is of size 512×512 pixels). However, optimization for throughput was not a consideration in this work.

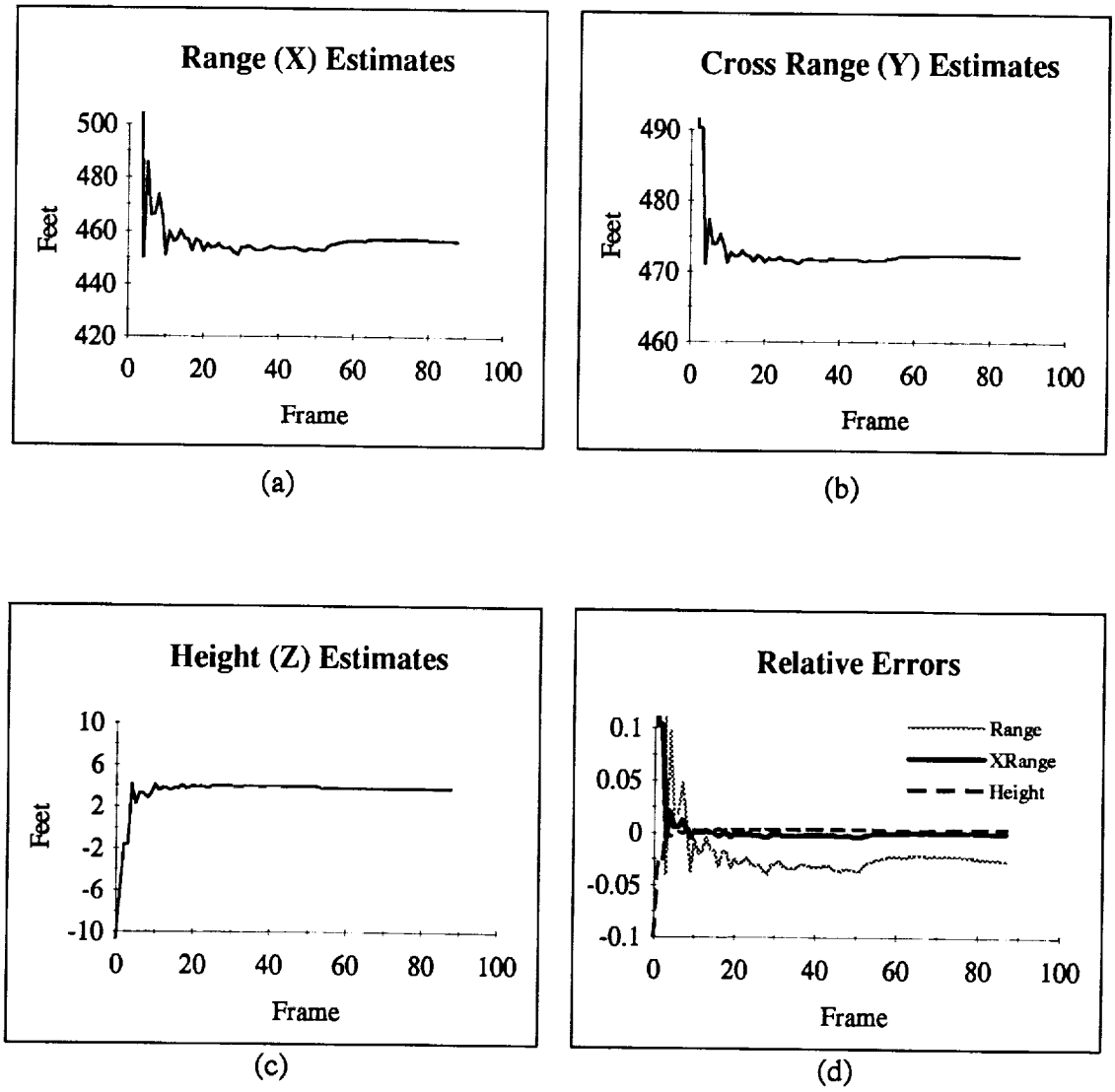


Fig. 4.7: Position estimates of truck 1 (a), (b), (c) and relative errors (d) as a function of the number of frames processed. The ground truth is $(X, Y, Z) = (461.5, 472.3, 3.0)$ with accuracy ±2 feet.

Table 4.1: Position estimates of various trucks after processing 90 *line* frames.

	Initial and final ranges (unit: feet)	Absolute errors (unit: feet)	Relative errors (unit: %)	# frames for 10% range estimate error
Truck 1 (right most)	(272.7, 176.1)	(5.1, 0.1, 3.8)	(2.7, 0.03, 2.0)	5
Truck 2 (left most)	(365.5, 268.9)	(5.1, 0.6, 2.0)	(1.8, 0.2, 0.7)	20
Truck 3 (between center and right most)	(503.0, 406.4)	(12.2, 0.5, 1.2)	(2.8, 0.1, 0.3)	30
Truck 4 (between center and left most)	(615.9, 519.3)	(37.3, 6.0, 0.3)	(7.0, 1.1, 0.1)	70
Truck 5 (center)	(771.8, 675.2)	(83.7, 0.4, 0.5)	(12.4, 0.1, 0.1)	> 90

Table 4.2: Position estimates of various trucks after processing 90 *arc* frames. (Truck 1 is not visible in all frames).

	Initial and final ranges (unit: feet)	Absolute errors (unit: feet)	Relative errors (unit: %)	# frames for 10% range estimate error
Truck 2 (left most)	(224.9, 107.5)	(2.6, 1.3, 0.6)	(1.7, 0.8, 0.4)	20
Truck 3 (right most)	(384.6, 267.2)	(4.1, 1.9, 1.7)	(1.5, 0.7, 0.6)	20
Truck 4 (second from left)	(475.3, 357.9)	(90.0, 19.2, 0.6)	(24.5, 5.2, 0.2)	> 90
Truck 5 (second from right)	(631.2, 513.8)	(79.6, 3.7, 1.3)	(14.9, 0.7, 0.2)	> 90

Table 4.3: Comparison of the estimated range values for objects in the *line* sequence.

	# frames processed	Absolute range errors by Smith's approach (unit: feet)	Absolute range errors by our approach (unit: feet)
Truck 1	60	1.8	5.1
Truck 2	60	5.5	0.1
Truck 3	60	56.4	16.5
Truck 4	60	93.8	70.7
Truck 5	60	139.2	100.6

5. Summary and Conclusions

A vision system for runway recognition and obstacle detection has been developed for low-altitude flying aircraft/rotorcraft. This system provides a general approach to assist pilots in recognizing objects and detecting obstacles. Four main functions of the system are identified: first, the runway is recognized as the aircraft is approaching the airport; second, moving objects in the scene are detected; third, the moving objects are tracked in the image sequence and their motion parameters are estimated; and fourth, the positions of the stationary objects are estimated. Of the four functions identified above the algorithms for runway recognition and estimation of position for stationary objects were presented in this report. These estimates provide vital information to enhance the safety of navigation and to relieve pilots from tiring, monotonous control of aircraft as well. This research clearly demonstrates the potential for using a computer vision system for the automation of navigation.

Real world flight images have been used to demonstrate the feasibility of estimating object structures/motions from an aircraft equipped with a single camera and an inertial navigation system (INS). Several difficulties in the research area of computer vision can be overcome by utilizing the camera motion information provided by the INS. They are listed as follows:

1. As indicated in section 2.3, by incorporating the camera pose information, the computational complexity of model-based object recognition can be reduced from $O(mn)$ to $O(m)$, where m is the number of model features and n is the number of image features.

2. It is difficult to perform the *structure from motion* technique when the camera's moving path is not linear. With camera motion information, the path can be approximated as piece-wise linear, hence facilitating the estimation of object positions.

Now consider some possible future directions of research. First, throughout this work, the camera motion information is assumed available and reasonably accurate. No attempt has been made to estimate the camera motion parameters. It is desirable, however, to utilize the image analysis results as feedback to refine the camera motion information. It is also possible to use other sensors, such as forward looking infrared (FLIR), millimeter waves (MMW), and low-light-level-television (LLLTV), together with the visible light sensors in order to deal with different weather conditions. Such a multi-sensor system provides more accurate camera information and hence results in much better motion analysis. Second, this research has been dealing with monocular sequences of images. Using stereo sequences of images is expected to render better results. Hence, research topics involving stereo image analysis need further study. Third, real-time implementation is another important issue since accurate real-time response of the navigation system is essential to keep the aircraft from collision. And finally, the

interactions between the vision system and the control system have to be studied in depth in order to build a completely autonomous navigation system.

APPENDIX

CONSTRUCTION OF THE EPIPOLAR PLANES AND LINES

To construct a pencil of epipolar lines on an image plane, the epipolar planes have to be created first. The epipolar planes are defined as a pencil of planes all intersecting at the *linear* camera motion path. The number of epipolar planes thus determines the 3D resolution of the space. The epipolar lines are just the intersections of the epipolar planes and the image plane. Two pencils of epipolar lines created at different camera locations may not appear the same on the image planes depending on the orientations of the camera at the two time instances. However, the epipolar planes always remain the same as long as the camera is moving along a straight line. In this appendix, the procedures in constructing the epipolar planes and lines are described in detail.

A.1. Camera Path

Assuming the camera is moving along a straight line and given two camera center locations, $P_i = (X_i, Y_i, Z_i)$ and $P_{i+1} = (X_{i+1}, Y_{i+1}, Z_{i+1})$, at time instances i and $i+1$, the expression of the camera path L can be represented as

$$L: \quad \frac{X - X_i}{a_1} = \frac{Y - Y_i}{a_2} = \frac{Z - Z_i}{a_3} \tag{A.1}$$

where $\vec{a} \equiv (a_1, a_2, a_3)^T$ is the unit directional vector of L.

A.2. Epipolar Planes

The epipolar planes are defined to be perpendicular to the camera path L and equally spaced in terms of their orientation. To define a pencil of epipolar planes, the first step is to create a set of vectors $\{\vec{v}_j\}_{j=0...Q-1}$, all perpendicular to the camera path, where Q is the number of epipolar planes. Each epipolar plane is then determined by two vectors, i.e., $\vec{v}_j$ and $\vec{a}$. The procedures for deriving vector $\vec{v}_j$ are described as follows. First, according to the definition, we have

$$\vec{v}_j \perp \vec{a} \quad \Rightarrow \quad \vec{v}_j \cdot \vec{a} = 0 \tag{A.2}$$

Since any two adjacent epipolar planes have equal angular spacing, the angle between $\vec{v}_j$ and $\vec{v}_{j+1}$ is π/Q. Let vector $\vec{v}_0$ be parallel to the world XY plane and θ_j be the angle between vectors $\vec{v}_j$ and $\vec{v}_0$ as shown in Fig. A.1. $\vec{v}_0$ can thus be represented by $\vec{v}_0 = (a_1', a_2', 0)^T$, and since $\vec{v}_0 \perp \vec{a}$, vector $\vec{v}_0$ can be computed as:

$$\vec{v}_0 \perp \vec{a} \quad \Rightarrow \quad \vec{v}_0 \cdot \vec{a} = 0 \quad \Rightarrow \quad a_1 a_1' + a_2 a_2' = 0 \tag{A.3}$$

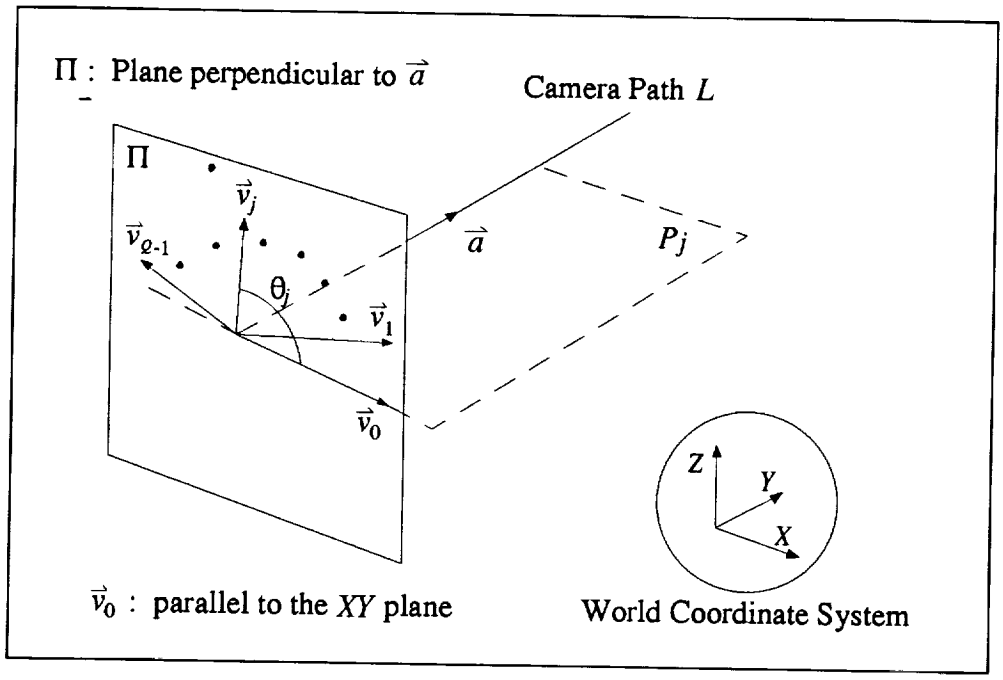


Fig. A.1: An epipolar plane P_j is determined by two vectors, $\vec{v}_j$ and $\vec{a}$.

$$\vec{v}_0 = \left(\frac{-a_2}{\sqrt{a_1^2 + a_2^2}}, \frac{a_1}{\sqrt{a_1^2 + a_2^2}}, 0 \right)^T \tag{A.4}$$

The following equation is therefore readily available:

$$\vec{v}_j \cdot \vec{v}_0 = \cos \theta_j, \; j = 0, .., Q-1, \text{ and } 0 \le \theta_j < \pi \tag{A.5}$$

Letting $\vec{v}_j = (v_1, v_2, v_3)^T$, three simultaneous equations can be obtain as $\vec{v}_j$ is a unit vector, $\vec{v}_0 \perp \vec{a}$, and $\vec{v}_j \cdot \vec{v}_0 = \cos\theta_j$:

$$\begin{cases} v_1^2 + v_2^2 + v_3^2 = 1 \\ a_1 v_1 + a_2 v_2 + a_3 v_3 = 0 \\ -a_2 v_1 + a_1 v_2 = \sqrt{a_1^2 + a_2^2}\cos\theta_j \end{cases} \tag{A.6}$$

The solution to $\vec{v}_j$ is:

$$\begin{cases} v_1 = \dfrac{1}{\sqrt{a_1^2 + a_2^2}}\left(a_2\cos\theta_j \pm a_1 a_3 \sin\theta_j\right) \\ v_2 = \dfrac{1}{\sqrt{a_1^2 + a_2^2}}\left(-a_1\cos\theta_j \pm a_2 a_3 \sin\theta_j\right) \\ v_2 = \mp\sqrt{a_1^2 + a_3^2}\sin\theta_j \end{cases} \tag{A.7}$$

For a special case where the camera is moving only in the Z direction, i.e., (a_1=a_2=0), vector $\vec{v}_j$ can be computed as $\vec{v}_j = (\cos\theta_j, \sin\theta_j, 0)^T$. As an epipolar plane P_j is determined by two vectors, $\vec{a}$ and $\vec{v}_j$, the normal vector $\vec{E}_j$ of P_j is determined by:

$$\vec{E}_j = \vec{v}_j \times \vec{a} = (a_3 v_2 - a_2 v_3,\ a_1 v_3 - a_3 v_1,\ a_2 v_1 - a_1 v_2)^T \tag{A.8}$$

Having the normal vector and the camera path, each epipolar plane is therefore fully specified.

A.3. Epipolar Lines

The pencil of epipolar lines are the intersections of the image plane I_i at time i and the pencil of epipolar planes. An epipolar line in the 3D space is specified by a unit directional vector $\vec{e}$ and a point which is well-known as the focus of expansion (FOE). The first step is to determine the location of the FOE at time i, which is described in Section A.3.1. And then, Section A.3.2. shows how to derive the unit directional vector.

A.3.1. Location of the FOE

The location of the FOE at time i is the intersection of the camera moving path and the image plane I_i. Since the moving direction of the camera is known, all that needs to derive is the image plane. This information can be obtained by deriving the normal unit vector of the image plane. The camera viewing direction, namely the direction of the optical axis, is the vector $(1, 0, 0)^T$ in the camera coordinate system. It is also the normal vector of the image plane. Since the rotation matrix from the world to the camera coordinate frame is expressed by $R_{ce} = R(\psi_{ce}, \theta_{ce}, \phi_{ce})$, the camera viewing direction at time i can be represented by the vector $\boldsymbol{c}_i$ with reference to the world coordinate system:

$$\vec{c}_i \equiv (c_1, c_2, c_3)^T = R_{ce}^{-1} \cdot (1, 0, 0)^T \qquad \text{(A.9)}$$

Knowing the camera moving direction $\vec{a}$ and the normal vector of the image plane, the location of the focus of expansion (FOE) with reference to the world coordinate system

can be computed. As shown in Fig. A.2, supposing the camera center is $(X_{ci}, Y_{ci}, Z_{ci})^T$ at time i, we have

$$\mathrm{FOE}_i \equiv (X_{fi}, Y_{fi}, Z_{fi})^T = (X_{ci}, Y_{ci}, Z_{ci})^T + f\frac{\vec{a}}{\vec{c}_i \cdot \vec{a}} \qquad \text{(A.10)}$$

Let the translation matrix from world to camera coordinate system be T_{ce}. The location of the FOE_i in the camera coordinate, (f, y_{fi}, z_{fi}), can be obtained:

$$(f, y_{fi}, z_{fi}) = R_{ce} \cdot ((X_{fi}, Y_{fi}, Z_{fi})^T - T_{ce}) \qquad \text{(A.11)}$$

A.3.2. The Normal Vector of the Epipolar Line

Let $\{e_{ij}\}_{j=0...Q-1}$, be the pencil of epipolar line on image plane I_i. All e_{ij}'s pass through the focus of expansion. Since the epipolar lines are the intersections of the epipolar planes and the image plane, e_{ij} will be perpendicular both to the camera viewing direction $\vec{c}_i$ and the normal vectors $\vec{E}_{ij}$ of the epipolar planes. Let the normal directional vector of e_{ij} be $\vec{e}_{ij}$, hence

$$\vec{e}_{ij} \perp \vec{c}_i,\ \vec{e}_{ij} \perp \vec{E}_{ij} \ \Rightarrow\ \vec{e}_{ij} = c_i \times \vec{E}_{ij} \qquad \text{(A.12)}$$

Let $\vec{e}_{ij} \equiv (0, e_y, e_z)^T$ be the unit directional vector of the epipolar line e_{ij} in the camera coordinate system, the epipolar line on the image plane I_i can be specified as a line

function, in which the epipolar line passes through the $FOE_i=(f, y_f, z_f)^T$ and has a unit directional vector $(e_y, e_z)^T$:

$$\frac{y-y_{fi}}{e_y}=\frac{z-z_{fi}}{e_z} \tag{A.13}$$

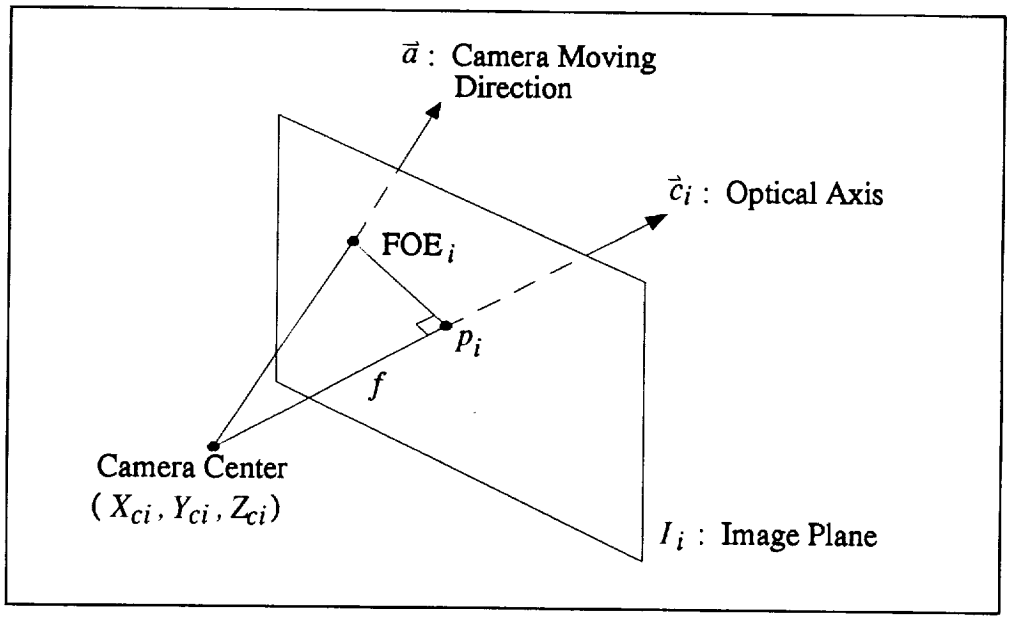


Fig. A.2: At time i, the image plane I_i passes through the world point p_i. The viewing direction is $\vec{c}_i$ with reference to the world coordinate system.

REFERENCES

Advisory Circular, 1980, Department of Transportation, Federal Aviation Administration, AC 150/5340-1E, Nov. 11.

Ayache, N. and O.D. Faugeras, 1986, "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 44–54.

Baker, H.H. and R.C. Bolles, 1989, "Generalizing Epipolar-Plane Image Analysis on the Spatiotemporal Surface," *Int. J. Computer Vision*, pp. 33–49.

Besl, P.J. and R.K. Jain, 1985, "Three-Dimensional Object Recognition," *ACM Computing Surveys*, vol. 17, no. 1, pp. 75–145.

Bhanu, B., B. Roberts, and J.C. Ming, 1989b, "Inertial Navigation Sensor Integrated Motion Analysis," *DARPA Image Understanding Workshop*, pp. 747–763.

Binford, T.O, 1982, "Survey of Model-Based Image Analysis Systems," *Int. J. Robotics Research*, vol. 1, no. 1, pp. 18–64.

Bolles, R.C., H.H. Baker, and D.H. Marimont, 1987, "Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion," *Int. J. Computer Vision*, pp. 7–55.

Brady, J.P, N. Nandhakumar and J.K. Aggarwal, 1989, "Recent Progress in Object Recognition from Range Data," *Image and Vision Computing*, pp. 295–307.

Broida T.J. and R. Chellappa, 1986, "Estimation of Object Motion Parameters from Noisy Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 1, pp. 90–99.

Canny, J., 1986, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698.

Cheng, V.H.L. and B. Sridhar, 1991, "Considerations for Automated Nap-of-the-Earth Rotorcraft Flight," *Journal of the American Helicopter Society*, vol. 36, no. 2, pp. 61 –69.

Chin, R.T. and C.R. Dyer, 1986, "Model-Based Recognition in Robot Vision," *ACM Computing Surveys*, vol. 18, no. 1, pp. 67–108.

Cox, I.J., 1993, "A Review of Statistical Data Association Techniques for Motion Correspondence," *Int. J. Computer Vision*, vol. 10, no. 1, pp. 53–66.

Crowley, J.L., P. Stelmaszyk, and C. Discours, 1988, "Measuring Image Flow by Tracking Edge-Lines," *Int. Conf. Computer Vision*, pp. 658–664.

Dhome, M., M. Richetin, J.-T. Lapresté, and G. Rives, 1989, "Determination of the Attitude of 3-D Objects from a Single Perspective View," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 12, pp. 1265–1278.

Etemadi, A., 1993, "Robust Segmentation of Edge Data," University of Surrey, Guild, United Kingdom.

Etemadi, A., J-P. Schmidt, G. Matas, J. Illingworth, and J. Kittler, 1993, "Low-Level Grouping of Straight Line Segments," Department of Electronic and Electrical Engineering, University of Surrey, Guildford, United Kingdom.

Foley, J.D., A. van Dam, S.K. Feiner, and J.F. Hughes, 1990, *Computer Graphics — Principles and Practice*, 2nd ed., Reading, Massachusetts, Addison-Wesley.

Grimson, W.E.L., 1990, *Object Recognition by Computer: The Role of Geometric Constraints*, Cambridge, Massachusetts, MIT Press.

Huttenlocher, D.P. and S. Ullman, 1987, "Object Recognition Using Alignment," *Int. Conf. Computer Vision*, pp. 102–111.

Lamdan, Y., J.T. Schwartz, and H.J. Wolfson, 1988, "Object Recognition by Affine Invariant Matching," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 335–344.

Lamdan, Y. and H.J. Wolfson, 1988, "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme," *Int. Conf. Computer Vision*, pp. 238–249.

Liou, S.-P. and R.C. Jain, 1989, "Motion Detection in Spatio-Temporal Space," *Computer Vision, Graphics, and Image Processing*, vol. 45, pp. 227–250.

Liu, Y. and T.S. Huang, 1988, "A Linear Algorithm for Motion Estimation Using Straight Line Correspondences," *Computer Vision, Graphics, and Image Processing*, vol. 44, pp. 35–57.

Longuet-Higgins, H.C., 1981, "A Computer Algorithm for Reconstructing a Scene from Two Projections," *Nature*, vol. 293, pp. 133–135.

Lucas, B. and T. Kanade, 1981, "An Iterative Image Registration Technique with an Application to Stereo Vision," *DARPA Image Understanding Workshop*, pp. 121–130.

Matthies, L., T. Kanade, and R. Szeliski, 1989, "Kalman Filter-Based Algorithms for Estimating Depth from Image Sequences," *Int. J. Computer Vision*, vol. 3, no. 3, pp. 209–236.

McIntosh, J.H. and K.M. Mutch, 1988, "Matching Straight Lines," *Computer Vision, Graphics, and Image Processing*, vol. 43, pp. 386–408.

Mitiche, A., O. Faugeras, and J.K. Aggarwal, 1989, "Counting Straight Lines," *Computer Vision, Graphics, and Image Processing*, vol. 47, pp. 353–360.

Morefield, C.L., 1977, "Application of 0-1 Integer Programming to Multitarget Tracking Problems," *IEEE Trans. Automatic Control*, vol. AC-22, no. 3, pp. 302–312.

Nelson, R.C., 1991, "Qualitative Detection of Motion by a Moving Observer," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 173–178.

Press, W.H., S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, 1992, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed., Cambridge University Press.

Roberts, B., B. Sridhar, and B. Bhanu, 1991, "Inertial Navigation Sensor Integrated Motion Analysis for Obstacle Detection," *Digital Avionics Systems Conference*, pp. 131–136.

Sawhney, H.S., J. Oleinsis, and A.R. Hanson, 1993, "Image Description and 3-D Reconstruction from Image Trajectories of Rotational Motion," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 885–898.

Silberberg, T.M, L.S. Davis, and D.A. Harwood, 1984, "An Iterative Hough Procedure for Three-Dimensional Object Recognition," *Pattern Recognition*, vol. 17, no. 6, pp. 612–629.

Smith, P. and G. Beuchler, 1975, "A Branching Algorithm for Discriminating and Tracking Multiple Objects," *IEEE Trans. Automatic Control*, vol. AC-20, pp. 101–104.

Smith, P.N., 1990, *NASA Image Data Base User's Guide*, NASA Ames Research Center, Moffett Field, CA., Version 1.0.

Smith, P.N., B. Sridhar, and B. Hussien, 1992, "Vision-Based Range Estimation Using Helicopter Flight Data," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 202–208.

Sommer, H.J., Professor of Mechanical Engineering, The Pennsylvania State University, University Park, Pennsylvania, private disscusion, 1994.

Spetsakis, M.E. and J. Aloimonos, 1987, "Closed Form Solution to the Structure from Motion Problem from Line Correspondences," *National Conf. Artificial Intelligence*, pp. 738–743.

Sridhar, B. and A.V. Phatak, 1988, "Simulation and Analysis of Image-Based Navigation System for Rotorcraft Low-Altitude Flight," *AHS National Specialists' Meeting, Automation Applications of Rotorcraft.*

Sridhar, B., V.H.L. Cheng, and A.V. Phatak, 1989, "Kalman Filter Based Range Estimation for Autonomous Navigation Using Imaging Sensors," *IFAC Symposium on Automatic Control in Aerospace.*

Tang, Y.-L., S. Devadiga, and R. Kasturi, 1993, "A Model-Based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images," *Proc. Augmented Visual Display (AVID) Research Workshop*, NASA Conference Publication 10128, pp. 313–328. 94N25490

Tang, Y.-L., S. Devadiga, and R. Kasturi, 1994, "Model-Based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images," *IS&T/SPIE Int. Symposium on Electronic Imaging, Image and Video Processing II*, pp. 320–330.

Tang, Y.-L. and R. Kasturi, 1994, "Accurate Estimation of Object Location in an Image Sequence Acquired Using a Moving Camera," *NASA Goddard Conf. Space Applications of Artificial Intelligence*, pp. 147–157.

Therrien, C.W., 1989, *Decision, Estimation and Classification: An Introduction to Pattern Recognition and Related Topics*, New York, Wiley.

Thompson, D.W. and J.L. Mundy, 1987, "Three-Dimensional Model Matching from an Unconstrained Viewpoint," *IEEE Int. Conf. Robotics Automation*, pp. 208–220.

Tsai, R.Y. and T.S. Huang, 1984, "Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces," *IEEE. Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 1, pp. 13–27.

Tseng, G.J. and A.K. Sood, 1989, "Analysis of Long Image Sequence for Structure and Motion Estimation," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 19, no. 6, pp. 1511–1526.

Turney, J.L., T.N. Mudge, and R.A. Volz, 1985, "Recognizing Partially Occluded Parts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 7, no. 4, pp. 410–421.

Weng, J., T.S. Huang, and N. Ahuja, 1992, "Motion and Structure from Line Correspondences: Closed-Form Solution, Uniqueness, and Optimization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 3, pp. 318–336.

Wolfson, H.J., 1990, "Model Based Object Recognition by Geometric Hashing," *First European Conf. Computer Vision*, pp. 526–536.