The Pennsylvania State University
The Graduate School
Department of Computer Science and Engineering

IMAGE SEQUENCE ANALYSIS FOR OBJECT DETECTION AND SEGMENTATION

A Thesis in
Computer Science and Engineering
by
Tarak L. Gandhi

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2000

We approve the thesis of Tarak L. Gandhi.

Date of Signature

Rangachar Kasturi
Professor of Computer Science and Engineering
Thesis Adviser
Chair of Committee

Octavia I. Camps
Associate Professor of Computer Science and Engineering
and Electrical Engineering

Lee D. Coraor
Associate Professor of Computer Science and Engineering

Rajeev Sharma
Assistant Professor of Computer Science and Engineering

David J. Miller
Assistant Professor of Electrical Engineering

Dale A. Miller
Professor of Computer Science and Engineering
Head of the Department of Computer Science and Engineering

Abstract

A sequence of images contains more information than a single image. Due to this reason, image sequence analysis has been used in computer vision for quite some time. In particular, a sequence of images is useful for object detection when the camera moves relative to the object. Due to the relative motion, objects at different distances from the camera have different image motion. Using this property, one can obtain information on structure of the 3-D scene as well as the relative motion between the camera and the scene. Furthermore, the individual images are corrupted by camera noise. Use of a sequence of images enables suppression of this noise for reliable detection of low contrast objects. In this research, image sequence analysis is used for detection of objects in 3-D space as well as on planar surfaces. The work would be useful for detecting obstacles in the flight path of an aircraft. The research also explores the use of these principles for segmenting scene text objects in video sequences with a relative motion between the camera and the scene.

A computer vision based system that can aid the pilot to detect obstacles in the flight path of an aircraft can be useful for avoiding collisions. Such a system would also be useful for development of a Synthetic Vision System (SVS) proposed for use in a High Speed Civil Transport (HSCT) aircraft with limited cockpit visibility. For this purpose, we had implemented a number of algorithms to detect airborne obstacles using image sequences obtained from a camera mounted on an aircraft. The performance of these algorithms was characterized in presence of camera noise using theoretical and experimental methods. Since the performance degrades in the presence of background clutter, a special approach to address the problem of hazard detection in presence of clutter was studied. This approach uses the differences in the behavior of translation and expansion of image features corresponding to the objects on a collision course and the background clutter. Algorithm fusion for combining different algorithms to overcome their individual limitations was also studied. In addition to this work on detecting objects on collision course, algorithms for detecting objects crossing the aircraft were designed and implemented on a real-time system.

Previous to our work on airborne object detection, we had worked on object detection on runways in presence of extraneous features, such as tire-marks. The work done on this topic is briefly described, citing references for details. The concepts used in

this work were applied to another application: extraction of scene text in video image sequences with a relative motion between the camera and the scene. Assuming that the scene text generally occur on planar surfaces, the planar motion model is used to segment the scene into planar surfaces and determine their parameters. The clutter features which do not satisfy the planar motion model are removed as outliers. The parameters of each segmented surface are used to correct the perspective distortion of the surface. Each surface can then be analyzed separately for detecting and recognizing text objects, with perspective correction potentially improving the recognition performance.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I am grateful and indebted to my thesis advisor, Prof. Rangachar Kasturi for the guidance, patience, and encouragement he has shown me during my time here at Penn State. I am also grateful and indebted to Prof. Octavia Camps for inspiration and enlightening discussions on a wide variety of topics. I am especially indebted for the financial support which they have provided to me over the years. I thank my committee members, Prof. Lee Coraor, Prof. Rajeev Sharma, and Prof. David J. Miller for their insightful commentary on my work.

I would also like to thank Dr. Jeffrey McCandless, Dr. Barbara Sweet, Dr. Al Ahumada, and Dr. Jeffrey Mulligan for the useful information and suggestions during my summer visit to the NASA Ames Research Center. I am thankful to my friends, Dr. Sadashiva Devadiga and Mau-Tsuen Yang who have worked on the NASA project with me. I also thank Dr. Ullas Gargi and Sameer Antani for helping me during my stay.

I am greatful to my uncle Ashok Gandhi and aunt Smita Gandhi for their guidance and support without which I could not have acomplished this. Finally, I am most greatful to my parents for all they have done for me through my life.

Chapter 1

Introduction

Image sequence analysis has been widely used in computer vision. This thesis describes the use of image sequences for detection of objects in 3-D space as well as on planar surfaces. This work is useful for detection of obstacles in the flight path of an aircraft using on-board video camera images. Another application of these principles for segmenting text objects in video image sequences is also explored in this thesis.

Continued advances in the fields of image processing and computer vision have raised interest in their suitability to aid pilots to detect possible obstacles in their flight paths. For the last few years, NASA has been exploring the use of image sequences for detecting obstacles in the flight path of an aircraft. NASA Langley Research Center supported a project to enable pilots to 'see through fog' using Passive Milli-Meter Wave (PMMW) images of low resolution. For this project, Tang and Devadiga [36] from our group had developed methods to locate the runway and detect obstacles on and outside the runway. The resulting output can be used by the pilots to decide whether to land or not.

Obstacle detection is also possible with visible-light image sequences. In the design of a High Speed Civil Transport (HSCT) aircraft with a limited cockpit visibility, NASA has proposed a Synthetic Vision System (SVS) in which high resolution video images would be obtained using cameras mounted on the aircraft. These images can be used to detect obstacles in the flight path to warn the pilots and avoid collisions. For aircraft operations, both airborne obstacles, as well as the obstacles on the runway surface should be detected.

Algorithms for detection of airborne objects from images are abundant in the published literature. A systematic performance characterization of a number of target detection algorithms was performed by using image degradation models for digital cameras. It was observed that the algorithms that were studied have a good performance on images which do not have background clutter. However, the performance degrades severely when background clutter is present. Thus, the goal of this work has been to design algorithms which perform better in cluttered background environments, with low

probabilities of false alarms and mis-detections and capability of target detection early enough to avoid a possible collision. To achieve this goal, a special approach was used to discriminate hazardous objects on collision course from the background clutter. Algorithm fusion was studied for combining different algorithms in a statistical framework, to overcome their individual limitations. The performance of the fused algorithm was found to be better than the individual algorithms under appropriate conditions.

Previous to the work on airborne object detection, we have worked on detection of objects on the runway in presence of extraneous features, such as tire-marks. Similar to the work by Sull and Sridhar [56], it was assumed [23, 35] that the runway is a planar surface and the obstacles lie above the runway and/or are in motion relative to the runway. The image motion of the runway was compensated using the known parameters of camera motion and runway plane. The image features with residual motion were then classified as obstacles. Devadiga [20] used a planar motion model for the runway to eliminate the need for knowing camera motion and plane parameters. The features which do not satisfy this planar model are classified as obstacles. In case the scene contains a number of planar surfaces, a recursive motion-based segmentation algorithm is used to separate these surfaces.

This approach of segmenting a scene into a number of planar surfaces can also be useful for other applications. We have studied the feasibility of using this motion segmentation procedure for a project in collaboration with DoD, on detection and localization of text in video sequences. Scene text in video sequences generally lie on planar surfaces. When the camera undergoes motion relative to the surface, the image motion of the surface can be modeled using the planar motion model. This model is used for detecting and segmenting planar surfaces expected to contain scene text. The clutter features which do not satisfy the planar motion model are removed as outliers. The parameters of the planar surfaces are estimated and used to correct the perspective distortion of the surfaces. Each segmented surface can then be analyzed separately for detecting text objects. Perspective correction applied to the surfaces is expected to improve the recognition performance.

The thesis dissertation is organized as follows: Chapter 2 describes the basic, well-known algorithms used for detection of airborne obstacles. These algorithms were tested on real image sequences provided by NASA. In Chapter 3, the performance of these algorithms is experimentally characterized using the approach described by Kanungo et al. [33]. The theoretical characterization of the algorithms' performance is

described in Chapter 4, and the experimental performance is compared with the theoretical performance.

The main contribution of the research for the detection of hazardous objects is described in the next two chapters. A special approach is proposed for discrimination of objects on collision or near-collision course from background clutter. This approach is described in Chapter 5 where differences in the behavior of translation and expansion in the image are used to separate hazardous objects from clutter. Chapter 6 describes the Bayesian methodology used for combining detection algorithms in a statistical framework. Performance of fused algorithm is compared with that of the individual algorithms.

In addition to hazardous objects, it is also useful to detect and track objects crossing in front of the aircraft. A real-time system using pipelined image processing hardware was designed for this purpose. Chapter 7 describes the image processing operations which are performed by the pipelined hardware, and the tracking operations performed on the host machine to form a complete real-time system.

Chapter 8 briefly describes the work done by Gandhi et al. [23] and Devadiga [20] for detection of obstacles on the runway. Appropriate references are cited for details of the work. In Chapter 9, the principle of using planar motion for segmentation is applied to the application of separating text objects which would usually lie on planar surfaces from background clutter. Chapter 10 describes the sensitivity of various steps in the procedure.

Chapter 11 concludes the thesis and explores avenues for future work.

Chapter 2

Object Detection Algorithms

This chapter describes the algorithms that were implemented to detect airborne obstacles in the flight path of a flying aircraft. Statistical theory used for target detection is first described, followed by a number of basic steps useful for removing background clutter, amplifying the signal to noise ratio, and detecting objects having different sizes and velocities. Results obtained by using real image sequences are also described.

2.1 Background

NASA's need for enhanced capabilities in obstacle detection using image processing requires robust, reliable and fast techniques. These techniques should provide a high probability of detection while maintaining a low probability of false alarm in noisy, cluttered images of possible targets, exhibiting a wide range of complexities. The size of the image target can be quite small, from sub-pixel to a few pixels in size. As an example, consider a Cessna aircraft that has a length and wing-span of approximately $9\,m$ $(30\,ft)$ and the fuselage diameter of approximately $1.2\,m$ $(4\,ft)$ [64]. The detection algorithm must be capable of detecting this small target at least 25 seconds prior to a possible collision to allow for corrective actions by the pilot. Assuming that both the aircraft are traveling at $125\,m/s$ (250 knots), their relative velocity can be as high as $250\,m/s$ (500 knots). In such case, they would be $6.25\,km$ (3.5 nautical miles) apart 25 seconds before collision. Using a camera with a resolution of 60 pixels per degree, the image size of the aircraft is 5.0×0.7 pixels from a side view, but only 0.7×0.7 pixels from a front view. Furthermore, the detection algorithm must report such targets in a timely fashion, imposing severe constraints on their execution time. Finally, the system must not only work well under the controlled conditions found in a laboratory and with data closely matching the hypothesis used in the design process, but it must be insensitive – i.e., must be *robust* – to data uncertainty due to various sources, including sensor noise, weather conditions, and cluttered backgrounds.

Extensive work has been done on the problem of target detection. When the signal to noise ratio is low, it is preferable to use the 'track before detect' approach. In this

approach, an object is tracked over multiple frames before making a hard decision on the presence or absence of a target. The simplest way to integrate the input images over multiple frames is by temporally averaging them. When the image motion of the object is very small, as in the case of an object being exactly on a collision course [38], this happens to be the best approach. However, if the object has a significant image motion, other approaches are needed. Nishiguchi et al. [47] proposed the use of a recursive algorithm to integrate multiple frames while accounting for small object motion. A dynamic programming approach was used by Barniv [8] and Arnold et al. [4] to detect moving objects of small size. The theoretical performance of this approach was characterized by Tonissen and Evans [61].

The above algorithms perform well when the background is uniform. However, in real situations the hazardous object should be detected not only against uniform background, but also against backgrounds such as clouds, ground or water. The features introduced due to a non-uniform background which interfere with object detection are collectively known as clutter. Thus, the objective of the detection algorithms is to successfully detect the hazardous object, without giving unnecessary false alarms from clutter. Subtraction of consecutive images is often used to remove stationary clutter. However, an object on a collision course could be nearly stationary in the image [38]. Hence, this method is not useful for our application, since it could remove the object as well. Alternatively, morphological filtering [17] removes objects of large size, usually corresponding to clutter while retaining the objects of small size. This approach is useful in removing large clutter, such as clouds. But it does not remove small-sized clutter.

2.2 Statistical decision theory for target detection

Statistical decision theory [37, 49] can be used to design optimal or near-optimal detection algorithms, as well as to characterize their performance. The input to the algorithm is a sequence of images, each composed of a large number of individual pixels. These pixels are degraded by various sources, such as atmosphere, lens, and camera noise. Based on the statistical behavior of this degradation, the image pixels can be combined in space and time, to make statistically optimal decision about the presence or absence of a target. For making these decisions, probabilistic models of the signal and its degradation can be used.

Let H_0 and H_1 denote the hypotheses that the target is absent or present, respectively, and $P(H_0)$ and $P(H_1)$ denote their respective prior probabilities. Let z represent

the vector of observations from which one is supposed to determine the presence or absence of a target. By Bayes' rule, the posterior probabilities are given by:

$$P(H_1|z) = \frac{p(z|H_1)P(H_1)}{p(z)}\,,\; P(H_0|z) = \frac{p(z|H_0)p(H_0)}{p(z)} \tag{2.1}$$

The ratio of these probabilities is given by:

$$\frac{P(H_1|z)}{P(H_0|z)} = \frac{P(H_1)p(z|H_1)}{P(H_0)p(z|H_0)} = \frac{P(H_1)}{P(H_0)}L_H(z) \tag{2.2}$$

where $L_H(z)$ proportional to the ratio of the probabilities is called the likelihood ratio.

When the algorithm reports a target even where there actually is none, it is called a false alarm, whereas when it does not report an existing target, it is called a mis-detection. The performance of a detection algorithm is characterized in terms of false alarms and mis-detections. According to the Neyman Pearson criterion [37, 49], the number of mis-detections for a given rate of false alarms can be minimized by thresholding the likelihood ratio $L_H(z)$. The threshold is a function of the required rate of false alarms. In place of the likelihood ratio, any of its monotonic function (such as the logarithm) can be used. Such a function is called a discriminant function.

To decrease the probabilities of false alarms and mis-detections, one can integrate observations spatially or temporally. Let the N elements $z_1, z_2 \ldots z_N$ of z be independent observations. The likelihood ratio and its logarithm (log likelihood ratio) are given by:

$$L_H(z) = \frac{p(z_1, z_2 \ldots z_N|H_1)}{p(z_1, z_2 \ldots z_N|H_0)} = \prod_{i=1}^{N} \frac{p(z_i|H_1)}{p(z_i|H_0)} \tag{2.3}$$

$$l(z) = \log L_H(z) = \sum_{i=1}^{N} [\log p(z_i|H_1) - \log p(z_i|H_0)] \tag{2.4}$$

In the case of z_i's having normal distributions in absence and presence of target, such that their probability density functions are:

$$p(z_i|H_0) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{z_i^2}{2\sigma^2}\right]\,,\; p(z_i|H_1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[\frac{(z_i-\mu)^2}{2\sigma^2}\right] \tag{2.5}$$

The log likelihood ratio is given by:

$$l(z) = \log L_H(z) = \sum_{i=1}^{N} \frac{-(z_i - \mu)^2 + z_i^2}{2\sigma^2} = \frac{\mu}{\sigma^2}\left[\sum_{i=1}^{N} z_i\right] - \frac{N\mu^2}{2\sigma^2} \tag{2.6}$$

This is a monotonic function of $\sum z_i$. Hence, thresholding the sum (or mean) of the observations yields an optimal detector. Since sum and mean are linear functions, they are also normally distributed.

Consider a discriminant function which is normally distributed in absence and presence of target as $N(\mu_0, \sigma_0^2)$ and $N(\mu_1, \sigma_1^2)$, respectively with equal variances $\sigma_0^2 = \sigma_1^2$ but unequal means μ_0 and μ_1. If this function is thresholded to obtain a particular false alarm rate, it can be shown that the corresponding mis-detection rate is a function of its Signal to Noise Ratio (SNR) given by $(\mu_1 - \mu_0)/\sigma_0$. Hence in this case, the performance in terms of false alarm and mis-detection rates is determined by the SNR.

If N independent normal observations are made, their sum is distributed as $N(0, N\sigma^2)$ in absence of target, and $N(N\mu, N\sigma^2)$ in presence of target. Hence, the SNR is given by $N\mu/\sqrt{N\sigma^2} = \sqrt{N}\mu/\sigma$ – i.e., amplified by a factor of $\sqrt{N}$. In other words, a signal with SNR of $S/\sqrt{N}$ integrated over N frames could yield the same rate of false alarms and mis-detections as one would get using a single observation with SNR of S. Hence, the SNR *required* for detection *reduces* by $\sqrt{N}$ when N frames are added. The same result is true for averaging of N frames, since the signal as well as the noise would be reduced by a factor of N.

2.3 Pre-processing

Before any other algorithms can be applied, pre-processing should be performed on the input images to suppress the background. The following approaches were used for pre-processing the images.

2.3.1 Low-stop filter

In the case of an image with little or no clutter, a low-stop filter which subtracts from every pixel, the local average of the neighborhood of that pixel effectively suppresses the background intensity. This filter can be implemented by convolving the image with a 2-D mask corresponding to the filter. Since the amount of computation increases with the mask size, a small sized mask was used in conjunction with the pyramid approach described in Section 2.4 to simulate the effect of a large sized mask.

2.3.2 Morphological filter

If the background has significant clutter, the low-stop filter is not as effective for removing it. A morphological filter [17] can remove large sized features (usually clutter), while retaining small sized features (usually targets).

The gray-scale morphological operations of dilation ($\oplus$) and erosion ($\ominus$) are defined as:

$$(f \oplus m)(x,y) = \max_{(x',y')\in m} \{f(x-x', y-y') + m(x',y')\} \tag{2.7}$$

$$(f \ominus m)(x,y) = \min_{(x',y')\in m} \{f(x+x', y+y') - m(x',y')\} \tag{2.8}$$

where m is the mask using which the morphological operation is performed, and f is the image which is considered to have a default value of $-\infty$ outside its domain. Morphological closing and opening can be defined using the above operations as:

$$(f \bullet m) = (f \oplus m) \ominus m \geq f \tag{2.9}$$

$$(f \circ m) = (f \ominus m) \oplus m \leq f \tag{2.10}$$

A difference between the original image and its morphological opening, known as the top-hat transform outputs small-sized positive targets – i.e., bright targets in dark background. On the other hand, the difference between the morphological closing and the original image, known as the bottom-hat transform outputs negative targets – i.e., dark targets in bright background. Each of these images are non-negative, and can be separately used to detect targets.

A single mask for these morphological operations gives undesirable outputs for jagged boundaries of large features. Hence, horizontal mask m_x and vertical mask m_y were used separately as proposed by [17]. These masks are of length 5 with origin at the center of the mask, with all the pixels having the default value of zero. The outputs are given by:

$$F_+ = F - \max\{F \circ m_x, F \circ m_y\} \tag{2.11}$$

$$F_- = -F + \min\{F \bullet m_x, F \bullet m_y\} \tag{2.12}$$

2.4 Spatial integration

To detect targets of a number of different sizes and velocities, and to amplify the SNR, the target pixels in a given image can be integrated by forming an image pyramid. For this purpose, the following basic operations are used:

1. Low-pass filter (LP or $\overline{LP}$): Convolves the image in x and y directions with the masks $m_x = m_y = [1, 3, (3), 1]/8$, or their mirror images. The parentheses denote the origins of the masks.

$$\begin{aligned} f_{LP}(x,y) &= \sum_{x'}\sum_{y'} f(x-x', y-y')\, m_x(x')\, m_y(y') \\ f_{\overline{LP}}(x,y) &= \sum_{x'}\sum_{y'} f(x+x', y+y')\, m_x(x')\, m_y(y') \end{aligned} \tag{2.13}$$

2. Down-sampler (DS): Selects even numbered pixels in the input image to give an image with half the resolution.

$$f_{DS}(x,y) = f(2x, 2y) \tag{2.14}$$

3. Up-sampler (US): Forms the output image by putting the input image pixels in even numbered positions, and zeros in odd numbered positions. The image is scaled by 2 to maintain the image intensity during subsequent low-pass filter step.

$$f_{US}(x,y) = 2f(x/2, y/2) \text{ when } x,\ y \text{ are even; 0 otherwise} \tag{2.15}$$

These steps are combined to form two types of operations:

1. Low-pass down-sample operation ($LP \rightarrow DS$): Decreases the resolution of the image by two. Low-pass filter prevents aliasing of high frequencies in the image by suppressing them.

2. Up-sample low-pass operation ($US \rightarrow \overline{LP}$): Increases the resolution of the image by two. Low-pass filter smoothes the output of the up-sampler (containing zeros at odd pixels) to produce the effect of interpolation. In this case, the mirror image masks are used to compensate the asymmetry in the masks.

The above operations can be used to combine pyramid formation with low stop or morphological filtering by using the system shown in Figure 2.1. Images $pyr[i]$ are

formed by successively applying low-pass and down-sample operations on the original image. These images can be directly used as inputs to the morphological filter to detect targets at different resolutions. Images $pyr'[i]$ are formed by successively applying up-sample and low-pass operations to the lowest resolution image $pyr[n]$, where n is the number of pyramid levels. These operations remove the high frequency components of the original image. Low-stop filtered images are given by $ls[i] = pyr[i] - pyr'[i]$, and retain only the higher frequency components not subtracted out by $pyr'[i]$.

In this way, a hierarchy of images, each with half the resolution of the previous one is formed. The size as well as the velocity of the object in the image scales as the resolution is lowered. There is a particular resolution at which the object occupies no more than 2 to 3 pixels in length and width, which would be optimal for detection of the object.

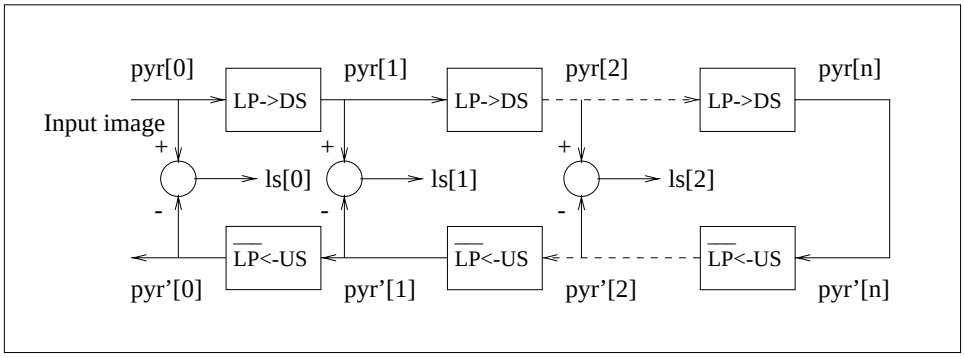


Fig. 2.1. Spatial integration using pyramid construction: LP or $\overline{LP}$: low-pass filtering with original mask or its mirror image, DS: down-sampling, US: up-sampling. The pyramid images at stage $i = 0 \ldots n$ are denoted by $pyr[i]$. Low-stop filtered images are obtained the by subtracting the corresponding up-sampled pyramid outputs $pyr'[i]$ from $pyr[i]$ and are denoted by $ls[i]$.

2.5 Temporal integration

As shown in Section 2.2, integration of pixels corresponding to a target results in amplification of the target SNR, and increased reliability of detection. Depending on the image motion of the target, the following approaches can be used for integration of target pixels over a number of image frames. The performance of these approaches is characterized experimentally and theoretically in Chapters 3 and 4, respectively.

2.5.1 Recursive temporal averaging

In the case of objects on a collision course [38] the image motion is very small. Hence, pixel wise temporal averaging of a sequence of images would improve the detection performance. However, direct use of temporal averaging results in infinite memory. To give a higher weight to more recent observations, a recursive filter can be used. The output $F(k)$ at time k for any pixel is recursively obtained from the input $f(k)$ at the same pixel using the following steps:

1. Initialization: $F(0) = 0$
2. Recursion: $F(k) = f(k) + \alpha F(k-1)$

where α is a forgetting factor between 0 (full forgetting) and 1 (no forgetting).

2.5.2 Dynamic programming

In the case of moving targets, the temporal averaging filter does not improve the detection. A dynamic programming algorithm [4] is more effective in detection of moving targets. The algorithm is based on shifting the images before averaging them so as to align the target to be detected. Since the velocity of the target could be arbitrary, the velocity space (u, v) is discretized within the range of possible target velocities. A set of intermediate images F, each corresponding to a particular velocity (u, v), are created recursively using the following steps:

1. Initialization: For all pixels (x, y) and all velocities (u, v), set

$$F(x, y; u, v; 0) = 0 \tag{2.16}$$

2. Recursion: At time k, set

$$F(x,y;u,v;k) = f(x,y;k) + \alpha \max_{(x',y')\in Q} F(x-u-x', y-v-y'; u, v; k-1) \quad (2.17)$$

where

$$Q = \{(x',y') | x'_{min} \leq x' \leq x'_{max}\ ,\ y'_{min} \leq y' \leq y'_{max}\} \quad (2.18)$$

3. Termination: At time K, take

$$F_{max}(x,y;K) = \max_{(u,v)\in P} F(x,y;u,v;K) \quad (2.19)$$

where

$$P = \{(u,v) | u_{min} \leq u \leq u_{max}\ ,\ v_{min} \leq v \leq v_{max}\} \quad (2.20)$$

The maximum operation in the recursion step is performed using the set Q, which ensures that the targets with velocities which do not fall on the grid are not missed. The set of discretized velocities denoted by P determines the range of target velocities that can be detected by the algorithm. The final maximum in the termination step combines the targets corresponding to all the velocities. The number of elements in P and Q are denoted by p and q, respectively.

In the recursion step, a maximum is taken over q pixels. If these pixels are all noise pixels, they are more likely to give a false alarm if q is large. Thus, the rate of false alarms increases with q. To get better performance, a smallest possible q should be used. The value of $q = 4$ has been used in our experiments corresponding to a 2×2 neighborhood, given by:

$$Q = \{(0,0), (-1,0), (0,-1), (-1,-1)\} \quad (2.21)$$

This ensures that the targets having fractional velocities are not missed. The asymmetry in this neighborhood is compensated by choosing $u_{min} = u_{max} - 1$ and $v_{min} = v_{max} - 1$. For the case of $u_{max} = v_{max} = 1$, $p = 4$ and P is given by:

$$P = \{(0,0), (1,0), (0,1), (1,1)\} \quad (2.22)$$

The algorithm then detects targets with a maximum velocity of 1 pixel per frame. However, when spatial integration is performed prior to dynamic programming, targets with larger sizes and velocities can be detected.

On the other hand, if $P = Q = \{(0,0)\}$ so that $p = q = 1$, the algorithm reduces to recursive temporal averaging, which gives the best performance for stationary targets. However, the performance of temporal averaging sharply degrades if the target is moving, whereas that of dynamic programming algorithm does not.

The output of the dynamic programming algorithm is an image, with large values at positions where the target strength is high. However, the pixels in the neighborhood of the target will also have a significantly large value. This can be resolved by using non-maximal suppression, where the output is smoothed using a Gaussian filter with $\sigma = 1.0$, and each pixel which is not a local maximum in its 3×3 region is set to zero. After this, only the pixels which are local maxima remain, which can be thresholded to obtain the target locations.

It should be noted that separate processing should be performed if the targets are negative – i.e., dark targets on a bright background. In the case of low-stop pre-processing, this is done by using the negative of the pre-processed image, whereas in the case of morphological pre-processing, both original minus open and closed minus original images are processed separately.

2.6 Composite system

The above mentioned algorithms have been combined to form a composite system for target detection. The steps that form this composite system are:

1. Temporal Averaging: This step is performed first in the case of objects in a uniform background, having a very small image motion, such as those on a collision or near-collision course. In such a case, temporal averaging improves the SNR and reduces the processing rate required for subsequent steps.

2. Pyramid construction with low-stop or morphological filtering: In this step, a pyramid is constructed to accommodate different sizes and velocities of objects. For pre-processing the images, low-stop or morphological filtering is performed at each pyramid level to remove background intensity. Low-stop filtering is more effective in low clutter situations, whereas morphological filtering [17] is more effective in suppressing background clutter due to clouds and ground.

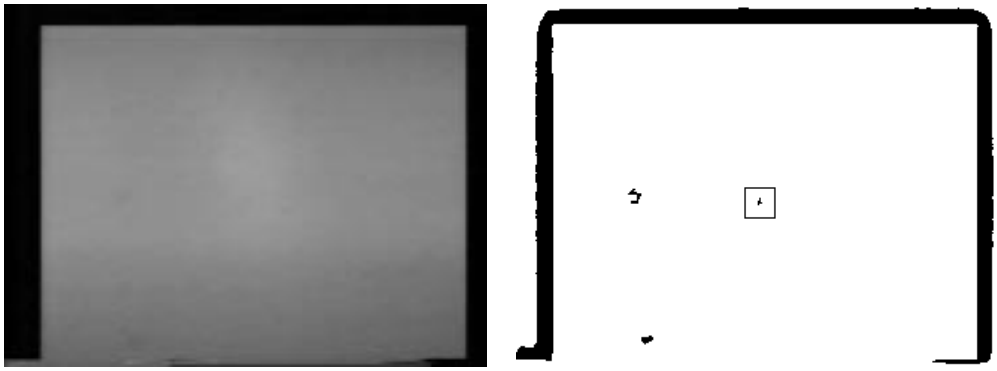
3. Dynamic Programming: A dynamic programming algorithm [4] is performed on pre-processed frames to integrate the signal over a number of frames by taking the target motion into consideration. Non-maximal suppression and thresholding are then performed on the output.

It should be noted that one or more of these steps can be bypassed so that any of the basic algorithms described above can be tested individually using the same system.

2.7 Results using analog camera

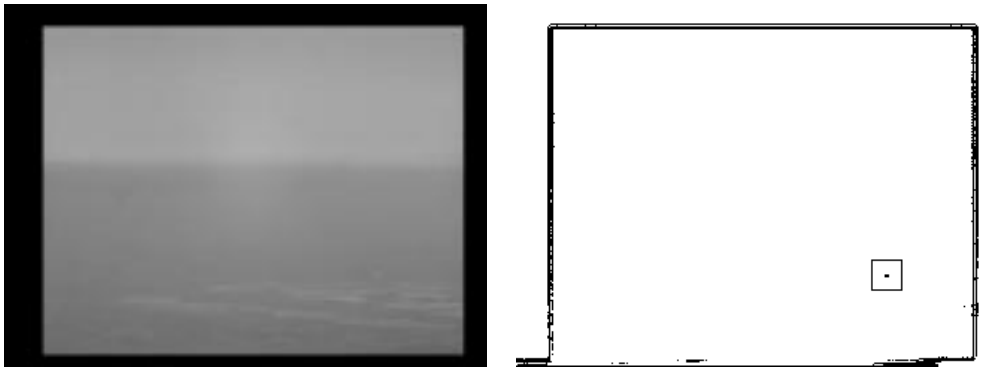
The above target detection algorithms were applied to real image sequences obtained from NASA. Figure 2.2 (a) shows an image from the sequence with the target aircraft flying away from the host aircraft. The sequence can be played in reverse to simulate the aircraft on a collision course. Since the aircraft on a collision course have a small image motion, temporal averaging was the optimal detection algorithm in this particular case. The aircraft was at a distance of approximately 4 nautical miles ($7.4\,km$), and was barely visible in a single image. Low-stop filter was applied before temporal averaging to remove the near-uniform background. After temporally averaging and thresholding, the aircraft was detected as shown in Figure 2.2 (b). Dynamic programming algorithm was performed on a sequence of images (after applying low-stop filter as pre-processing) in which an aircraft was flying from right to left across the image as shown in Figure 2.3 (a). Dynamic programming algorithm detected the aircraft with a low rate of false alarms. However, the target was dilated by the use of this algorithm. Clutter removal using morphological filtering was also explored. Figure 2.4 (a) shows a small aircraft flying in the middle-right part of the image. The image was actually obtained by averaging 10 motion compensated images from an image sequence, in which an aircraft was flying on the collision course. Application of morphological filter removed most of the clutter due to edges of large-sized features. This aircraft which was on a collision course, was retained. However, other small-sized features were also retained, resulting in a number of false alarms. The result is shown in Figure 2.4 (b).

Chapter 3 presents a systematic performance characterization for temporal averaging as well as dynamic programming using statistical image models for digital cameras. It was observed that the algorithms performed very well when the background was clear. However, the performance degraded severely in presence of clutter. In the case of cluttered images, pre-processing using morphological filter worked better than that using low-stop filter. Most of the clutter was removed, but small sized clutter, especially due



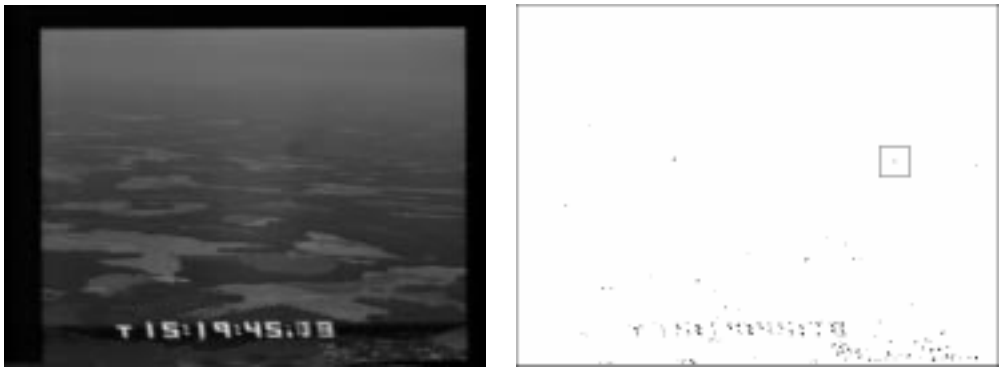
(a) (b)

Fig. 2.2. Target detection using temporal averaging: (a) Original image with a distant contracting target at 4 nautical miles. The target is approximately in the middle of the image. However, due to degradation of image quality, it is very faint. (b) Detection of the distant contracting target using low-stop filter pre-processing, temporal averaging and thresholding. A false alarm in the mid-left area is most likely due to a smudge on the camera.



(a) (b)

Fig. 2.3. Target detection using dynamic programming: (a) Original image frame with a translating target. (b) Location of the detected target using dynamic programming (following a low-stop pre-processing step).



(a) (b)

Fig. 2.4. Detection using morphological processing: (a) An average of ten motion-compensated frames of an image sequence. The aircraft is in the middle-right part of the image. (b) Detection using morphological filter. False alarms due to other features are also seen.

to specular reflection from water remained. Finally it was observed that the number of false alarms after applying the algorithm, in general, was reduced but still significant.

2.8 Data collection using digital camera

The real data that was used for our previous work was captured using an analog camera and recorded using NTSC video, thus containing additional noise that should not be present when a digital camera is used on the actual flight. Hence, the performance of the algorithms should be characterized without the undue interference from video noise. For this purpose, a system was designed to capture image sequences from an aircraft using a digital camera, and record them digitally on a disk. The camera used was $1K \times 1K$ Kodak MegaPlus ES1.0 camera with the output at approximately 30 frames per second and a gray scale resolution of 8 bits. Hence, a bandwidth of 30 MBytes per second and a storage of 108 GBytes per hour of recording is required.

To capture the video image sequences with these large bandwidth and storage requirements, as well as perform the image processing operations in real time, a real-time image processing system with pipelined image processor called DataCube MaxPCI was procured. This system is a cost-effective way to meet high-throughput low-latency demands and has become popular among researchers working on real-time vision problems. The New Technology Disk (NTD) available with the DataCube MaxPCI has the required ability to perform high-speed digital image recording. NTD is a Redundant Array of Inexpensive Disks (RAID) that enables high-speed lossless digital image recording and playback. The image data can be recorded and played back at a real-time frame rate (overall 40 MBytes/sec).

Image data has been obtained from flight tests conducted at NASA Langley Research Center. A sample image captured using this system is shown in Figure 2.5. Work on implementing the detection algorithms on the DataCube hardware using these images is described in [34]. Detection of objects crossing the aircraft (instead of those on a collision course) was performed on the DataCube system in real time. The algorithms used for this purpose are described in Chapter 7.

Fig. 2.5. An image captured from an aircraft using the digital recording system. The target aircraft is in the middle-right part of the image.

Chapter 3

Performance Characterization of Detection Algorithms

The most common tool used to characterize the performance of a detection algorithm is a plot of its probability of mis-detection versus its probability of false alarm, as some tuning parameter is changed. This plot is commonly known as the "receiver operating curve" of the system, or ROC, for short. Although ROCs are useful to represent the system performance as a parameter is varied, they have several limitations. One disadvantage in using ROCs is due to the fact that only one parameter can be varied at a time. Thus, if the effect of variations of multiple variables needs to be studied, a different curve must be determined for each of these variables making the analysis of the system performance more difficult. A second disadvantage is that it is difficult to compare ROCs for different algorithms since they may take different variables into account. Finally, obtaining ROCs is an expensive process where factorial experiments must be carried out to determine the system performance at all performance levels with the probability of false alarms ranging from zero to one.

In Kanungo et al. [33], a methodology which was adapted from the psychology literature, and is discussed next, was proposed as an alternative characterization tool to summarize multiple ROCs into a single curve, solving the problems described above. This chapter describes how to use this methodology to characterize the performances of the algorithms described in Chapter 2. The performance of the dynamic programming algorithm is compared against that of temporal averaging, and thresholding of a single image frame.

3.1 Performance characterization methodology

For the sake of completeness, the methodology for performance characterization proposed in [33] is described here. Consider a detection algorithm that must report whether a given image has a target or not. Typically, the algorithm would compute some measure of evidence of target presence and compare it to some given threshold value. Whenever the evidence measure is greater than the given threshold, a target would be reported. The performance of the algorithm is affected by several factors, such as image

contrast, target size, complexity of the background, etc. The effect of variations of these variables on the overall performance can be measured through the use of equivalent effects of some critical signal variable by following the four steps described below.

1. Obtain evidence distributions: The first step consists on estimating distributions of evidence measures, one for images with target and another for images without target, as illustrated in Figure 3.1 (a). This estimation is done non-parametrically by randomly presenting the algorithm with images of both types and recording the frequency of the evidence measure values reported by the algorithm, using a histogram. It should be noted that the frequency distributions are used here only for estimating the false alarm and mis-detection rates. The evidence measure which is thresholded may or may not be derived from these distributions according to Bayes' rule. Hence, the performance of optimal as well as non-optimal detectors can be characterized by this approach.

2. Obtain ROCs: The second step consists on constructing an ROC as the one shown in Figure 3.1 (b) by varying the threshold used by the algorithm to compare against the computed evidence measure. False alarms occur when a pixel in the given image does not contain a target, but the evidence measure is greater than the threshold being used. Mis-detections occur when the given image contains a target, but the evidence measure is less than the threshold. The probabilities of false alarms and mis-detections can be approximated by their frequency ratios:

$$\begin{aligned} P(FA) &= P(H_1|H_0) = \frac{\text{Number of false alarms}}{\text{Total number of input pixels without target}} \\ P(MD) &= P(H_0|H_1) = \frac{\text{Number of mis-detections}}{\text{Total number of targets in input images}} \end{aligned}$$

   where H_0 and H_1 denote the hypotheses corresponding to the absence and presence of a target, respectively.

3. Determining the optimal operating point: The optimal operating point (or its corresponding threshold value) can be specified in different ways, depending on how much prior knowledge is available. If the prior probabilities and costs are known, the optimal operating point can be defined as the one minimizing the expected cost. Let C_{10}, C_{01}, C_{11}, and C_{00}, be the costs of a false alarm, a mis-detection, a correct detection, and a correct rejection, respectively. The expected cost is then

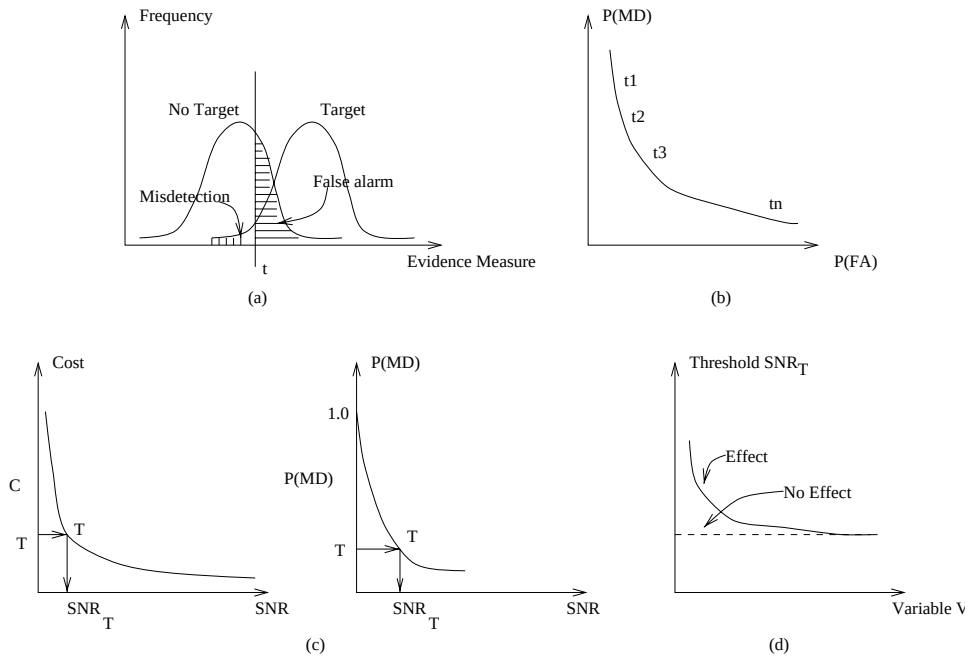


Fig. 3.1. Steps for performance characterization: (a) Step 1: Obtain the frequency distributions of the evidence measure for images with and without target. (b) Step 2: Obtain the ROC. (c) Step 3: Determine the optimal operating point using either the expected cost or the probability of detection given the probability of false alarm. (d) Step 4: Plot the threshold value corresponding to the optimal operating point versus a variable of interest.

given by:

$$\begin{aligned} E[C] &= [P(H_0|H_0)C_{00} + P(H_1|H_0)C_{10}]\,P(H_0) \\ &+ [P(H_0|H_1)C_{01} + P(H_0|H_1)C_{11}]\,P(H_1) \end{aligned} \tag{3.1}$$

The optimal operating point is found by minimizing $E[C]$ with respect to the threshold to be used by the algorithm. In the most likely case when the costs are difficult to set, an alternative way to define the required operating point is to use the Neyman-Pearson criterion – i.e., to maximize the probability of detection for a *given* probability of false alarm.

Independently of which definition is used, the optimal operating point depends on the signal to noise ratio (SNR) in the input image. For example, increasing the target contrast results in an increase of the SNR and, hopefully, in an improvement of the algorithm performance for a given threshold value. The optimal operating points for different SNRs can be found by repeating steps 1 and 2 for the corresponding SNR values and determining the optimal point for each of the resulting ROCs. Once this is done, a graph of the expected cost or the probability of detection versus SNR can be plotted, depending on which definition of operating point is being used. This is illustrated in Figure 3.1(c). Finally, let SNR_T and T be the SNR and the associated threshold values for the optimal operating point for a given level of performance, as shown in the figure. The level of performance is specified by either a desired expected cost of classification or a desired probability of mis-detection, again, depending on which optimal criterion is used.

4. Performance analysis with respect to variables of interest: Besides SNR, other factors affect the algorithm performance and merit study. Examples are the size of the target, the amount of target motion on the images, and the amount and nature of image clutter. In order to study these effects, steps 1 to 3 are repeated for different values of variables representing these variations. These results are then summarized in a graph where the threshold T determined in step 3 is plotted against the value of the variable of interest, as shown in Figure 3.1(d). A fairly flat plot indicates that the effect of the variable being considered on the optimal operating point of the algorithm is negligible. On the other hand, a steep plot indicates that the variable has a high impact on the performance.

It should be noted that a smaller SNR threshold T implies better performance, since weaker targets can be detected with the same given rates of false alarms and mis-detections. Measuring the performance in terms of the SNR threshold makes it easier to measure and compare the performance of different algorithms, or the same algorithm with different parameters. This is because the variables, such as the false alarm and mis-detection rates are eliminated from the curves, making place for other parameters.

3.2 Experimental protocol

In this section, the experimental protocol used to characterize the performance of the target detection algorithms, is described in detail. The protocol consists of the following components, specifying how to

1. Generate images of simulated targets,
2. Apply the detection algorithm,
3. Estimate the rates of false alarms and mis-detections (ROCs) for different sets of parameters, and
4. Characterize the algorithm performance by condensing the ROCs into a performance curve.

3.2.1 Image generation

In order to characterize the performance of the detection algorithm, it is applied to sequences of synthetic images with and without targets. While the images with targets are used to estimate the mis-detection rate, the images without targets are used to estimate the false alarm rate. The images can have the following different types of backgrounds:

1. Synthetic noise from camera model: The background is assumed to have a constant value A_{bg}. The noise is artificially simulated, using the camera noise model.
2. Real noise from a digital camera: The background images are taken from a sequence of images obtained from a digital camera looking at a scene with constant intensity such as clear sky, or white paper.
3. Real background an from analog camera: The background images are obtained using a sequence of images with significant clutter. The sequence, which was provided

by NASA, was captured using an analog camera mounted on a flying aircraft. Figure 3.2 shows a typical frame of this sequence.

Fig. 3.2. A sample image from the real background sequence provided by NASA. The image sequence was taken from an analog camera mounted on an aircraft.

Generation of image sequences

To estimate the number of false alarms, the background images themselves, without any addition of targets are used directly. The size of these images is $N_x \times N_y$. For estimation of the rate of mis-detections, simulated targets are inserted in the background images generated as described below. For each simulation, a target file is created having information on the position, velocity, size, amplitude and each target to be placed in an image. The image size is taken as $N_x \times N_y$. The number of targets to be inserted in every image is N_{targ}. The target trajectories are generated in such a way that the detection of one target does not interfere with the detection of another. This is accomplished by drawing a window around each target trajectory. The next generated trajectory is

valid only if the window around it does not overlap with the windows around the previously generated targets. Otherwise, the procedure is repeated by generating another trajectory, until the total number of valid trajectories is N_{targ}.

The velocity (V_x, V_y) of the targets is uniformly distributed so that $-u_{max} \leq V_x \leq u_{max}$ and $-v_{max} \leq V_y \leq v_{max}$. The position of the targets is specified for the *last* frame – i.e. when the detection is completed. The position of the target in other frames is given by $(x - V_x\Delta t, y - V_y\Delta t)$, where Δt is the time-interval between the given frame and the last frame.

A target can be a point target, or have a specified height and width. The size of the target is given by $s_x \times s_y$. The target amplitude is given by A. For point targets, the amplitude corresponds to the contrast of the pixel it occupies, with respect to the background. However, for an extended target, the contrasts of all the occupied pixels are given by the product of the target amplitude and the fraction of the area in the respective pixel that is covered by the target.

Figure 3.3 (a) shows the trajectories of simulated targets to be added to an image, and Figure 3.3 (b) shows a zoomed part on a portion of the image. The end of the trajectories are marked by blobs. The black box around the target denotes the region where another target cannot be present, to reduce the interference between the targets.

Once the file describing the targets is created, an image sequence of N_{frame} frames is generated. For each frame, the position of the targets are calculated, and the targets are inserted accordingly. For point targets, the amplitude is added to the background image in the target position pixel. For extended targets occupying a number of pixels (fully or partially), the product of the amplitude and the fractional occupancy is added to the background image at that pixel.

Addition of noise

Two types of camera noise [25, 34], the Fixed Pattern Noise (FPN) and the temporal noise are added to the sequences created using synthetic backgrounds. FPN has two components, additive and multiplicative. The parameters of this noise change from pixel to pixel, but do not change with time. The parameter values for each pixel are determined a priori using the camera, and stored as images. On the other hand, the temporal noise is completely random, and is generated separately for each frame. The temporal noise approximately follows a Gaussian distribution with a variance of:

$$\sigma^2_{noise} = w_0 + w_1 I$$

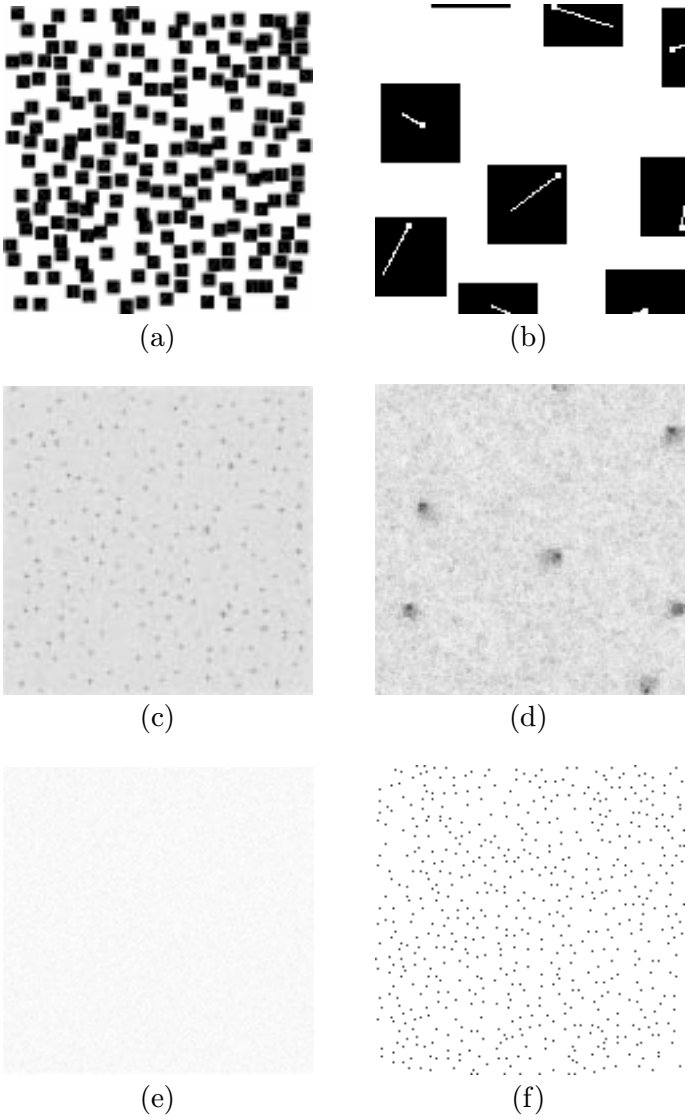


Fig. 3.3. Detection using dynamic programming: (a) Simulated targets trajectories. There are 200 targets, and the image size is 960×960. The end of the trajectory is marked by a blob. The targets are separated so that there the interference between them is reduced. The black box around the target denotes the region where another target cannot be present. (b) A zoomed part of the target trajectory image. (c) The dynamic programming output of a typical experiment (before non-maximal suppression). (d) Zoomed part of the output. (e) The dynamic programming output of the same experiment without adding targets – i.e., false alarms. (f) Zoomed part of the output.

where I is the expected gray value of the pixel, and w_0, w_1 are the parameters of the particular camera. However, since the background amplitude A_{bg} is constant for the experiments with simulated noise, and the target amplitude $A \ll A_{bg}$, we have $I = A + A_{bg} \simeq A_{bg}$ and σ^2_{noise} is approximately constant, given by:

$$\sigma^2_{noise} \simeq w_0 + w_1 A_{bg}$$

Hence, the noise can be approximated as Gaussian noise with a constant standard deviation of σ_{noise}. The values of the parameters for the particular camera were estimated [34] as $w_0 = 0.171$ and $w_1 = 0.0056$. For background $A = 128$, this gives $\sigma_{noise} = 0.942$. The image is quantized to give the output in byte format.

3.2.2 Algorithm application

The target detection algorithm whose performance is to be characterized is applied to each simulated image sequence. In the cases of synthetic images, and digital camera sequences, fixed pattern noise (FPN) can be corrected in advance by using pre-computed parameters of FPN for each pixel. However, these parameters are perturbed by a random amount corresponding to their estimated covariance, to model the error in estimating these values. Experiments are performed without and with correction of FPN, and the results are compared.

According to the type of background used, preprocessing in the form of a low-stop filter or a morphological filter are performed before applying dynamic programming. After dynamic programming is applied, non-maximal suppression is performed to ensure correct counting of false alarms and mis-detections. The output (before non-maximal suppression) of a typical experiment with 200 targets is shown in Figure 3.3 (c) and (d) where the latter shows a zoomed part of the output.

3.2.3 Estimation of false alarms (FA) and mis-detections (MD)

The algorithm to be characterized is applied on the image sequences with as well as without targets. The sequences without targets are used to estimate the false alarm rate, whereas the sequences with targets are used to estimate the mis-detection rate.

For the false alarm rate, the histogram of the output image is obtained. Using this histogram, the false alarm rates for different thresholds can be obtained. For the mis-detection rate, only the pixels in a specified window of 5×5 pixels around the specified target position are checked. For each such window corresponding to a single target, the

maximum value of the algorithm output is taken. A histogram of these maximum values is formed, and processed to obtain the mis-detection rates for different thresholds. The false alarm and mis-detection rates are averaged over a number of simulations N_{FA} and N_{MD}, respectively.

The number of simulations to test can be specified so that the standard deviation in the estimate of the false alarm or mis-detection rate is below a given value. This can be seen by observing that the occurrence of an event such as a false alarm or a mis-detection can be modeled as a Poisson process and therefore the variance of the total number of events is equal to the mean. Thus, if n events are observed, the standard deviation of the absolute error in the number of events is $\sqrt{n}$, and that of the relative error is $1/\sqrt{n}$. For example, for $n = 10$ events, the error σ is 3.2, or 32 % of the number of events. This error estimate can be confirmed by measuring the variance of these rates across the simulations.

3.2.4 Performance characterization

Using the estimated false alarm and mis-detection rates, the receiver operating curve (ROC) can be plotted showing the rate of mis-detection against the rate of false alarms. The mis-detection rate for a specified false alarm rate (FA_T) is noted from the curve. The simulations are repeated for a number of signal amplitudes A. The ratio of this amplitude to noise level corresponds to the SNR. The value of the signal amplitude for a specified mis-detection rate (MD_T), and the above false alarm rate is obtained. This is considered as the threshold signal value (A_T). The number of simulations used is at least $N_{FA} = 10/FA_T$ in the case of false alarms and $N_{MD} = 10/MD_T$ in the case of mis-detections, so that for the rates FA_T and MD_T, an average of at least 10 events would be observed, giving an error σ of at most 32 %. Due to constraints on the execution time, larger number of experiments were not used, although they would be desirable for reducing this error.

Other parameters, such as the size of the target, can be varied one at a time, and the variation of A_T can be plotted against the respective parameter to determine the effect of the parameter on the algorithm performance.

3.3 Results

The target detection algorithm was tested on 3 categories of images as described in the protocol. The results are shown and compared in the following sections.

Table 3.1. Table of parameters used for the experiments with the following image categories: (1) Synthetic noise from camera model, (2) Real noise from a digital camera, (3) Real background from an analog camera.

Description	Parameter	Category		
		(1)	(2)	(3)
Image x size	N_x	960	960	640
Image y size	N_y	960	960	480
No. of targets	N_{targ}	200	200	50
Maximum x velocity	u_{max}	1	1	1
Maximum y velocity	v_{max}	1	1	1
x size	s_x	0.5 to 2	2	2
y size	s_y	0.5 to 2	2	2
Amplitude	A	1.0 to 15.0	1.0 to 6.0	10.0 to 70.0
Number of frames	N_{frame}	32	32	32
Background value	A_{bg}	128	$\simeq 200$	not used
Noise standard deviation	σ_{noise}	0.942	not used	not used
Forgetting factor	α	15/16	15/16	15/16
Number of FA simulations	N_{FA}	500	1	1
Number of MD simulations	N_{MD}	50	10	10
Threshold FA rate	FA_T	0.02	10	10
Threshold MD rate	MD_T	0.001	0.01	0.01

3.3.1 Synthetic noise from camera model

In this case, the noise was synthetically generated using the noise model of the Kodak Megaplus ES 1.0 digital camera. Targets of varying size were added for mis-detection analysis. Experiments without and with correction of FPN were performed.

Figure 3.4 (a) and (b) show the plots of the false alarm and mis-detection rates, respectively, against the threshold value, for experiments without FPN correction. The mis-detection rates are shown for a number of signal amplitudes for 1×1 targets. The mis-detection rate is measured as the ratio of the average number of mis-detections, to the total number of targets in a simulation. However, the false alarm rate is measured as the average number of false alarms per simulation, instead of the ratio of the number of false alarms to the total number of pixels. This is done to give a better idea of the algorithm performance.

Figure 3.4 (c) shows the plot of mis-detection rate against false alarm rate for different amplitude values for 1×1 targets. The point of threshold false alarm rate FA_T is set to 0.02 false alarms per simulation, which corresponds to a total of 10 false alarms for $N_{FA} = 500$ simulations. Figure 3.4 (d) shows the plot of mis-detection rate against the amplitude values for the above rate of false alarms. The A_T for the threshold mis-detection rate of MD_T is interpolated, and marked as a circle. The MD_T is set to a probability of 0.001 per target, which corresponds to an average of 0.2 mis-detections per simulation for a simulation with 200 targets, or a total of 10 mis-detections for $N_{MD} = 50$ simulations. The corresponding graphs for the case where fixed pattern noise compensation was applied are shown in Figure 3.5.

The above experiments are repeated for other sizes of targets, and the A_T calculated from these is plotted against the size of the target. Resulting plots for the experiments without FPN correction are shown in Figure 3.6 (a) for square targets (size $x \times x$) and in Figure 3.6 (b) for rectangular targets (size $1 \times x$). The corresponding results for the experiments with FPN correction are shown in Figure 3.6 (c) and (d). The threshold amplitudes for various sizes are tabulated in Table 3.2. It is seen that larger targets require smaller signal amplitudes for detection implying better performance. Similarly, the signal amplitudes required when FPN correction is applied are much smaller than those when the correction is not applied, implying better performance in the former case.

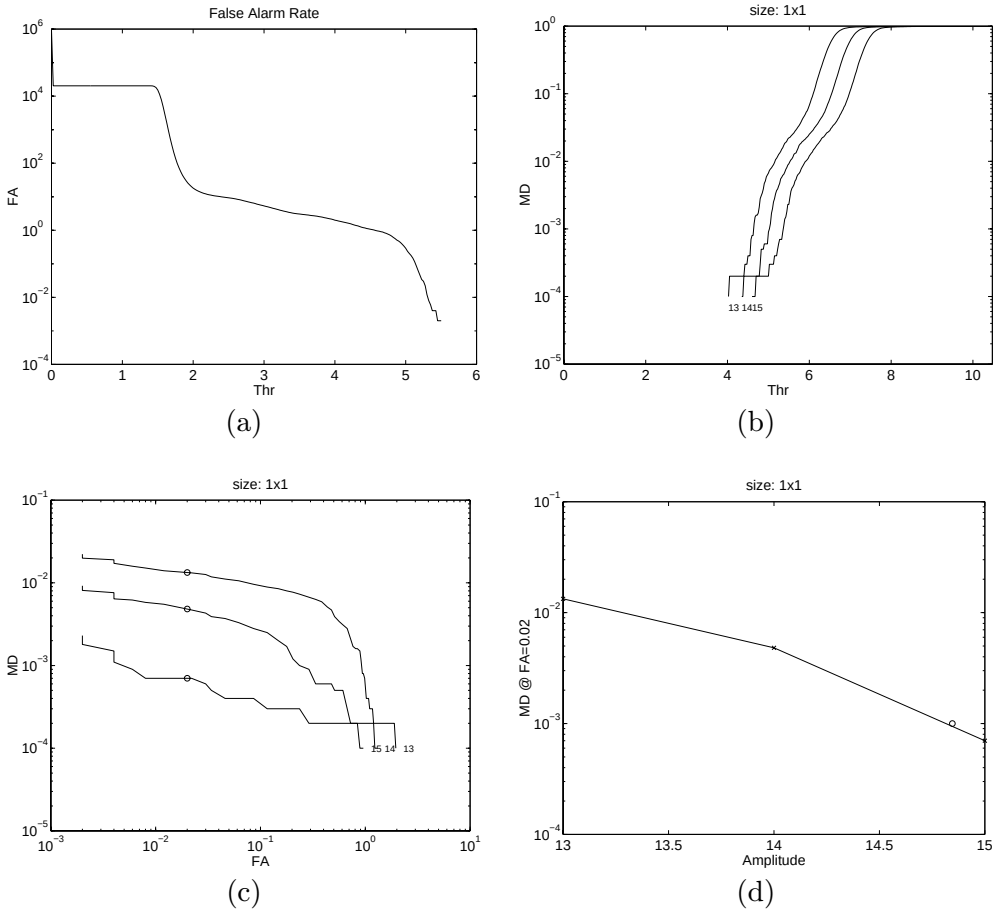


Fig. 3.4. Results for camera noise model without FPN correction: (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right) for 1×1 targets. (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value amplitude when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

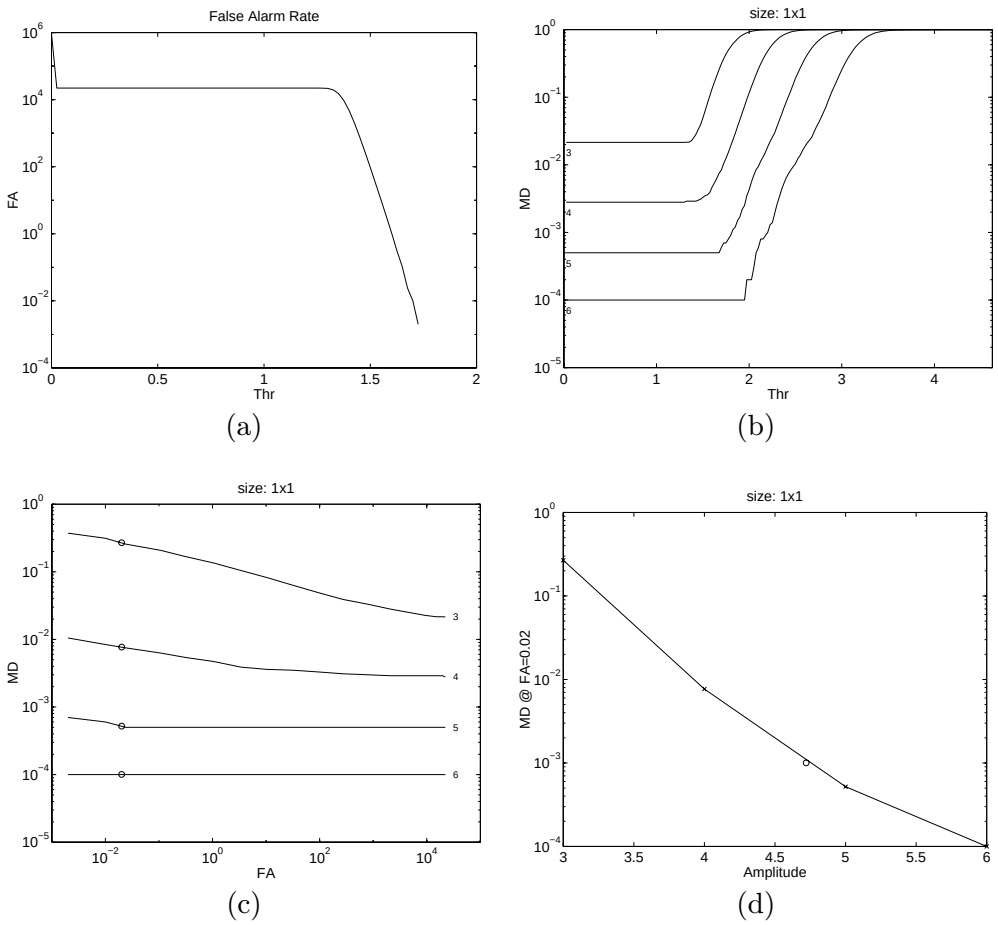


Fig. 3.5. Results for camera noise model with FPN correction: (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right) for 1×1 targets. (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value amplitude when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

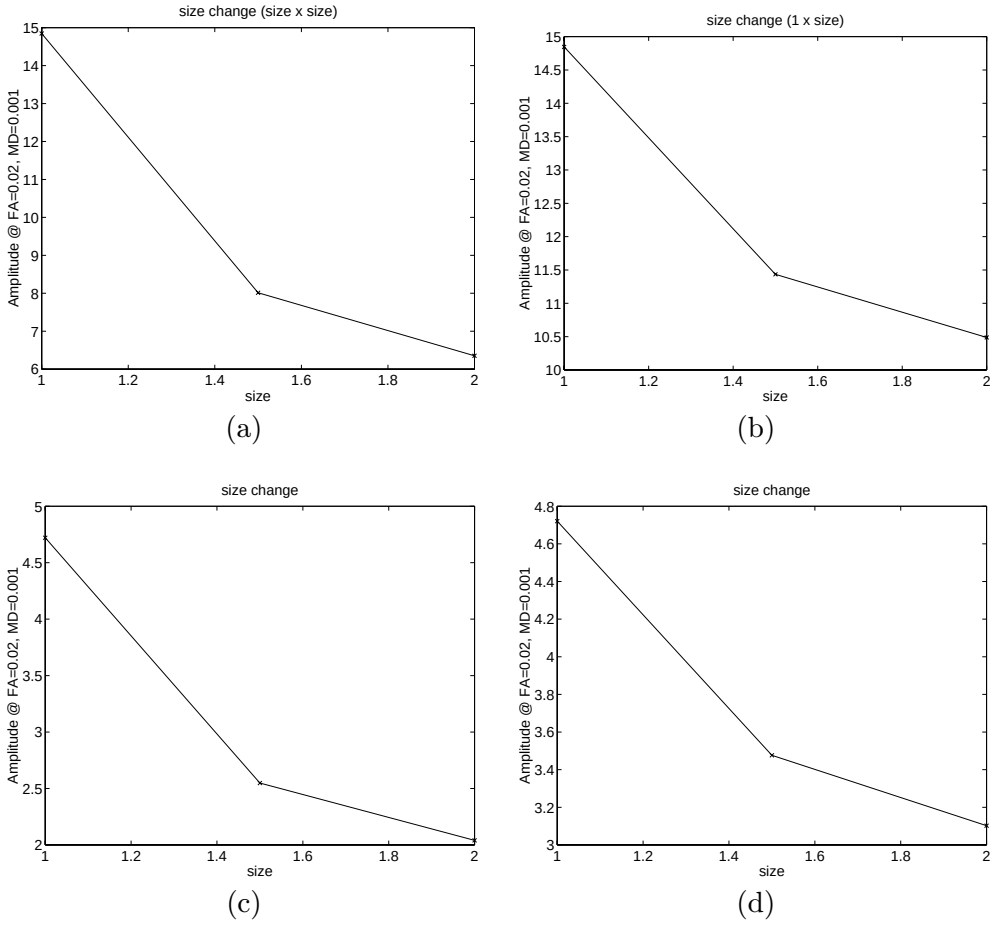


Fig. 3.6. Performance curves for simulated targets: (a) Plot of amplitude against the target size ($x \times x$) for experiments without FPN correction. The data points are marked as crosses. (b) Plot of amplitude against the target size ($1 \times x$). (c) and (d) Corresponding plots for experiments with FPN correction.

Table 3.2. Results of dynamic programming algorithm on simulated image sequences without and with FPN correction. Threshold amplitudes are shown for false alarm rate of 0.02 per simulation and mis-detection rate of 0.001 per target.

Size	No FPN correction	With FPN correction
1×1	14.85	4.72
1×1.5	11.43	3.48
1×2	9.38	3.10
1.5×1.5	10.49	2.55
2×2	6.35	2.04

3.3.2 Real noise from a digital camera

In this case, instead of synthetically generating the noise, background images captured using the Kodak Megaplus ES 1.0 digital camera looking at the sky were used. Targets of size 2×2 pixels were synthetically added for the mis-detection analysis. Experiments without and with correction of FPN were also performed.

The false alarm threshold was set $FA_T = 10$ per simulation, resulting in a total of 10 false alarms for $N_{FA} = 1$ simulation. The mis-detection threshold was set to $MD_T = 0.01$ per target, corresponding to 20 mis-detections for $N_{MD} = 10$ simulations with $N_{targ} = 200$ targets. Unfortunately, the performance at lower rates of false alarms and mis-detections could not be reliably estimated because of the limited number of background images available. However, one can extrapolate the false alarm and mis-detection rates to study the behavior of the algorithm for lower rates. Due to the normal distribution of noise, even a small increase in the threshold reduces the false alarm and mis-detection rates dramatically. Hence, a somewhat higher target amplitude can be expected to reduce these rates to an acceptable level.

In the case of the experiments without FPN correction, the plot of mis-detection rate against false alarm rate for different levels of target amplitude is shown in Figure 3.7 (a). The plot of mis-detection rate against SNR for false alarm rate of $FA_T = 10$ per simulation is shown in Figure 3.7 (b). The corresponding plots for the experiments with FPN correction are shown in Figure 3.7 (c) and (d). The target strength required for detection at the specified rates of false alarms and mis-detections are marked by circles in Figures 3.7 (b) and (d). It can be seen that the target strength required when

FPN is not corrected ($A_T = 3.22$) is higher than that required when FPN correction is applied ($A_T = 1.86$), implying better performance in the latter case.

3.3.3 Real background an from analog camera

In this case, a real aerial background, obtained from an analog camera used during a flight test was employed. Targets of size 2×2 pixels were synthetically added for mis-detection analysis.

In order to suppress the background, low-stop and morphological pre-processing were separately applied, and the results compared. Since the background was cluttered, a much higher signal was required for satisfactory detection. Even then, the false alarm rate does not reduce sufficiently, thus showing that more post-processing would be required after applying the algorithm. However, since the number of false alarms (plus true candidates) would be small after this processing, the time complexity of subsequent algorithms would be reduced significantly. The techniques described in Chapter 5 can be used to separate the remaining background clutter from the genuine targets. These techniques utilize the difference in the image translation and expansion between an object on a collision course, and the background clutter.

The false alarm threshold was set $FA_T = 10$ per simulation resulting in a total of 10 false alarms for $N_{FA} = 1$ simulation. The mis-detection threshold was $MD_T = 0.01$ per target, corresponding to 10 mis-detections for $N_{MD} = 20$ simulations with $N_{targ} = 50$ targets. Again, unfortunately, lower rates for false alarm and mis-detection cannot be reliably estimated due to the limited number of background images available.

The results for the morphological filter and the low-stop filter are shown in Figures 3.8 and 3.9, respectively. It can be seen that the target strength required when the morphological filter ($A_T = 17.8$) is used is much lower than that required when the low-stop filter ($A_T = 57.8$) is used. The morphological filter is thus better, and the reason for this is that the morphological filter reduces clutter corresponding to large features, whereas the low-stop filter does not do this effectively. However, both result in much poorer performance than that obtained with a digital camera with clear background.

3.3.4 Comparison with other methods

The performance of the dynamic programming algorithm was also compared with other methods such as simple thresholding on a single frame, and temporal averaging on the same number of frames. The comparison was made using FPN correction on

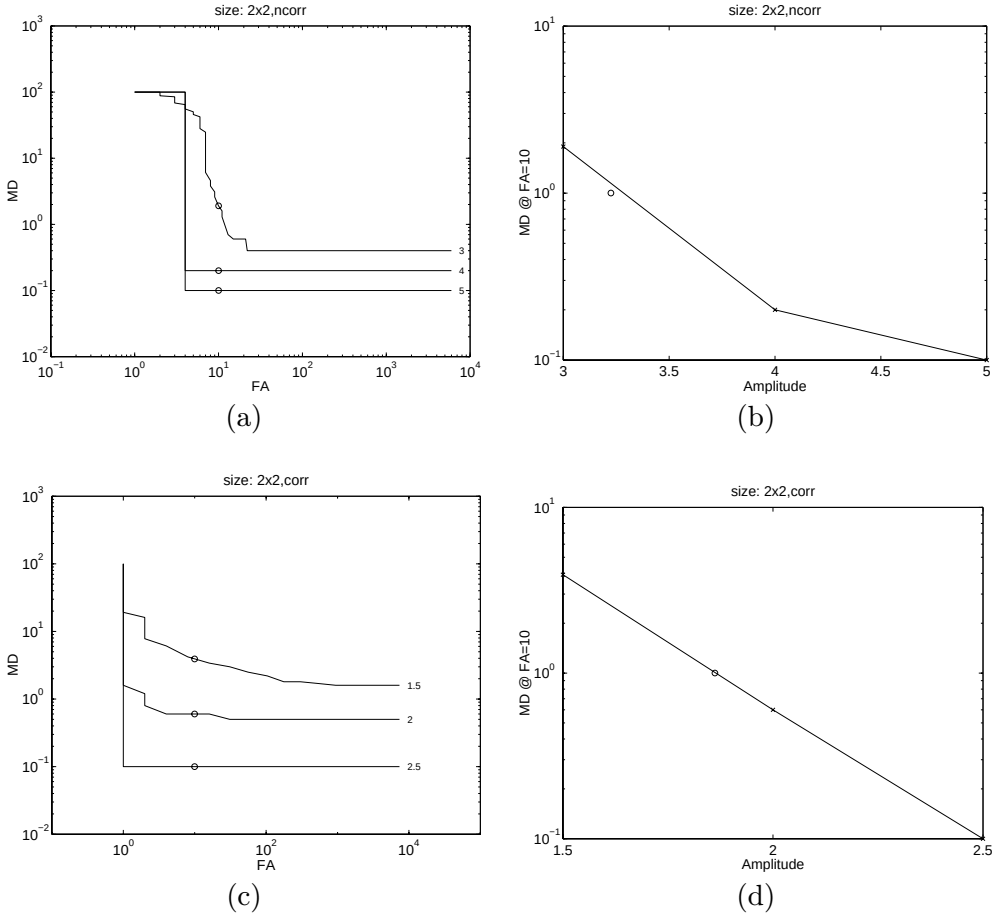


Fig. 3.7. Results for real noise from camera for 2×2 targets: (a) Plot of MD rate against FA rate (for marked amplitude) for images without FPN correction. The data points are marked as crosses. The MD rate when FA rate is $FA_T = 10$ per simulation is interpolated, and plotted as circle. (b) Plot of MD rate against amplitude for FA rate of $FA_T = 10$ per simulation. The data points are marked as crosses. The value of A_T where MD rate is $MD_T = 0.01$ per target is interpolated and marked as a circle. (c) and (d) Corresponding plots for FPN corrected images.

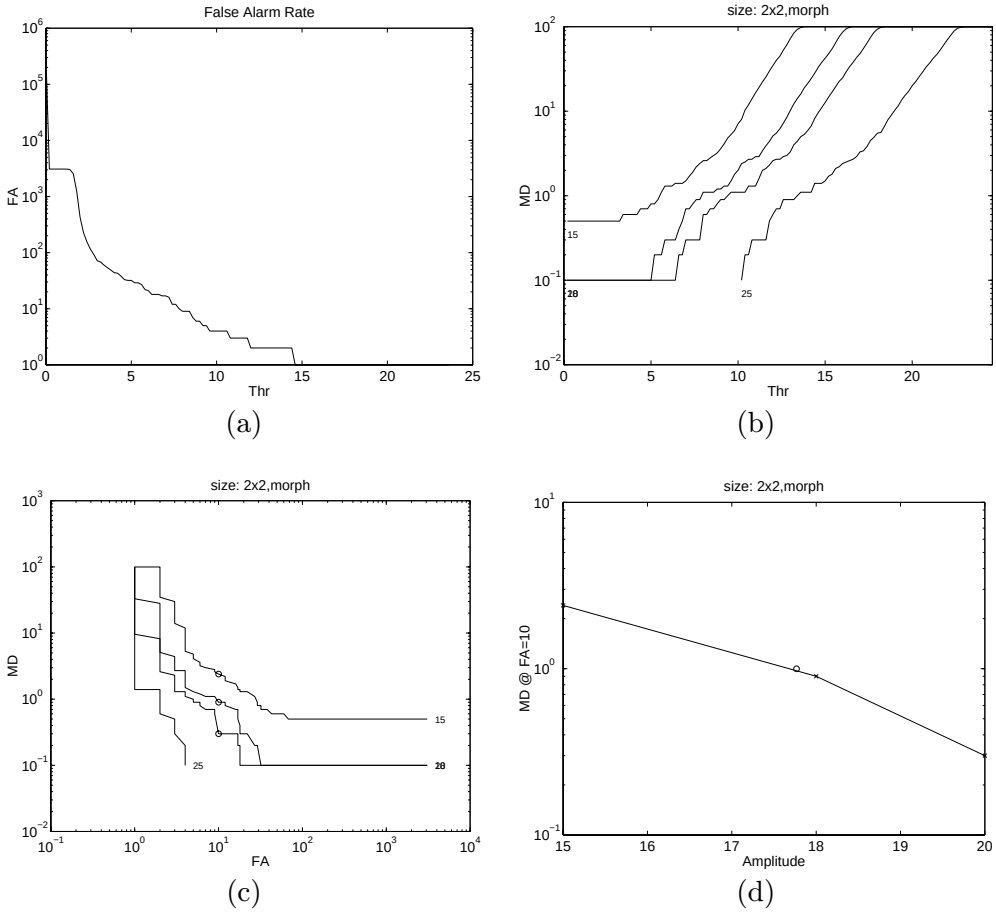


Fig. 3.8. Results for real cluttered background for 2×2 targets using morphological filter in the preprocessing: (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 10$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 10$ per simulation. The data points are marked as crosses. The value of A_T where MD rate is $MD_T = 0.01$ per target is interpolated and marked as a circle.

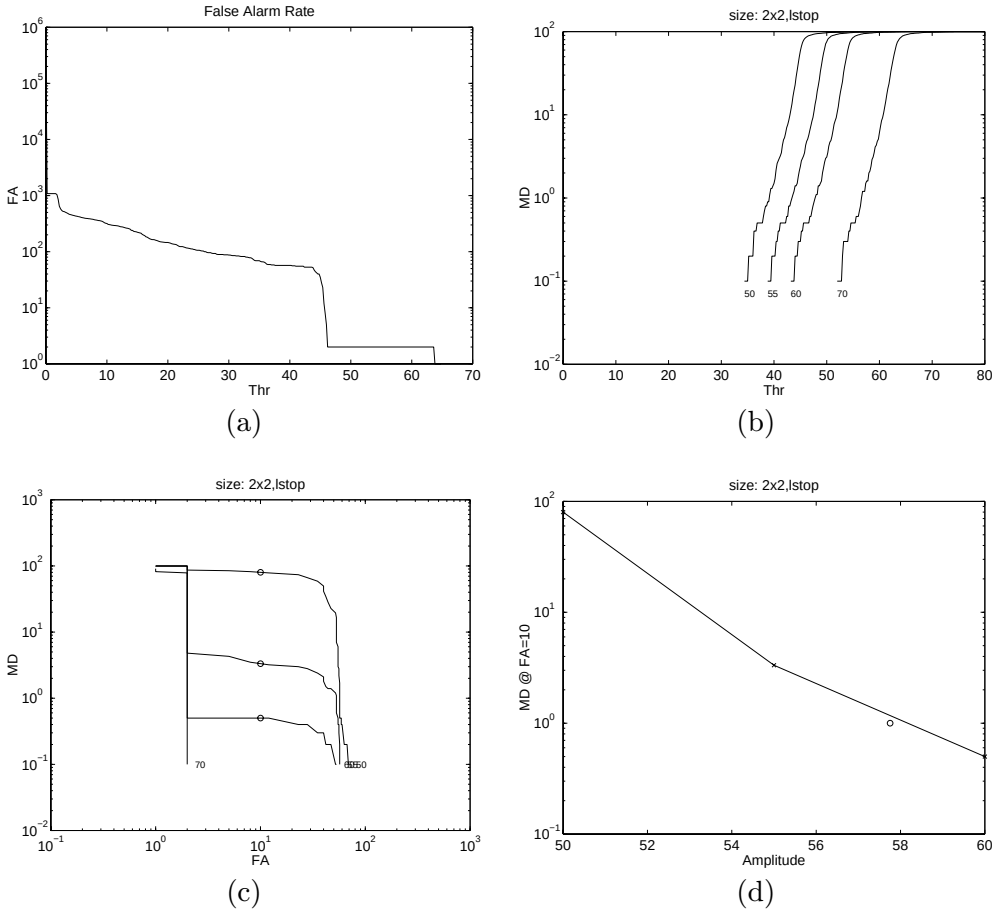


Fig. 3.9. Results for real cluttered background for 2×2 targets using low stop filter in the preprocessing: (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 10$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 10$ per simulation. The data points are marked as crosses. The value of A_T where MD rate is $MD_T = 0.01$ per target is interpolated and marked as a circle.

images with simulated camera noise. The results of applying the dynamic programming algorithm, simple thresholding on a single frame, and temporal averaging on image sequences with 2×2 moving targets are shown in are shown in Figures 3.10, 3.11 and 3.12 respectively. Temporal averaging was also applied on image sequences with stationary targets instead of moving targets, the results of which are shown in Figure 3.13.

Similar experiments were performed with other target sizes. Table 3.3 shows the comparison the for these algorithms using various target sizes. The plots of the threshold amplitudes against target sizes are shown in Figure 3.14. Again, smaller threshold amplitudes imply better performance as explained before.

It can be seen that the performance of single frame thresholding, as well as temporal averaging are much poorer than that of the dynamic programming. However, if stationary targets are used instead of moving targets, the performance of temporal averaging is slightly better than that of dynamic programming, showing that temporal averaging is the best choice when the targets are stationary.

Table 3.3. Results of target detection algorithms on simulated image sequences with FPN correction. Threshold amplitudes are shown for false alarm rate of 0.02 per simulation and mis-detection rate of 0.001 per target.

Size	Dynamic prog.	Single frame thresh.	Temp. Avg. (moving)	Temp. Avg. (stat.)
1×1	4.72	23.03	33.82	4.63
1.5×1.5	2.55	10.68	16.99	2.11
2×2	2.04	8.17	11.67	1.65

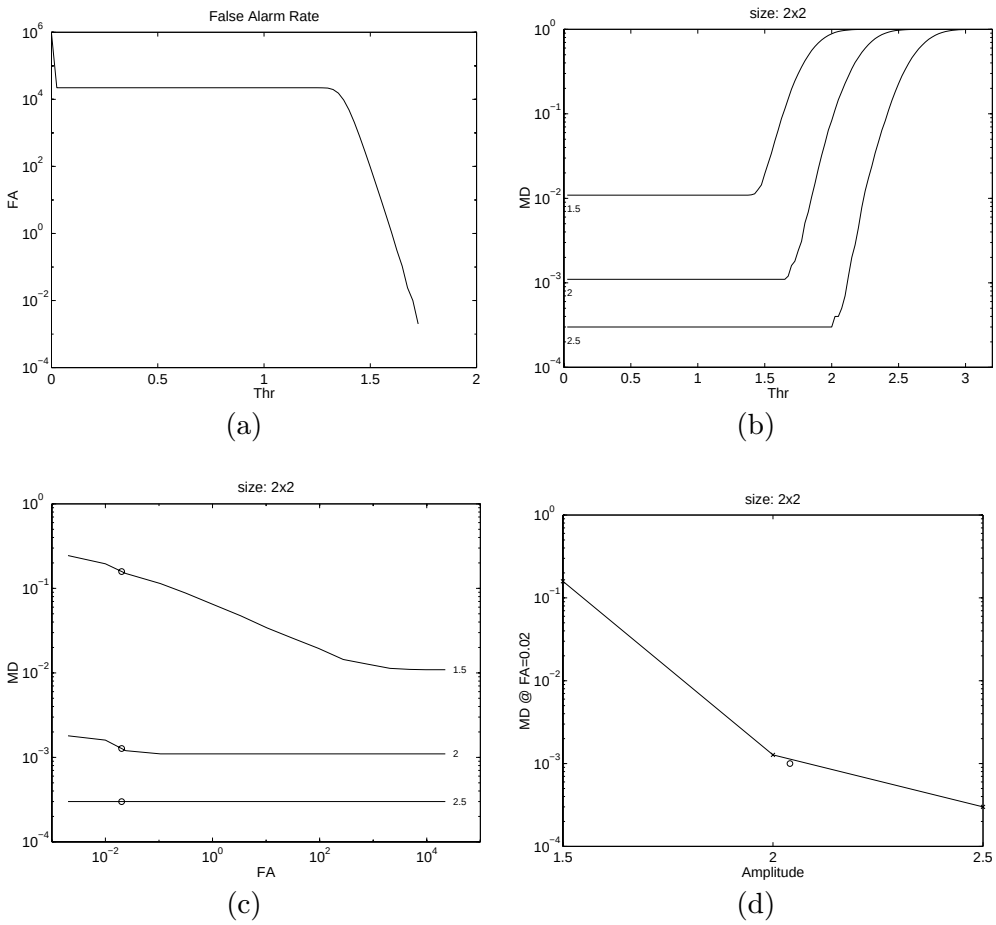


Fig. 3.10. Results for camera noise model with FPN correction for 2×2 targets using dynamic programming: (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value of A_T when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

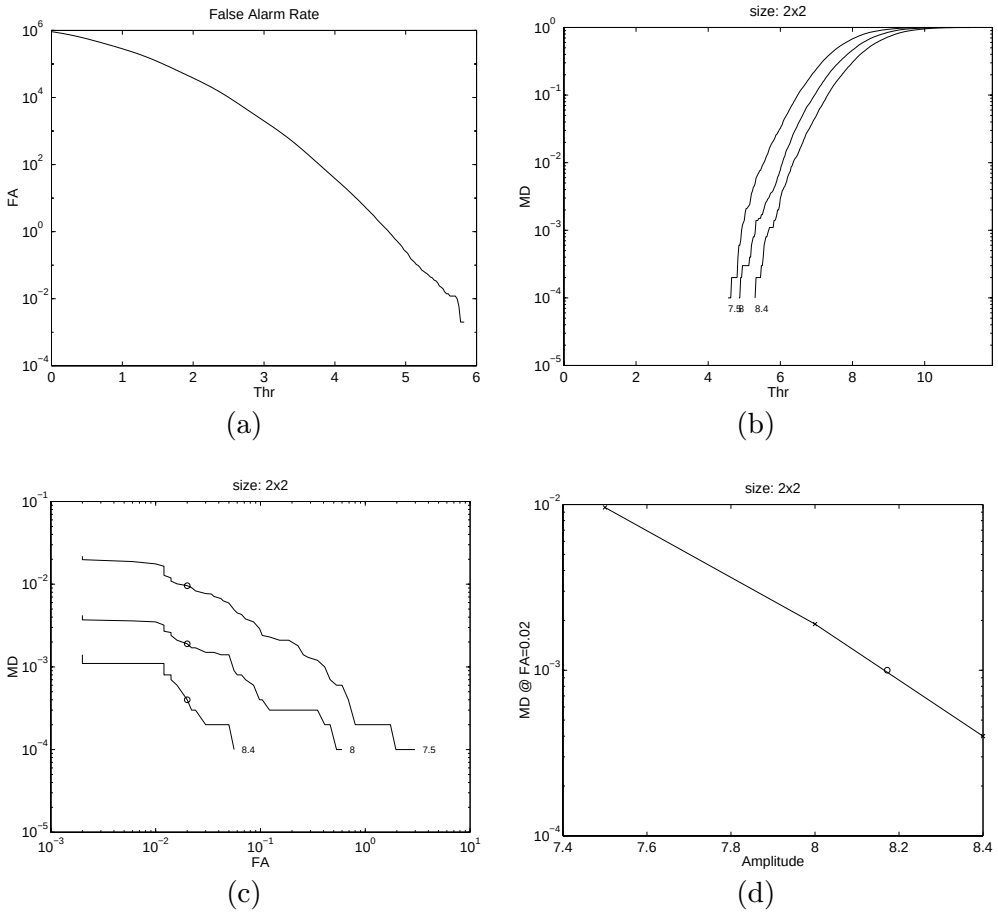


Fig. 3.11. Results for camera noise model with FPN correction for 2×2 targets by thresholding a single frame. (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value of A_T when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

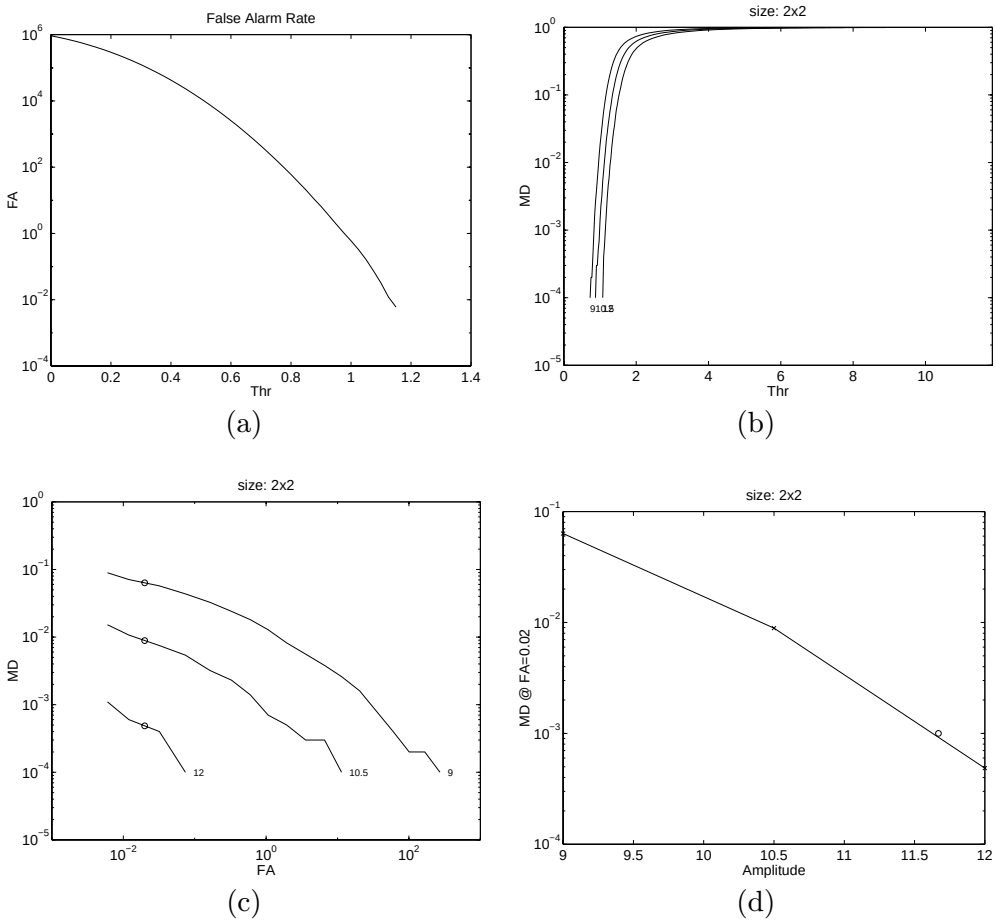


Fig. 3.12. Results for camera noise model with FPN correction for 2×2 *moving* targets using temporal averaging. (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value of A_T when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

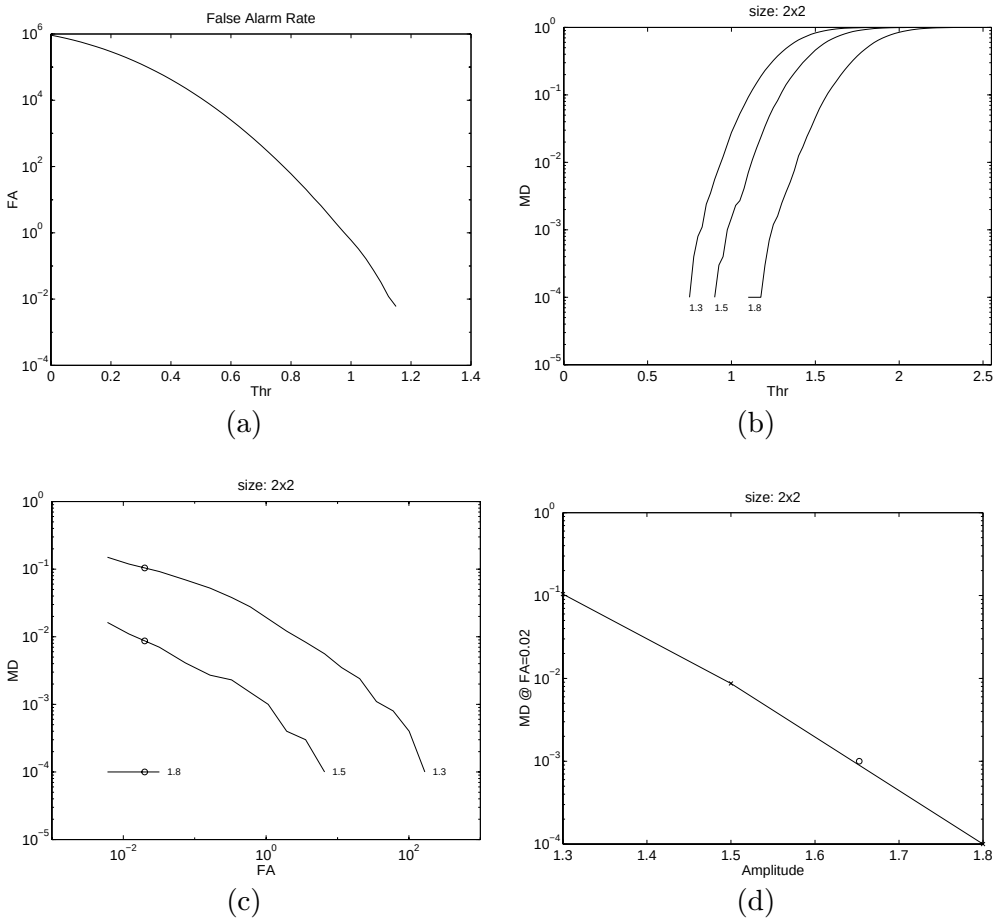


Fig. 3.13. Results for camera noise model with FPN correction for 2×2 *stationary* targets using temporal averaging. (a) Plot of FA rate (average number per simulation) against threshold (b) Plot of MD rate against threshold, for a number of signal amplitudes (higher amplitudes towards right). (c) Plot of MD rate against FA rate (for marked amplitude). The data points are marked as crosses. The MD rate when FA rate is $FA_T = 0.02$ per simulation is interpolated, and plotted as circle. (d) Plot of MD rate against amplitude for FA rate of $FA_T = 0.02$ per simulation. The value of A_T when MD rate is $MD_T = 0.001$ per target is interpolated and marked as a circle.

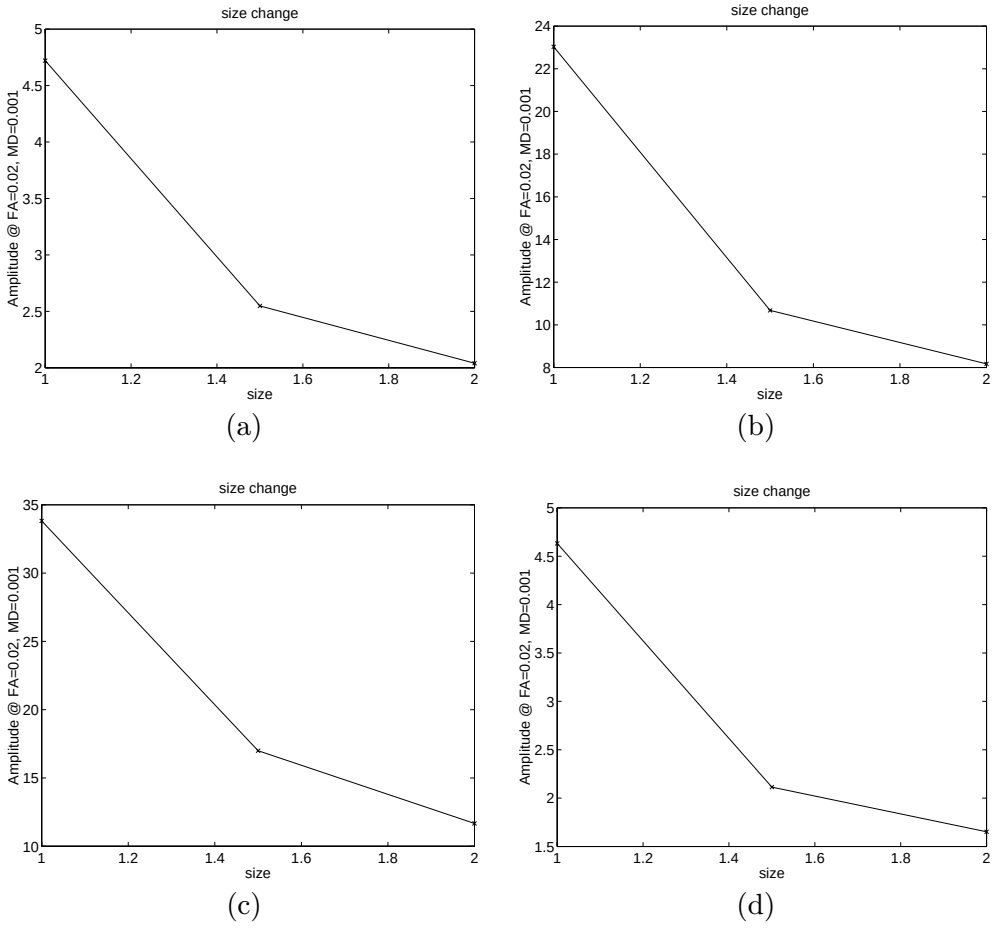


Fig. 3.14. Performance comparison of several algorithms: Plot of amplitude against the target size ($x \times x$) for experiments without FPN correction using (a) Dynamic programming, (b) Thresholding single frame, (c) Temporal averaging (moving targets), (d) Temporal averaging (stationary targets).

Chapter 4

Theoretical Performance of Detection Algorithms

In this chapter, the approximate theoretical performance of the algorithms presented in Chapter 2 are derived. The theoretical derivations are based on the paper by Tonissen and Evans [61]. The theoretical performance is compared with the experimentally observed performance described in Chapter 3. Effects of approximations used in the derivations are also described.

4.1 Dynamic programming algorithm

The dynamic programming algorithm described in Chapter 2 can be summarized as follows:

1. Initialization: For all pixels (x, y) and all velocities (u, v), set

$$F(x, y; u, v; 0) = 0$$

2. Recursion: At time k, set

$$F(x, y; u, v; k) = (1 - \alpha) f(x, y; k) + \alpha \max_{(x', y') \in Q} F(x - u - x', y - v - y'; u, v; k - 1)$$

3. Termination: At time K, take

$$F_{max}(x, y; K) = \max_{(u,v) \in P} F(x, y; u, v; K)$$

The number of elements in sets P and Q are denoted by p and q, respectively. The values of $p = q = 4$ have been used in our implementation, with $u, v \in \{-1, 0\}$ and $x', y' \in \{0, 1\}$. Note that for theoretical analysis, the recursion step is replaced by:

$$F(x, y; u, v; k) = f(x, y; k) + \alpha \max_{(x', y') \in Q} F(x - u - x', y - v - y'; u, v; k - 1) \qquad (4.1)$$

However, this only changes F by a scale factor, and since both signal as well as noise would be scaled equally, SNR analysis does not change.

4.2 False alarm and mis-detection probabilities

Probability of false alarms P_{FA} is the probability that there is at least one state exceeding the threshold V_T out of p velocity states at the final output time K, for the pixel where there is no signal in its neighborhood – i.e., hypothesis H_0.

$$P_{FA}(x,y) = Pr\left[\max_{(u,v)\in P} F(x,y;u,v;K) \geq V_T | H_0\right] = 1 - [P_{0,K}(V_T)]^p \tag{4.2}$$

where $P_{0,K}(V_T)$ denotes the probability of F for hypothesis H_0 at time K, being less than or equal to the threshold V_T.

Probability of mis-detection P_{MD} is the probability that there is no output with correct velocity (u,v) exceeding the threshold at time K, within a neighborhood R of size $r+1$, where one cell contains signal – i.e., hypothesis H_1 – and the other r cells are noise. This allows for some tolerance in the location of target. For example, a 5×5 neighborhood corresponding to $r+1 = 25$ gives a tolerance of ± 2 pixels in the location of the target. On the other hand, a 1×1 neighborhood consisting of only the target position corresponds to $r+1 = 1$ or $r = 0$ giving no tolerance for the target position.

$$\begin{aligned} P_{MD}(x,y;u,v) &= Pr\left[\max_{x',y'\in R} F(x-x', y-y'; u, v; K) \leq V_T | H_1\right] \\ &= P_{1,K}(V_T)[P_{0,K}(V_T)]^r \end{aligned} \tag{4.3}$$

where $P_{1,K}(V_T)$ denotes the probability of F for hypothesis H_1 at time K being less than or equal to the threshold V_T.

4.3 Normal approximations

For an analytic solution of the performance of the dynamic programming algorithm, the distributions of the intermediate outputs can be approximated using normal approximations. Consider q independent standard normal variables $w_i \sim N(0,1)$. The cumulative distribution function (CDF) of the maximum of these variables is given by:

$$P(w) = Pr\left[\max_i w_i \leq w\right] = \prod_i Pr[w_i \leq w] = [\Phi(w)]^q \tag{4.4}$$

where $\Phi(\cdot)$ is the CDF of a standard normal variable. The probability density function (PDF) is the derivative of the CDF given by:

$$p(w) = q[\Phi(w)]^{q-1} G(w) \tag{4.5}$$

where $G(\cdot)$ is the standard normal PDF.

This distribution of maximum of q standard normal variables can be approximated as a normal distribution $N(\mu_q, \sigma_q^2)$, where μ_q and σ_q^2 denote the mean and the variance of the actual distribution. These are computed using numerical integration, and are tabulated in Table 4.1 for different values of q.

For general normal variables $z_i \sim N(\mu, \sigma^2)$, one can substitute: $z_i = \mu + \sigma w_i$ where w_i are standard normal variables. The maximum of z_i is approximately normally distributed with mean and variance given by:

$$\begin{aligned} E[\max z_i] &= \mu + \sigma E[\max w_i] = \mu + \sigma \mu_q \\ V[\max z_i] &= \sigma^2 V[\max w_i] = \sigma^2 \sigma_q^2 \end{aligned} \tag{4.6}$$

Table 4.1. Values of μ_q and σ_q^2 for a number of values of q.

q	μ_q	σ_q^2
1	0	1
4	1.029	0.491
9	1.485	0.3574
25	1.965	0.2585
49	2.241	0.2168

Let the input at any time k be normally distributed, both in absence and presence of the target, so that:

$$f(x, y; k|H_0) \sim N(\mu_n, \sigma_n^2)\ ,\ f(x, y; k|H_1) \sim N(\mu_s, \sigma_s^2) \tag{4.7}$$

Then, the distributions of the output F at time k will also be *approximately* normally distributed so that

$$F(x,y;u,v;k|H_0) \simeq N(M_{0,k}, S^2_{0,k})\ ,\ F(x,y;u,v;k|H_1) \simeq N(M_{1,k}, S^2_{1,k}) \tag{4.8}$$

where the M and S parameters are calculated below.

4.4 False alarm analysis

For noise pixels, we have:

$$\begin{aligned} F(x,y;u,v;0) &= 0 \\ F(x,y;u,v;k) &= f(x,y;k) + \alpha \max_{(x',y')\in Q} F(x-u-x', y-v-y'; u, v; k-1) \\ &\simeq N(M_{0,k}, S^2_{0,k}) \end{aligned} \tag{4.9}$$

Using equation (4.6), the mean and variance parameters at time k can be recursively expressed as:

$$\begin{aligned} M_{0,0} = 0\ ,\ M_{0,k} &= \mu_n + \alpha(M_{0,k-1} + \mu_q S_{0,k-1}) \\ S^2_{0,0} = 0\ ,\ S^2_{0,k} &= \sigma_n^2 + \alpha^2\sigma_q^2 S^2_{0,k-1} \end{aligned} \tag{4.10}$$

Solving these recursive equations yields expressions for mean and variance at time K:

$$\begin{aligned} S^2_{0,K} &= \sigma_n^2 \frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2} \\ M_{0,K} &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \alpha\mu_q \sum_{i=0}^{K-1} (\alpha^i S_{0,K-i-1}) \end{aligned} \tag{4.11}$$

To get approximate closed-form expressions for $M_{0,K}$, one can write $S_{0,k}$ as:

$$S_{0,k} = \sigma_n \sqrt{\frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2}} \simeq \frac{\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}} \left(1 - \gamma_k \alpha^{2k}\sigma_q^{2k}\right) \tag{4.12}$$

where γ_k is dependent on k but always lies between 0 and 1. Using $\gamma_k = 1/2$ is equivalent to using the first order term of binomial expansion, whereas $\gamma_k = 0$ corresponds to assuming that $S_{0,k}$ remains approximately constant with k, which is justifiable, since σ_q^2 is quite small. Accordingly, we have:

$$\begin{aligned} M_{0,K} &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \alpha\mu_q \sum_{k=0}^{K-1} (\alpha^k S_{0,K-k-1}) \\ &= \frac{1-\alpha^K}{1-\alpha}\mu_n + \frac{\alpha\mu_q\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}} \left[\frac{1-\alpha^K}{1-\alpha} - \gamma\alpha^{K-1}\frac{1-(\alpha\sigma_q^2)^K}{1-\alpha\sigma_q^2} \right] \end{aligned} \quad (4.13)$$

where γ is a function of all γ_k and also lies between 0 and 1. Values of $\gamma = 0$ and $\gamma = 1/2$ can be used as the zero order and first order approximations, respectively. For $K \to \infty, \alpha \neq 1$ such that $\alpha^K \ll 1$ (also, $\sigma_q^{2K} \ll 1$), we have:

$$\begin{aligned} S_{0,K}^2 &= \frac{\sigma_n^2}{1-\alpha^2\sigma_q^2} \\ M_{0,K} &= \frac{1}{1-\alpha}\left[\mu_n + \frac{\alpha\mu_q\sigma_n}{\sqrt{1-\alpha^2\sigma_q^2}} \right] \end{aligned} \quad (4.14)$$

For the case when $\alpha = 1$, the sum $\sum_{i=0}^{K}\alpha^i$ changes from $(1-\alpha^K)/(1-\alpha)$ to K. Hence, the expressions become:

$$\begin{aligned} S_{0,K}^2 &= \sigma_n^2 \frac{1-\sigma_q^{2K}}{1-\sigma_q^2} \\ M_{0,K} &= K\mu_n + \frac{\mu_q\sigma_n}{\sqrt{1-\sigma_q^2}} \left[K - \gamma\frac{1-(\sigma_q^2)^K}{1-\sigma_q^2} \right] \end{aligned} \quad (4.15)$$

Finally, the probability of false alarms is:

$$P_{FA} = 1 - [P_{0,K}(V_T)]^p \quad (4.16)$$

giving

$$P_{0,K}(V_T) = (1-P_{FA})^{1/p} = \Phi\left(\frac{V_T - M_{0,K}}{S_{0,K}}\right) \quad (4.17)$$

where $\Phi(\cdot)$ denotes the CDF of a standard normal variable. Hence, the threshold V_T can be expressed in terms of the mean $M_{0,K}$, variance $S_{0,K}$, and the false alarm probability P_{FA} as:

$$V_T = M_{0,K} + S_{0,K}\Phi^{-1}[(1-P_{FA})^{1/p}] = M_{0,K} + S_{0,K}\phi_{0,p} \tag{4.18}$$

where

$$\phi_{0,p} = \Phi^{-1}[(1-P_{FA})^{1/p}] \simeq \Phi^{-1}\left[1 - P_{FA}/p\right] \tag{4.19}$$

4.5 Missed detection analysis

The probability of mis-detection is given by:

$$P_{MD} = P_{1,K}(V_T)[P_{0,K}(V_T)]^r \leq P_{1,K}(V_T) \tag{4.20}$$

Substituting the expression of V_T in terms of false alarm rate, we have:

$$P_{MD} = (1-P_{FA})^{r/p} P_{1,K}(V_T) \tag{4.21}$$

giving

$$P_{1,K}(V_T) = \frac{P_{MD}}{(1-P_{FA})^{r/p}} = \Phi\left(\frac{V_T - M_{1,K}}{S_{1,K}}\right) \tag{4.22}$$

Hence,

$$V_T = M_{1,K} + S_{1,K}\Phi^{-1}\left[\frac{P_{MD}}{(1-P_{FA})^{r/p}}\right] = M_{1,K} - S_{1,K}\,\phi_{1,p} \tag{4.23}$$

where

$$\phi_{1,p} = -\Phi^{-1}\left[\frac{P_{MD}}{(1-P_{FA})^{r/p}}\right] = \Phi^{-1}\left[1 - \frac{P_{MD}}{(1-P_{FA})^{r/p}}\right] \simeq \Phi^{-1}\left[1 - P_{MD}\right] \tag{4.24}$$

since usually, $P_{FA} \ll 1$.

Approximations of $M_{1,K}$ and $S^2_{1,K}$, are obtained considering the exceeding of threshold only due to the signal part, and not due to the noise part. Also, it is assumed that the target occupies a single pixel. In such a case, we have:

$$F(x,y;u,v;k) \simeq f(x,y;k) + \alpha F(x,y;u,v;k-1) \simeq N(M_{1,k}, S^2_{1,k}) \tag{4.25}$$

It can be easily shown that:

$$M_{1,K} \simeq \frac{1-\alpha^K}{1-\alpha}\mu_s \, , \; S^2_{1,K} \simeq \frac{1-\alpha^{2K}}{1-\alpha^2}\sigma_s^2 \tag{4.26}$$

4.6 Calculation of required SNR

To calculate the SNR required for detection at particular rates of false alarms and mis-detections, equations (4.18) and (4.23) are combined to give:

$$M_{1,K} - M_{0,K} = S_{0,K}\,\phi_{0,p} + S_{1,K}\,\phi_{1,p} \tag{4.27}$$

Using expressions for $S_{0,K}$, $M_{0,K}$, $S_{1,K}$, and $M_{1,K}$, and assuming $\mu_n = 0$, $\mu_s = \mu$, and $\sigma_n = \sigma_s = \sigma$, equation (4.27) becomes:

$$\begin{aligned} \frac{1-\alpha^K}{1-\alpha}\mu - \frac{\sigma\alpha\mu_q}{\sqrt{1-\alpha^2\sigma_q^2}}\left[\frac{1-\alpha^K}{1-\alpha} - \gamma\alpha^{K-1}\frac{1-\alpha^K\sigma_q^{2K}}{1-\alpha\sigma_q^2}\right] \\ \simeq \sigma\sqrt{\frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2}}\,\phi_{0,p} + \sigma\sqrt{\frac{1-\alpha^{2K}}{1-\alpha^2}}\,\phi_{1,p} \end{aligned} \tag{4.28}$$

The SNR required for detection is given by:

$$\begin{aligned} SNR_T &= \frac{\mu}{\sigma} \simeq \frac{\alpha\mu_q}{\sqrt{1-\alpha^2\sigma_q^2}}\left[1 - \gamma\alpha^{K-1}\frac{1-\alpha}{1-\alpha^K}\cdot\frac{1-\alpha^K\sigma_q^{2K}}{1-\alpha\sigma_q^2}\right] \\ &+ \frac{1-\alpha}{1-\alpha^K}\sqrt{\frac{1-\alpha^{2K}\sigma_q^{2K}}{1-\alpha^2\sigma_q^2}}\,\phi_{0,p} + \sqrt{\frac{1-\alpha}{1+\alpha}\cdot\frac{1+\alpha^K}{1-\alpha^K}}\,\phi_{1,p} \end{aligned} \tag{4.29}$$

For $\alpha = 1$, replacing $(1-\alpha^K)/(1-\alpha)$ by K, we get

$$SNR_T = \frac{\mu}{\sigma} \simeq \frac{\mu_q}{\sqrt{1-\sigma_q^2}}\left[1 - \frac{\gamma}{K}\cdot\frac{1-\sigma_q^{2K}}{1-\sigma_q^2}\right] + \frac{1}{K}\sqrt{\frac{1-\sigma_q^{2K}}{1-\sigma_q^2}}\,\phi_{0,p} + \frac{1}{\sqrt{K}}\,\phi_{1,p} \tag{4.30}$$

For $K \to \infty, \alpha \neq 1$ such that $\alpha^K \ll 1$:

$$SNR_T = \frac{\mu}{\sigma} \simeq \frac{\alpha\mu_q}{1-\alpha^2\sigma_q^2} + \frac{1-\alpha}{\sqrt{1-\alpha^2\sigma_q^2}}\,\phi_{0,p} + +\sqrt{\frac{1-\alpha}{1+\alpha}}\,\phi_{1,p} \tag{4.31}$$

The above expressions of SNR_T can be written in the form:

$$SNR_T = A + B\,\phi_{0,p} + C\,\phi 1, p \tag{4.32}$$

where A, B, and C depend on K, q, and α. The terms B and C decrease with K, improving the algorithm performance as K increases. However, the term A *increases* with K, putting a lower bound on the *required* SNR, thus limiting the performance. It can be shown that this bound increases with q, and hence a lowest possible value of q should be used. This is intuitively explained, since a maximum is taken over q noise pixels and it is more likely to be a false alarm when q is large.

4.7 Temporal averaging and single frame thresholding as special cases

Recursive temporal averaging algorithm can be considered as a special case of dynamic programming with $p = q = 1$, for which $\mu_q = 0$ and $\sigma_q^2 = 1$. Hence, the threshold SNR for recursive temporal averaging becomes:

$$SNR_T = \frac{\mu}{\sigma} \simeq \sqrt{\frac{1-\alpha}{1+\alpha}\cdot\frac{1+\alpha^K}{1-\alpha^K}}\left[\phi_{0,1} + \phi_{1,1}\right] \tag{4.33}$$

This expression can also be obtained by using the recursive temporal averaging equations:

$$F(x,y;0) = 0\ ,\ F(x,y;k) = f(x,y;k) + \alpha F(x,y;k-1) \tag{4.34}$$

Also, for $\alpha = 1$, this expression takes the limit:

$$SNR_T = \frac{1}{\sqrt{K}}\left[\phi_{0,1} + \phi_{1,1}\right] \tag{4.35}$$

The same result would be obtained by using $\alpha = 1$ in original equations. For $K \to \infty, \alpha \neq 1$ such that $\alpha^K \ll 1$,

$$SNR_T = \sqrt{\frac{1-\alpha}{1+\alpha}}\left[\phi_{0,1} + \phi_{1,1}\right] \tag{4.36}$$

For single frame thresholding ($K = 1$ or $\alpha = 0$), the threshold SNR reduces to $\phi_{0,1}+\phi_{1,1}$.

Note that the first term from the dynamic programming algorithm disappears in these expressions, and there is no lower limit to the performance if $\alpha = 1$.

4.8 Theoretical performance plots

This section describes the behavior of the required signal to noise ratio SNR_T for different values of parameters. It should be noted that lower *required* SNR means *better* performance. Figure 4.1 (a) shows plots of SNR_T against K for dynamic programming algorithm with $p = q = 4$ and a number of values of α. The false alarm rate is 2×10^{-8} (0.02 per simulation for a 1 mega-pixel image), and the mis-detection rate is 0.001. It can be seen that SNR_T decreases with increase in K, but saturates at a certain point depending on α. Figure 4.1 (b) shows the corresponding plot for $p = q = 1$ – i.e., recursive temporal averaging. Figures 4.1 (c) and (d) show the plots of SNR_T against K with $\alpha = 1$ and $\alpha = 15/16$, respectively, for a number of values of p and q. It is observed that SNR_T increases with q as expected. The SNR_T also increases slightly with p, but the plots cannot show the change. Except in the case of $\alpha = 1$ and $p = q = 1$ – i.e., temporal averaging – the SNR_T saturates at some minimum value as $K \to \infty$.

4.9 Comparison between theoretical and observed performance

The parameters used in the calculation of theoretical performance of the algorithms for 2×2 targets are shown in Table 4.2. The calculated and the observed SNR threshold for these parameters for various algorithms are shown in Table 4.3.

Table 4.2. Parameters used for calculating the theoretical performance of algorithms.

Parameter	Dynamic prog	Single frame	Temp. Avg. (stat)
FA	$2 \times 10^{-8}/pixel = 0.02/image$		
MD	$0.001/pixel$		
α	15/16		
K	32	1	32
q	4	—	1

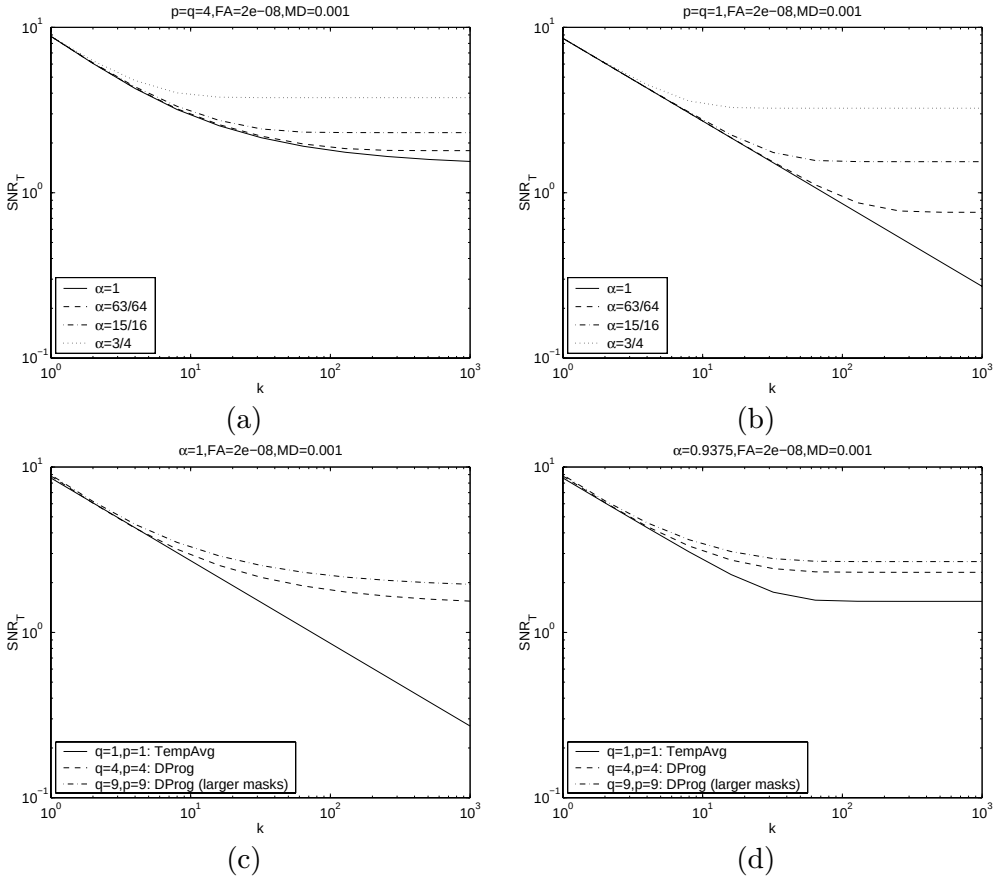


Fig. 4.1. Plots of SNR_T against K for: (a) $p = q = 4$ (dynamic programming) and number of α values. (b) $p = q = 1$ (temporal averaging) and number of α values. (c) $\alpha = 1$ and number of p and q values. (d) $\alpha = 15/16$ and number of p and q values. The parameters used are: $FA = 2 \times 10^{-8}, MD = 0.001$.

Table 4.3. Comparison of theoretical performance of the algorithms with observed performance on 2×2 targets.

Algorithm	Theoretical SNR	Observed SNR
Dynamic Prog	2.4540	2.04
Single frame	8.5811	8.17
Temp. Avg. (stat.)	1.7507	1.65

One can observe that the actual performance of the algorithm for 2×2 targets is slightly better than the theoretical performance for most of the algorithms. The reason for this is, that a 2×2 target occupies at least one pixel completely, and a few other pixels partially. Hence, its performance should be slightly greater than the calculated performance in which one assumes that the target occupies exactly one pixel.

To correct this problem, point targets were used in place of 2×2 targets. The experiments in Chapter 3 were repeated using point targets. The comparison between the calculated and observed SNR for a number of false alarm and mis-detection rates are shown in Table 4.4. It can be seen that the calculated and observed SNR rates agree very well in most cases. However, in the case of extremely low false alarm and mis-detection rates, the observed SNR is greater than the calculated SNR for the dynamic programming algorithm. The reason for this is the normal approximation used for the distribution of resulting output.

4.10 Effect of approximations

Approximations were used to derive the closed form expressions. In this section, the effects of these approximations are described.

Normal approximation

Normal approximation was used for maximum of q normal variables. The comparison of the probability density, and the complementary cumulative distribution functions of the maximum of $q = 4$ standard normal variables, and their normal approximation are shown in Figure 4.2. It can be seen that the approximation is good in the interior,

Table 4.4. Comparison of theoretical performance of the algorithms with observed performance on point targets for a number of different values of false alarm (FA) and mis-detection (MD) rates.

Algorithm	FA rate	MD rate	Theo. SNR	Obs. SNR
Dynamic Prog.	2×10^{-8}=0.02/simul	0.001	2.4540	2.7172
Dynamic Prog.	10^{-6}=1/simul	0.01	2.2313	2.2928
Dynamic Prog.	10^{-6}=1/simul	0.1	2.0181	1.9862
Dynamic Prog.	10^{-4}=100/simul	0.1	1.9259	1.8401
Temp. Avg.	2×10^{-8}=0.02/simul	0.001	1.7507	1.7355
Temp Avg.	10^{-6}=1/simul	0.01	1.4444	1.4307
Temp Avg.	10^{-6}=1/simul	0.1	1.2313	1.2345

where probability density is high, but is inaccurate in the tails, where the probability density is low.

Due to the difference in these distributions, the probability of false alarms is underestimated. In fact, to get the actual value of the false alarm rate, the function corresponding to the actual cumulative distribution of the output F should be used in place of cumulative normal distribution. But this distribution is difficult to obtain in closed form.

To get an idea of the difference between the actual distribution and the normal approximation, consider the function corresponding to the complementary cumulative distribution $Q(x) = Pr[X > x]$ of the maximum of $q = 4$ normal variables as shown in Figure 4.2. For $Q(x) = 10^{-8}$, we get $x = 5.85$, whereas for normal distribution the corresponding $Q_n(x) = 10^{-8}$ gives $x = 4.95$. The difference is around 18 % but is smaller for smaller values of x.

At each step of the recursion, maximum of q instances of F at time $k-1$ are taken and added to the input f at time k to obtain the output F at time k. Hence, the distribution of the output F at each time should be a better approximation of normal distribution than $Q(\cdot)$, since a normal variable (f) is added to the maximum term for obtaining the output F. Also, since the normal approximation for F is good in the interior, the mean and variance of maximum of q instances of F will be close to what is

computed assuming the normal distribution. Hence, the mean and variance calculations are not affected much.

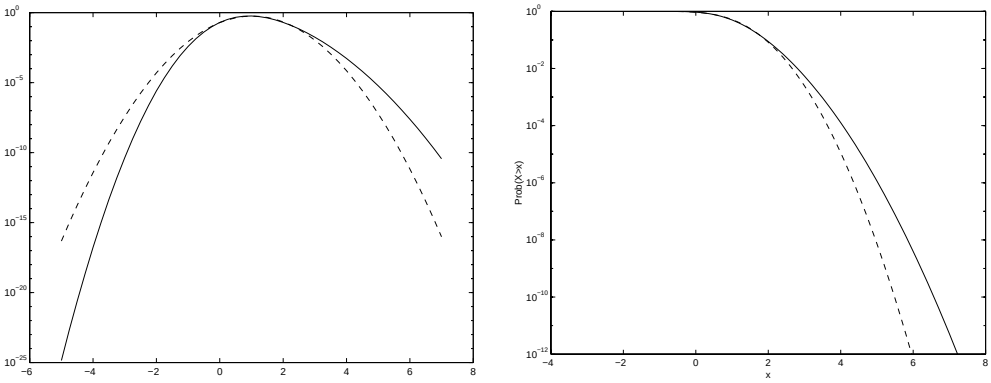
Furthermore, it is observed that the threshold SNR changes are small even for large changes in false alarm and mis-detection rates. In any case, one would not directly use the false alarm and mis-detection rates during the application of the algorithm, but estimate these dynamically using the output from the algorithm.

Approximation in false alarm estimation

Another approximation was performed while computing the mean value $M_{0,K}$ of the noise output, used in false alarm estimation. For equation (4.12), γ actually depends on k, which makes it impossible to get an exact analytical expression. It was assumed that γ is fixed and approximately equal to $1/2$, corresponding to a first order approximation. However, it is observed that the value of $M_{0,K}$ does not change much with γ even for the extremes of $\gamma = 0$ or $\gamma = 1$. Hence, the approximation is reliable.

Approximation in mis-detection estimation

In the case of mis-detections, the output of the algorithm at a target point is assumed to be solely due to the target, without the effect of noise. The noise can add or subtract the target intensity. However, since maximum is taken over q pixels at every stage, bias is likely towards adding. Hence, the mis-detections are likely to be less than what are estimated.



(a) (b)

Fig. 4.2. Probability distributions of normal approximations: (a) Probability density function ($p(x)$ and $p_n(x)$) (b) Complementary cumulative distribution ($Q(x)$ and $Q_n(x)$) against x of the maximum of $q = 4$ standard normal variables (solid line) and the normal approximation having same mean and variance.

Chapter 5

A Special Approach for Hazard Detection

It is well known in the pilots' community, that an object on a collision or near-collision course remains stationary or nearly stationary in its 2-D image view [38]. The closest distance that an aircraft would approach another before moving away from it, is known as the distance of passage, and the time to reach that point is known as the time to passage, or time to 'collision'. For ensuring safety, the distance of passage should be larger than a certain limit; and objects with a smaller distance of passage should be detected before the time to collision becomes too small. It can be shown that the rate of translation of the object in the image is proportional to the distance of passage. Using this property, the rate of image translation can be used to separate hazardous objects from clutter, since the former have a smaller rate of translation.

Another useful property which can be used to discriminate hazardous objects from clutter is the rate of image expansion, which is approximately inversely proportional to the time to collision of the object. Nelson and Aloimonos [46] use the image expansion in terms of the flow field divergence to estimate the time to collision, for separating obstacles. Francois and Bouthemy [22] separate the image motion into components of divergence, rotation, and deformation. Ancona and Poggio [3] use 1-D correlation to estimate optical flow for a time-to-crash detector. Baram and Barniv [7] rely on object texture to extract information on local expansion. Instead of estimating a numerical depth value, an object is classified as 'safe' or 'dangerous' using a pattern recognition approach.

Most of these methods are useful for objects of larger sizes. However, in this case, the object sizes can be very small, even sub-pixel, along with very small rates of expansion. Hence, a feature based approach was used in this work, where features were tracked, and their expansion estimated over a large number of frames.

This chapter describes the conditions under which the rates of image translation and expansion can be used to separate an object on collision course from the ground clutter. Methods to estimate the image translation and expansion are proposed and tested on real image sequences obtained from a camera mounted on an aircraft.

5.1 Scene geometry

Consider an object approaching towards the aircraft with a *relative* velocity of V as shown in Figure 5.1 (a). Let p be the distance of passage – i.e., the closest distance that the object approaches the camera – and ϕ be the angle between the line of sight of the target and the relative velocity vector V. Let τ denote the time to passage (or collision) which is the time the object takes to reach the distance of passage. The object distance is r, whereas distance that the object travels until it reaches the point of passage P is z.

5.2 Detection using translation

As the object moves, the angle ϕ as well as distances r and z change, but the distance of passage p is constant. The rate of angular translation of an object in the image is $T = \dot{\phi}$. The pixel translation is approximately given by multiplying the angular translation by the focal length. By geometry of Figure 5.1 (a), we have:

$$z = p \cot \phi \tag{5.1}$$

To find the rate of translation $\dot{\phi}$, this expression is differentiated to get:

$$\dot{z} = -p(\csc^2 \phi)\dot{\phi} \tag{5.2}$$

The magnitude of the relative velocity V is the rate of decrease of z, given by:

$$V = -\dot{z} = p(r/p)^2\dot{\phi} \tag{5.3}$$

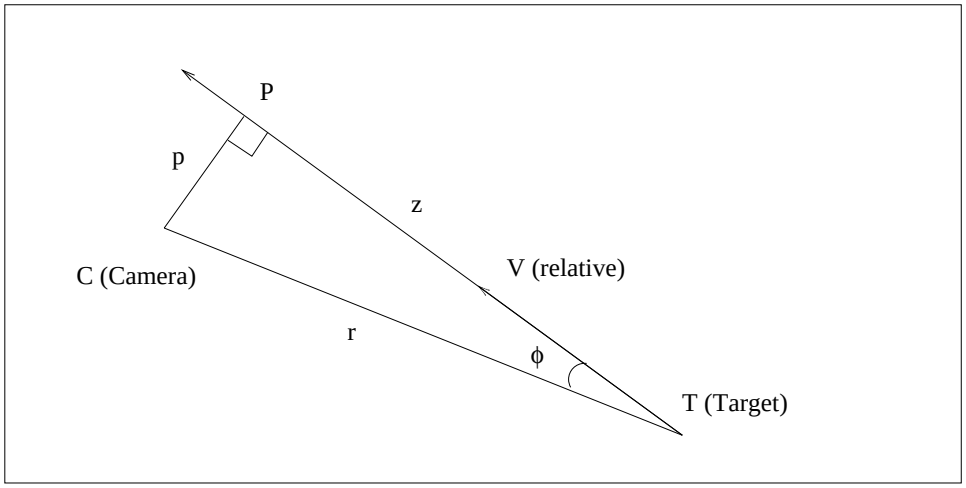
Also, the time of passage is given by:

$$\tau = z/V = r \cos \phi / V \tag{5.4}$$

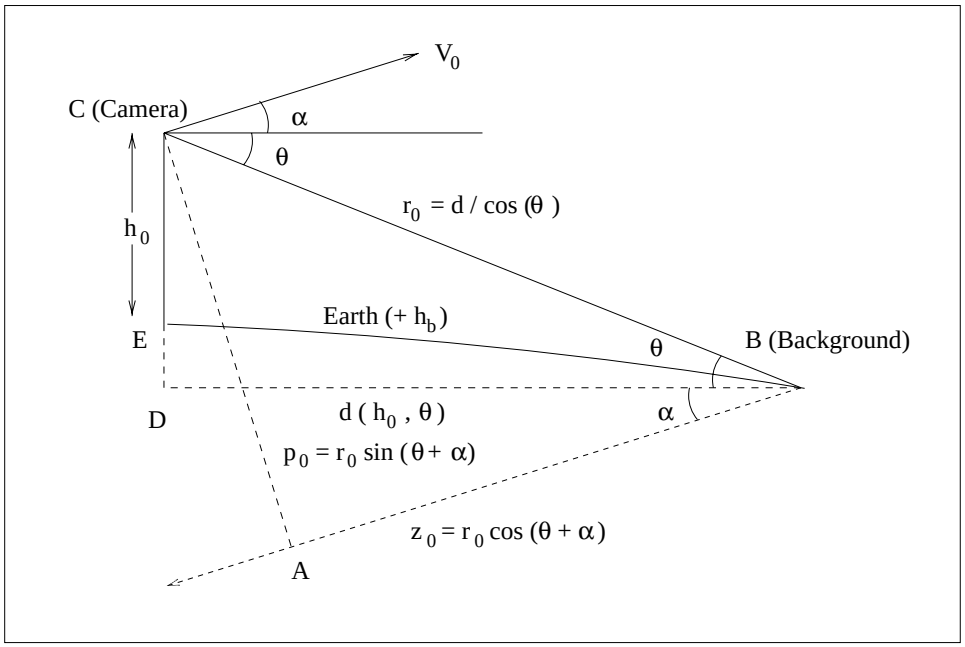
From equations (5.3) and (5.4), the rate of target translation is given by:

$$T = \dot{\phi} = \frac{pV}{r^2} = \frac{p \cos \phi}{\tau r} \tag{5.5}$$

Thus, the rate of image translation is proportional to the distance of passage, and the objects on a collision course are likely to have a smaller rate of translation compared



(a)



(b)

Fig. 5.1. Geometry of (a) target (b) background moving relative to the camera.

to other objects. However, this rate is also dependent on the target distance, and a nearer target moves faster in the image than a farther target with the same distance of passage. If S_{min} is the smallest visible dimension that an object can have, the corresponding size in the image is given by:

$$s \geq s_{min} = S_{min}/r \tag{5.6}$$

Hence, from equation (5.5), one can write:

$$\frac{T}{s} \leq \frac{p \cos \phi}{\tau S_{min}} \leq \frac{p}{\tau S_{min}} \tag{5.7}$$

Hence, an object on a near collision course, having sufficient time before imminent collision has the ratio of its image motion to its image size bounded by the above precomputable limit. For example, if the distance of passage of $p = 150\,m$ $(500\,ft)$ is allowed, and an object of smallest size of $S_{min} = 1.2\,m$ $(4\,ft)$ is to be detected before $\tau = 25$ seconds (750 frames), then this ratio becomes 1/6 – i.e., the image motion per frame is at the most $1/6^{th}$ of the image size of the object. However, in actual practice, a larger range of velocities should be checked, to have a safety margin.

It should be noted that the above relationship is valid only if the aircraft does not rotate or vibrate around its own axes. If there is rotation, it should be compensated by using the data from the aircraft navigation system. In the absence of this data, it may be possible to use image features due to clutter (if available) to perform the compensation, by modeling their image motion due to camera rotation.

If this compensation is successful, the velocity to size ratio of the object would be bounded. By reducing the image resolution to an appropriate level, the image velocity of the object would also be restricted. Hence, using pyramid construction, target detection can be performed at a number of resolutions, and the suitable resolution selected. This also leads to spatio-temporal integration of the image data and the amplification of SNR which could enable detection of sub-pixel or low-contrast objects in uniform background, such as clear or overcast sky.

The relationship between image motion and the distance of passage can be used to remove the clutter which is not on collision course and thus expected to have a large image motion. However, the image motion is inversely proportional to the distance of the object from the camera. Thus, if clutter is at a large distance, it too could have a small image motion. The conditions under which an object on the collision course can be distinguished from ground clutter at the same image position are derived below.

Let r_0 and p_0 denote the background distance, and the minimum distance of approach for the background, respectively, as shown in Figure 5.1 (b). The relative velocity V_0 between the camera and the background is actually the magnitude of the camera velocity, By substituting these parameters in equation (5.5), the rate of background translation can be written as:

$$T_0 = \frac{p_0 V_0}{r_0^2} \tag{5.8}$$

Let $h_0 = h_c - h_b$ denote the difference between the camera altitude h_c and the background altitude h_b. Also, the angle of the camera velocity above the horizontal (not horizon) is α. From Figure 5.1 (b), we have:

$$r_0 = d \sec\theta \tag{5.9}$$

$$p_0 = r_0 \sin(\theta + \alpha) \tag{5.10}$$

Here, d is a function of the relative height h_0 and the angle θ. If the earth were flat (or θ is large), refraction of light is negligible, and the terrain is smooth, the dotted line corresponding to d would coincide with the surface of the earth, and we would have

$$d = h_0 \cot\theta$$

However, if we express:

$$d(h_0, \theta) = h_0 \cot\theta f(h_0, \theta) \tag{5.11}$$

then the effects of the earth's curvature and refraction of light ray would be incorporated in the function f. If these factors can be neglected, then $f(h_0, \theta) \simeq 1$. The expression for f using the curvature of the earth is derived in Section 5.4. Also, using equation (5.9), one can write:

$$r_0 = h_0 \csc\theta f(h_0, \theta) \tag{5.12}$$

Substituting equations (5.10) and (5.12) in (5.8), the rate of background translation T_0 is given by:

$$T_0 = \frac{V_0 \sin(\theta + \alpha)}{r_0} = \frac{V_0 \sin(\theta + \alpha) \sin\theta}{h_0 f(h_0, \theta)} \tag{5.13}$$

If the hazard is to be discriminated from the background in the same line of sight, the rate of translation of the hazard must be much smaller than that of the background – i.e., $T \leq \eta_t^{-1} T_0$ with $\eta_t > 1$, having a larger value for greater discriminating power.

Using equations (5.5) and (5.13), we have:

$$\frac{p\cos\phi}{\tau r} \leq \eta_t^{-1}\frac{V_0\sin(\theta+\alpha)\sin\theta}{h_0 f(h_0,\theta)} \tag{5.14}$$

Hence, the object distance r should be larger than the following expression:

$$r \geq \frac{\eta_t p h_0 f(h_0,\theta)\cos\phi}{\tau V_0\sin(\theta+\alpha)\sin\theta} = \frac{\eta_t p D f(h_0,\theta)\sqrt{1-Q^2}}{\sin(\theta+\alpha)\sin\theta} \tag{5.15}$$

with

$$D = \frac{h_0}{\tau V_0}\,,\; Q = \frac{p}{r} = \sin\phi\,,\; \cos\phi = \sqrt{1-Q^2} \simeq 1 \text{ (for } p \ll r) \tag{5.16}$$

Hence, θ should satisfy:

$$\sin(\theta+\alpha)\sin\theta \geq \eta_t D Q\sqrt{1-Q^2} f(h_0,\theta) \tag{5.17}$$

Also, using $T \leq \eta_t^{-1}T_0$, with equations (5.5) and (5.13), one can write:

$$\frac{p\cos\phi}{\tau r} \leq \eta_t^{-1}\frac{V_0\sin(\theta+\alpha)}{r_0} \tag{5.18}$$

Since the object distance cannot be greater than the background distance in the line of sight, $r \leq r_0$. Hence, one can also write:

$$\sin(\theta+\alpha) \geq \frac{\eta_t p\cos\phi}{\tau V_0}\frac{r_0}{r} \geq \frac{\eta_t p\sqrt{1-Q^2}}{\tau V_0} \tag{5.19}$$

For $p \ll r$ or $Q \ll 1$, this condition is approximately independent of r. It can be said that for detection to be possible at all for a particular θ and α, the above condition is necessary irrespective of the target distance r, provided it is sufficiently large.

If the curvature of the earth and the refraction of light can be neglected, then $f \simeq 1$. The necessary condition in equation (5.19) does not simplify. However, equation (5.17) reduces to:

$$\sin(\theta+\alpha)\sin\theta \geq \eta_t D Q\sqrt{1-Q^2} \tag{5.20}$$

On solving for θ, this yields:

$$\theta \geq \frac{1}{2}\left[\cos^{-1}\left(-2\eta_t D Q\sqrt{1-Q^2}+\cos\alpha\right)-\alpha\right] \tag{5.21}$$

If $\alpha = 0$, the solution for θ is simpler:

$$\theta \geq \sin^{-1} \sqrt{\eta_t D Q \sqrt{1 - Q^2}} \tag{5.22}$$

For example, if we have:

$$p = 150\,m,\ \tau = 25\,s,\ V_0 = 150\,m/s,\ h_0 = 1\,km,\ \alpha = 0,\ \eta_t = 2.5 \tag{5.23}$$

For these values $D = 0.267$, and from equation (5.19) the necessary condition is $\theta \geq 5.7^\circ$. This condition corresponds to the target being at the same position as the background, which is $r = r_0 = 10\,km \simeq 5.4\ nmi$ or $Q = 0.015$, using equation (5.12). However, if the target is nearer, the condition on θ is determined by equation (5.17) or (5.20). For example, if a hazard should be detected at $r = 5\,km \simeq 2.7\,nmi$ or $Q = 0.03$, one would really need $\theta \geq 8.1^\circ$. The required θ increases as r decreases.

5.3 Detection using expansion

Another discriminating feature between objects on collision course, and objects much farther, is the time to collision. It is well known that the rate of image expansion, – i.e., the increase of the image size of an object – is inversely proportional to the time to collision.

In Figure 5.1 (a), as the object comes closer to the camera along the line of z, its size in the image will become larger. The rate of this expansion of any object is defined as the ratio of the rate of increase in its size to the size at that time, – i.e., $E = \dot{s}/s$ – where s is the size of the object in the image. Since $s = S/r$ where S is the object size which is assumed constant, we have $\dot{s} = -S\dot{r}/r^2$, and

$$E = -\dot{r}/r \tag{5.24}$$

By geometry of Figure 5.1 (a),

$$r^2 = z^2 + p^2 \tag{5.25}$$

To find the rate of expansion, this expression is differentiated to yield:

$$2r\dot{r} = 2z\dot{z} = -2zV \tag{5.26}$$

Hence, rate of target expansion is given by:

$$E = -\frac{\dot{r}}{r} = \frac{zV}{r^2} = \frac{V\cos\phi}{r} = \frac{\cos^2\phi}{\tau} \tag{5.27}$$

where the time to passage is:

$$\tau = z/V = r\cos\phi/V \tag{5.28}$$

For $\tau = 25\,s = 750$ frames, the ratio is 0.13 % per frame, which is a very small magnitude. This small expansion can be measured by tracking it over a large number of frames.

For estimating the rate of expansion of the background, the corresponding parameters for the background are substituted in equation (5.27) to give:

$$E_0 = \frac{z_0 V_0}{r_0^2} \tag{5.29}$$

Using $z_0 = r_0\cos(\theta+\alpha)$ with equations (5.9) and (5.11), the rate of background expansion can be written as:

$$E_0 = \frac{V_0\cos(\theta+\alpha)}{r_0} = \frac{V_0\cos(\theta+\alpha)\cos\theta}{d} = \frac{V_0\cos(\theta+\alpha)\sin\theta}{h_0 f(h_0,\theta)} \tag{5.30}$$

If reliable discrimination of the hazard from the background in the same line of sight is required, the rate of expansion of the hazard must be much larger than that of the background, – i.e., $E \geq \eta_e E_0$ with $\eta_e > 1$, having a large value for greater discriminating power. Using equations (5.27) and (5.30), one needs:

$$\frac{\cos^2\phi}{\tau} \geq \eta_e \frac{V_0\cos(\theta+\alpha)\sin\theta}{h_0 f(h_0,\theta)} \tag{5.31}$$

or

$$\cos(\theta+\alpha)\sin\theta \leq \frac{h_0 f(h_0,\theta)\cos^2\phi}{\eta_e \tau V_0} = \eta_e^{-1} D(1-Q^2) f(h_0,\theta) \tag{5.32}$$

where D and Q are given by equation (5.16). For the case of $f \simeq 1$, the equation (5.32) reduces to:

$$\cos(\theta+\alpha)\sin\theta \leq \eta_e^{-1} D(1-Q^2) \tag{5.33}$$

Explicit solution for θ is then given by:

$$\theta \leq \frac{1}{2}\left[\sin^{-1}\left(2\eta_e^{-1}D(1-Q^2)+\sin\alpha\right)-\alpha\right] \tag{5.34}$$

For the conditions in equation (5.23), we need $\theta \leq 6.2^\circ$ for reliable detection using expansion.

It should be noted that the expansion in image size can also be caused by the rotation of the target aircraft in a way which would expose a larger area to the camera. However, this false expansion takes place only in the direction perpendicular to the axis of rotation of the target aircraft, whereas the expansion due to a potential collision would take place uniformly in all directions. Also, the target expansion will cease after the aircraft fully rotates to a position where maximum area is exposed to the camera. It may be possible to use these properties to discriminate between the false expansion and the expansion due to a collision course.

5.4 Effect of horizon

In this section, function describing the effect of the curvature of the earth is calculated, neglecting the effects of refraction. Figure 5.2 shows the geometry of the earth's curvature. The coordinates used are with respect to earth's center. Using this, we have:

$$d = R\sin\gamma,\ d\tan\theta = h_0 + R(1-\cos\gamma) \simeq h_0 + d^2/(2R) \tag{5.35}$$

where $R = R_0 + h_b$, h_b is the altitude of the background, R_0 is the radius of earth, and γ is the angle subtended on the center of the earth by the triangle. Solving this equation yields:

$$d = R\left[\tan\theta \pm \sqrt{\tan^2\theta - 2h_0/R}\right] \tag{5.36}$$

The correct solution is the smaller value of d, since the larger value represents the other intersection of the line of sight with the earth.

$$d = R\left[\tan\theta - \sqrt{\tan^2\theta - 2h_0/R}\right] = \frac{2h_0}{\tan\theta + \sqrt{\tan^2\theta - 2h_0/R}} \tag{5.37}$$

By substituting in equation (5.11), we have:

$$f(h_0,\theta) = \frac{2}{1+\sqrt{1-2h_0/(R\tan^2\theta)}} \tag{5.38}$$

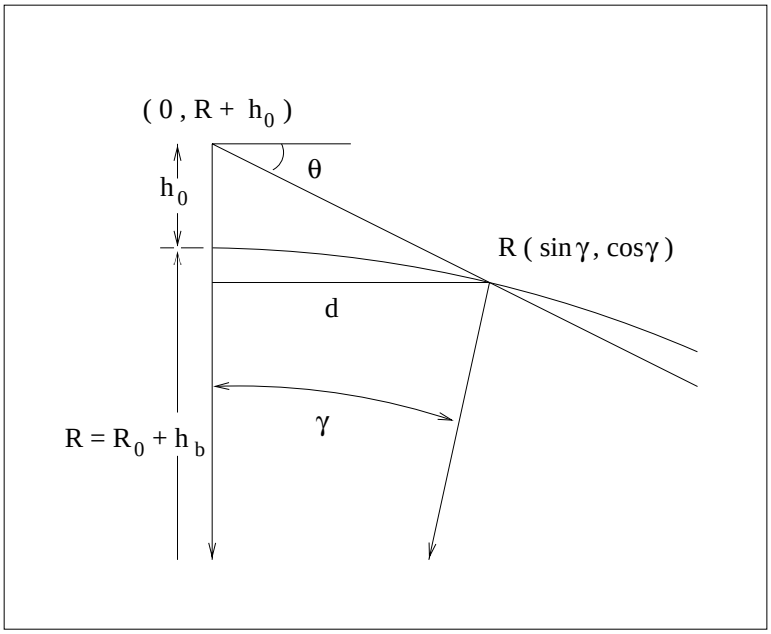


Fig. 5.2. Geometry of earth's curvature: The coordinates used are with respect to earth's center.

If $\theta \simeq \pi/2$, or R is large, h is small, then $f \simeq 1$, – i.e., the earth's curvature can be neglected. However, where the line of sight just touches the earth – i.e., at the horizon – the discriminant under the square root is zero, then $f = 2$ and the corresponding θ is:

$$\theta_h = \tan^{-1} \sqrt{2h_0/R} \tag{5.39}$$

Any value of θ smaller than this value corresponds to the line of sight not touching the earth – i.e., background above the horizon.

5.5 Behavior of translation and expansion

Figure 5.3 shows the variation of the required θ with the horizontal, for the possibility of detection using translation and expansion, against various parameters. Effect of horizon was neglected since it was observed that it does not affect the plots to a significant extent. The minimum θ for detection using translation, which is shown by dashed line, whereas the maximum θ for detection using expansion is shown by dotted line. However, the minimum θ criterion is only the necessary criterion. For actual discrimination using translation for an object at a given distance, a larger θ is required. The other curves show the required θ for detection using translation for various object distances in meters, and are enveloped by the dashed line curve.

Most of the information in these curves can be condensed using the parameter $D = h_0/(\tau V_0)$. Figure 5.4 (a) shows the contours of same D for different values of V_0 and h_0 for $\tau = 25\,s$. Plots of required θ for translation and expansion using a number of values of the target distance r in km, for the distance of passage $p = 150\,m$ are shown in Figure 5.4 (b). However, the necessary criterion for translation cannot be expressed using these plots.

5.6 Estimation of translation and expansion

To reduce the computational complexity of estimating the translation and expansion, a feature-based approach was used. A morphological filter [17] which subtracts the opening and closing of the image from the original image was used to detect positive and negative features, corresponding to light and dark objects, respectively.

To estimate the translation of the features over a number of frames, they were tracked over a number of frames. In case of navigation system data being available, the position of the features were compensated before performing the tracking. A nearest

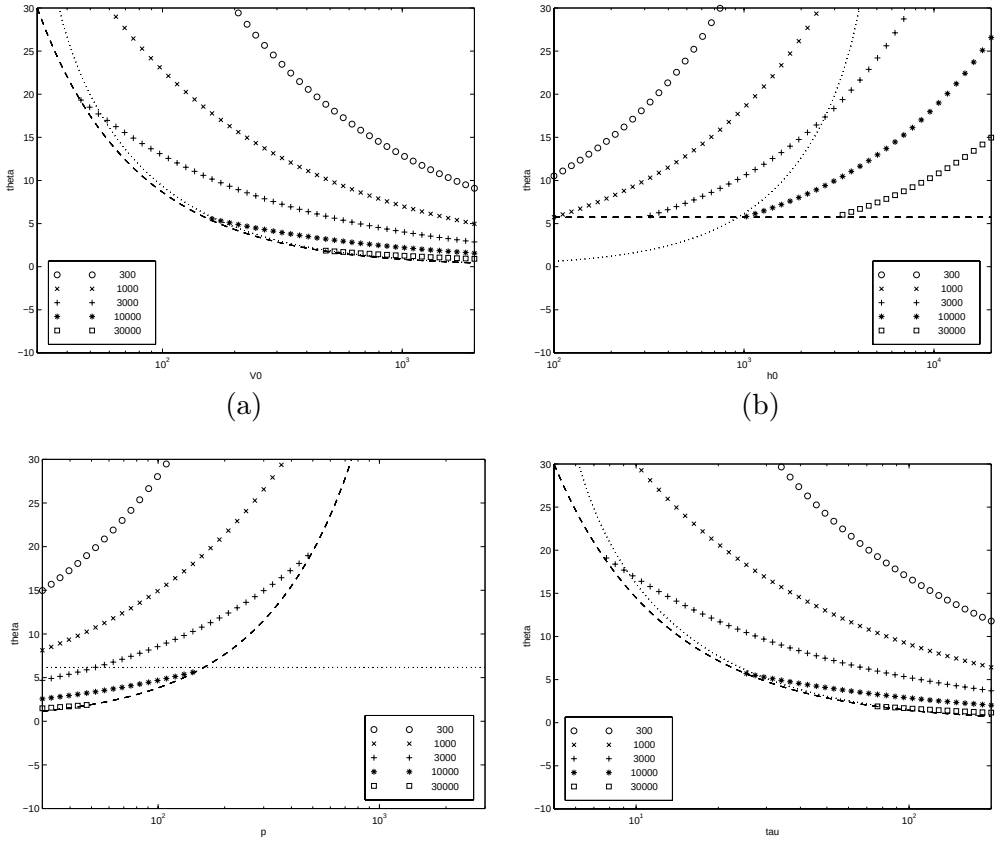
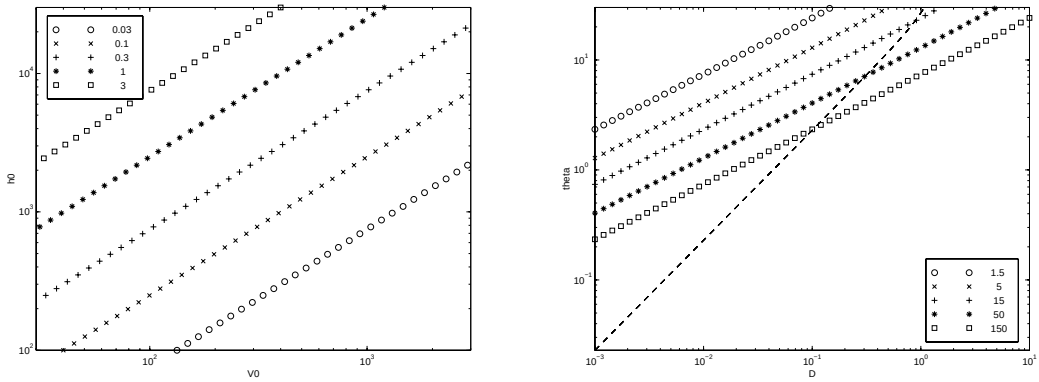


Fig. 5.3. Variation of the required θ with the horizontal, for the possibility of detection using translation and expansion, against a number of parameters: (a) Camera velocity: V_0, (b) Relative height between camera and background: h_0, (c) Distance of passage: p (d) Time of passage (or collision): τ. Default values of the parameters (except when they vary) are: $V_0 = 1\,km/s$, $h_0 = 1\,km$, $p = 150\,m$, and $\tau = 25\,s$, and $\eta_t = \eta_e = 2.5$. The minimum θ for detection using translation is shown by dashed line, whereas the maximum θ for detection using expansion is shown by dotted line. The other curves show the required θ for translation for various object distances in meters.



(a) (b)

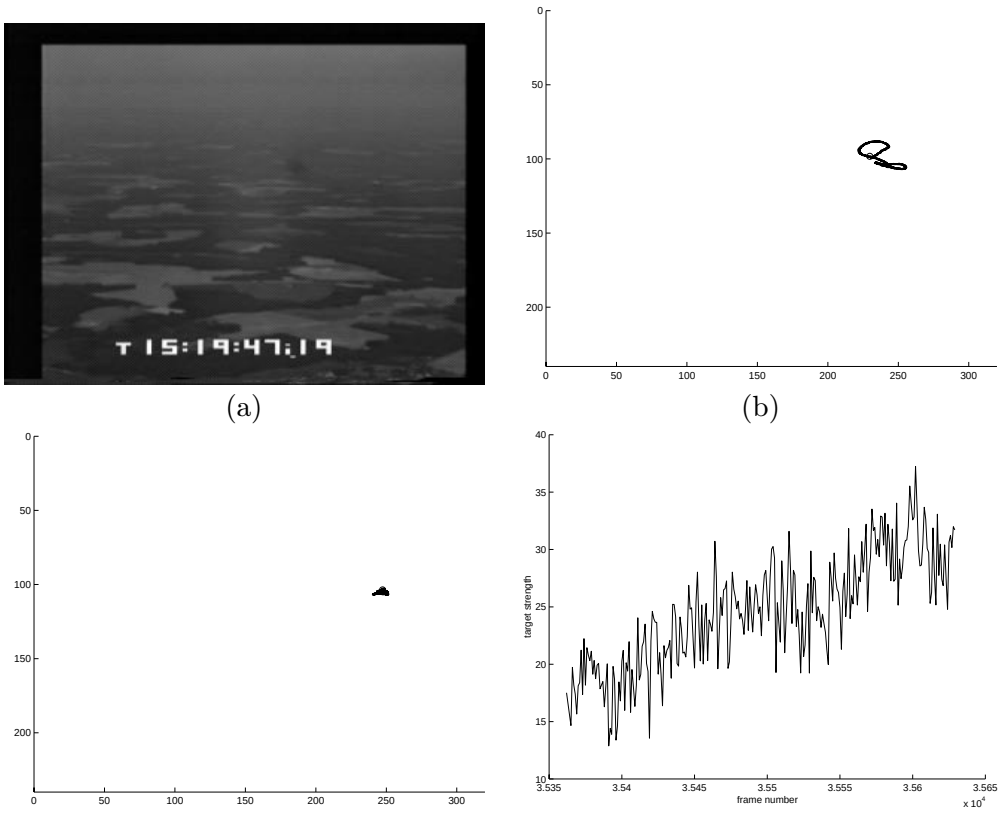
Fig. 5.4. Plots for detection using translation and expansion: (a) Plot showing the contours of same D for different values of V_0 and h_0 for $\tau = 25\,s$ (b) Plots of θ required for detection using translation are shown with various symbols for a number of values of the target distance r in km, for the distance of passage $p = 150\,m$. Plot of the θ required for detection using expansion is shown with a dashed line.

neighbor approach was used to determine the corresponding feature in the next frame, and the smoothed estimates of the feature position and velocity in each frame were obtained using Kalman filter approach. This procedure is similar to the one described in Chapter 7 used for detecting targets crossing the aircraft.

For detecting expansion, a 15×15 window around each feature was explored. The sub-image corresponding to the window was thresholded, and the connected component containing the center of the window was found. All the pixels in the sub-image that did not belong to the component were set to zero. To estimate the size of the component, the sub-image was convolved with a number of smoothing masks. These masks perform matched filtering with a object templates corresponding a number of different sizes. The maximum output from all these masks was considered as the measure of target strength. The rate of expansion was measured in terms of increase of the target strength, tracked over a number of frames. The target strength was plotted against the frame number, and the rate of expansion was estimated by applying least squares to the logarithm of the target strength.

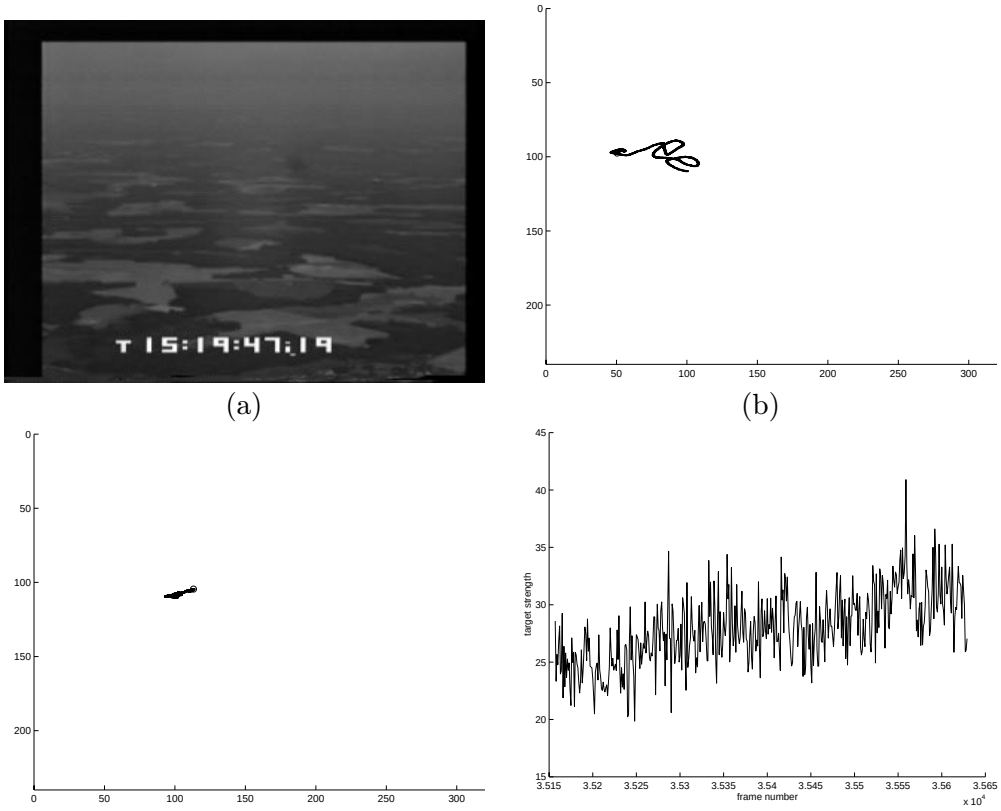
5.7 Results

The estimation of translation and expansion was performed on a sequence of images captured from an analog camera in which the target aircraft is approaching the camera. Figure 5.5 (a) shows a typical frame from the sequence. Figure 5.5 (b) and (c) show the target track in the original and the motion compensated images, respectively. Figure 5.5 (d) shows the plot of the estimated target size against the frame number. Corresponding plots for two clutter tracks are shown in Figures 5.6 and 5.7. It can be seen that the target expansion is the large for the target track, and small for the clutter tracks. On the other hand, the rate of target translation is small for the target track and large for the clutter tracks. Figure 5.8 shows the significant tracks before and after motion compensation. A scatter plot of the feature expansion against translation for these tracks, including the target track is shown in Figure 5.9. The rate of translation is measured in terms of the displacement magnitude of the compensated features in 100 frames, whereas the rate of expansion is measured in terms of the increase in the logarithm (to base 10) of the target strength in 100 frames. It is seen that the target has a large rate of expansion and a small rate of translation and is located in the upper left corner of the scatter plot.



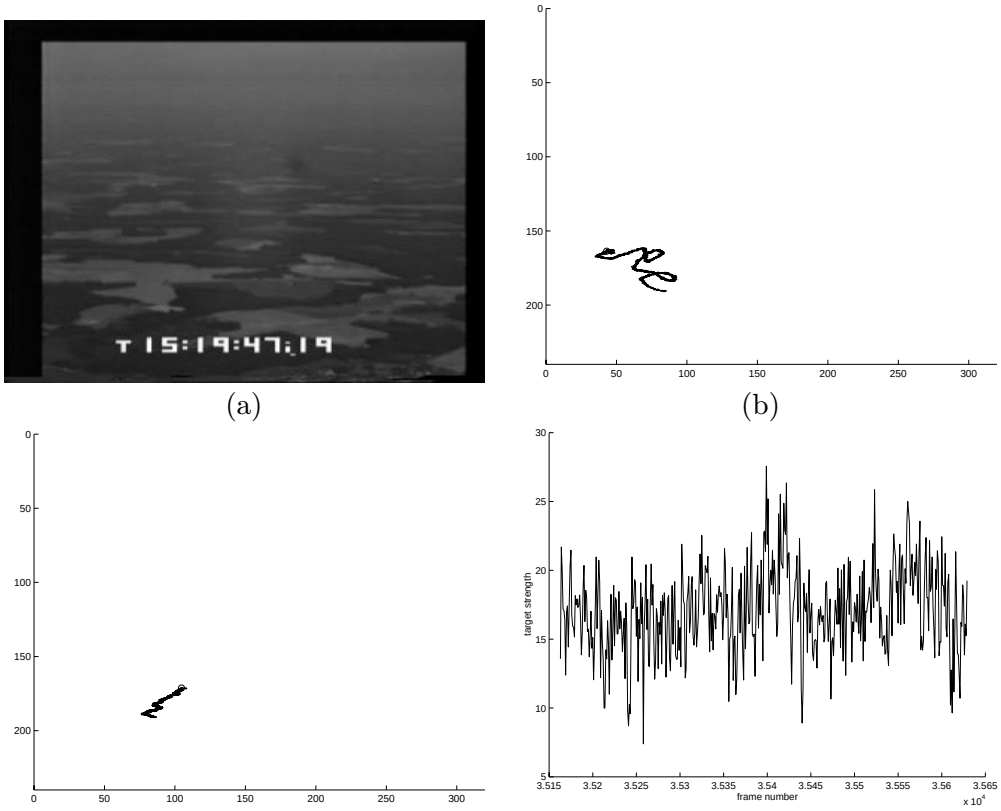
(a) (b) (c) (d)

Fig. 5.5. Translation and expansion for target track: (a) Sample image from the last frame. (b) Target track (c) Target track after compensation. Rate of translation is small for target track. (d) Plot of expansion against frame number. Rate of expansion is large for target track.



(a) (b) (c) (d)

Fig. 5.6. Translation and expansion for clutter track: (a) Sample image from the last frame. (b) Target track (c) Target track after compensation. Rate of translation is large for clutter track. (d) Plot of expansion against frame number. Rate of expansion is small for clutter track.



(a) (b)

(c) (d)

Fig. 5.7. Translation and expansion for another clutter track: (a) Sample image from the last frame. (b) Target track (c) Target track after compensation. Rate of translation is large for clutter track. (d) Plot of expansion against frame number. Rate of expansion is small for this clutter track.

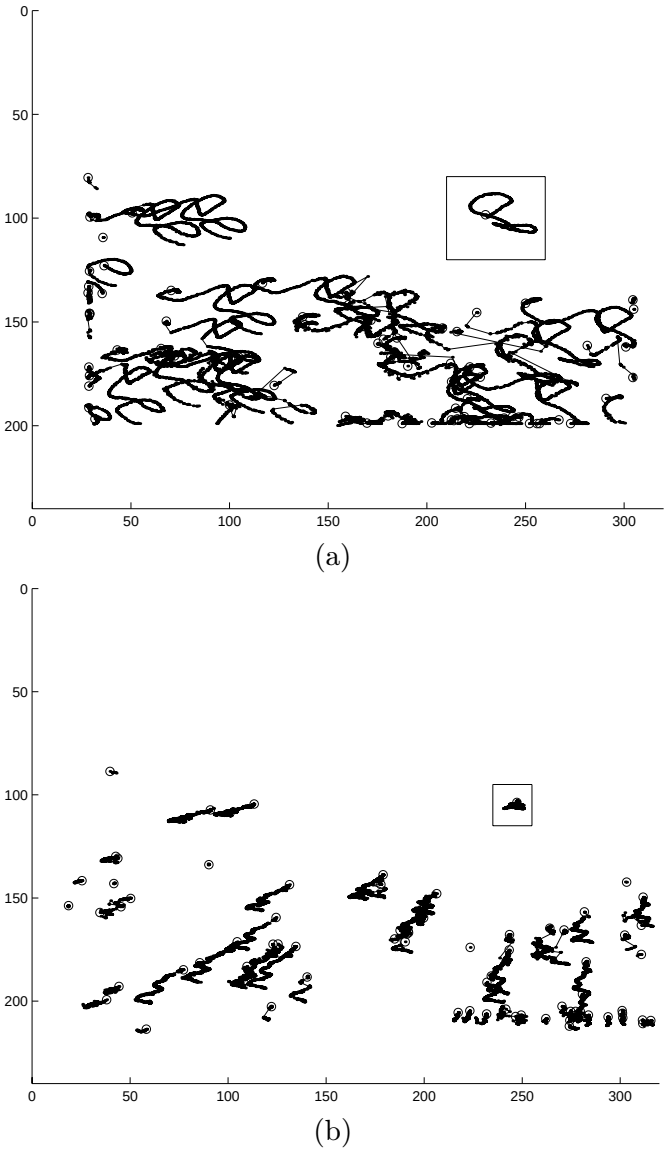


Fig. 5.8. Feature tracks (a) before, and (b) after rotation compensation: Target track surrounded by a rectangle has a small translation after compensation.

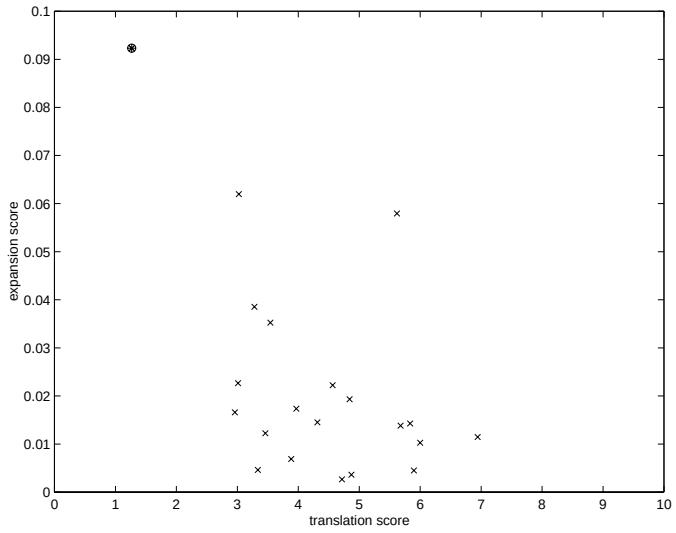


Fig. 5.9. Scatter plot of the feature expansion against translation: The rate of translation is measured in terms of the displacement magnitude of the compensated features in 100 frames, whereas the rate of expansion is measured in terms of the increase in the logarithm (to base 10) of the target strength in 100 frames. The target is marked as an encircled asterisk, and is in upper left corner, having a small rate of translation and a large rate of expansion.

Chapter 6

Algorithm Fusion

Each of the target detection algorithms has its own advantages and limitations. Hence, a combination of these algorithms may be used in the ultimate design to overcome their individual limitations while maximizing their advantages. This chapter describes a method for combining the algorithms using statistical approach to optimize the performance in terms of the mis-detection and false alarm rates. In particular, the pre-processing algorithms of low-stop and morphological filters, described in Chapter 2 are combined. The performance of the fused algorithm is compared with the original algorithms using the methodology described in Chapter 3.

6.1 Combination of algorithms using a statistical approach

According to the Neyman Pearson criterion, the optimal Bayesian detector which minimizes the rate of mis-detection for a particular rate of false alarms is obtained by thresholding the joint likelihood ratio of the individual detector outputs, or some monotonic function of the same. The threshold should be such that the desired false alarm rate is obtained.

Consider the joint likelihood ratio of the low-stop and the morphological filter. Let $z = (z_1, z_2)$ be the 2-D vector denoting the outputs of the low-stop and the morphological filters, respectively. Let $p(z|H_0, C)$ and $p(z|H_1, C)$ denote the joint probability density functions for the hypotheses denoting the absence and presence of a target, respectively, for clutter level estimate C. The likelihood ratio is then given by:

$$L_{H,C}(z) = \frac{p(z|H_1, C)}{p(z|H_0, C)} \tag{6.1}$$

6.2 Statistical behavior of low-stop and morphological filters

In the following analysis, it is assumed that the input image pixels are described by the sum of the signal θ, background level β, and the camera noise ν, which is modeled

as an uncorrelated Gaussian noise of zero mean and variance η^2.

$$x = \theta + \beta + \nu \tag{6.2}$$

If there is no clutter, the distributions of x in absence and presence of the target are given by:

$$p(x|H_0) \sim N(\beta, \eta^2)\ ,\ p(x|H_1) \sim N(\theta + \beta, \eta^2) \tag{6.3}$$

If clutter is present, the exact distributions would depend on the nature of the clutter. Here, it is assumed that the presence of clutter changes the mean background level, and the variance parameter of the noise, making these parameters space varying.

Low-stop filtering is performed by subtracting the low-pass filtered image, using a weighted spatial average of the neighborhood, from the original image. This filter attempts to subtract the background level. Since it is a linear filter, if the input is normally distributed, the output z_l will also be distributed as:

$$p(z_l|H_0) \sim N(0, \sigma_l^2)\ ,\ p(z_l|H_1) \sim N(\mu_l, \sigma_l^2) \tag{6.4}$$

with

$$\sigma_l = f_l \eta\ ,\ \mu_l = g_l \theta \tag{6.5}$$

where f_l and g_l are the amplification gains in the standard deviation and mean due to the filter. It should be noted that the background level β is subtracted out by the filter.

Morphological filtering is performed by taking the difference between the original image and its opening (positive targets) or closing (negative targets). Without loss of generality, only positive targets are considered, which are detected by subtracting the opening from the original image. This is expected to remove uniform background, as well as most of the clutter.

To obtain a model for the distribution of the morphological filter and to verify the distribution of low stop filter, simulations were performed. A large number of floating point images containing Gaussian noise were generated. Low-stop and morphological filter were applied to these images, and the histograms of the filter outputs were obtained. Figure 6.1 (a) shows the histogram of the original image with Gaussian noise. Figure 6.1 (b) shows the histogram of the low-stop filter output, which is normally distributed with zero mean, as expected. The histogram of the morphological filter output

is shown in Figure 6.1 (c). It can be seen that the histogram resembles a normal distribution with a positive mean. However, since the opening of an image is always less than or equal to the original image, the filter output is always non-negative. Hence, the distribution is truncated on the negative side, and has an impulse at zero in place of the negative values. For clarity, the distribution after removing the impulse is shown in Figure 6.1 (d).

This distribution can be modeled by using a hypothetical normally distributed variable $\xi_m \sim N(\mu_m, \sigma_m^2)$. The output z_m of the morphological filter can be expressed in terms of ξ_m as:

$$z_m = \max(\xi_m, 0) \tag{6.6}$$

It can be shown that the explicit distribution of z_m is given by:

$$p(z_m|H_0) = \frac{u(z_m)}{\sigma_m} G\left(\frac{z_m - \mu_m}{\sigma_m}\right) + \delta(z_m)\,\Phi\left(-\frac{\mu_m}{\sigma_m}\right) \tag{6.7}$$

where $u(\cdot)$ is the unit step function, $\delta(\cdot)$ is the Dirac impulse function, and $G(\cdot)$ and $\Phi(\cdot)$ are the probability density and cumulative distribution functions of a standard normal variable, respectively. It can be shown that the mean and variance of this distribution, which are different from the parameters μ_m and σ_m^2, can be expressed as:

$$\begin{aligned} m_m &= \mu_m \Phi(\mu_m/\sigma_m) + \sigma_m G(\mu_m/\sigma_m) \\ s_m^2 &= \sigma_m^2 \Phi(\mu_m/\sigma_m) - m_m(m_m - \mu_m) \end{aligned} \tag{6.8}$$

Hence, the parameters μ_m and σ_m can be obtained from the observed values of m_m and s_m^2 by using a numerical method. It can be shown that this procedure yields the maximum likelihood estimates of the parameters. The parameters derived from the above simulations are shown in Table 6.1.

To obtain the distribution in presence of a target, a number of simulated targets of fixed amplitude were added to each of the images generated above. Morphological filter was applied to these images, and a histogram of pixel values only at the target positions was obtained. However, since the number of targets is not as large as the total number of pixels in the image, the histogram is less reliable than in the case of absence of targets. These experiments were repeated for various signal amplitudes and the sample mean and variance of the outputs were computed. The sample means and variances were taken as the estimates of the means and variances of the distributions. For the low-stop

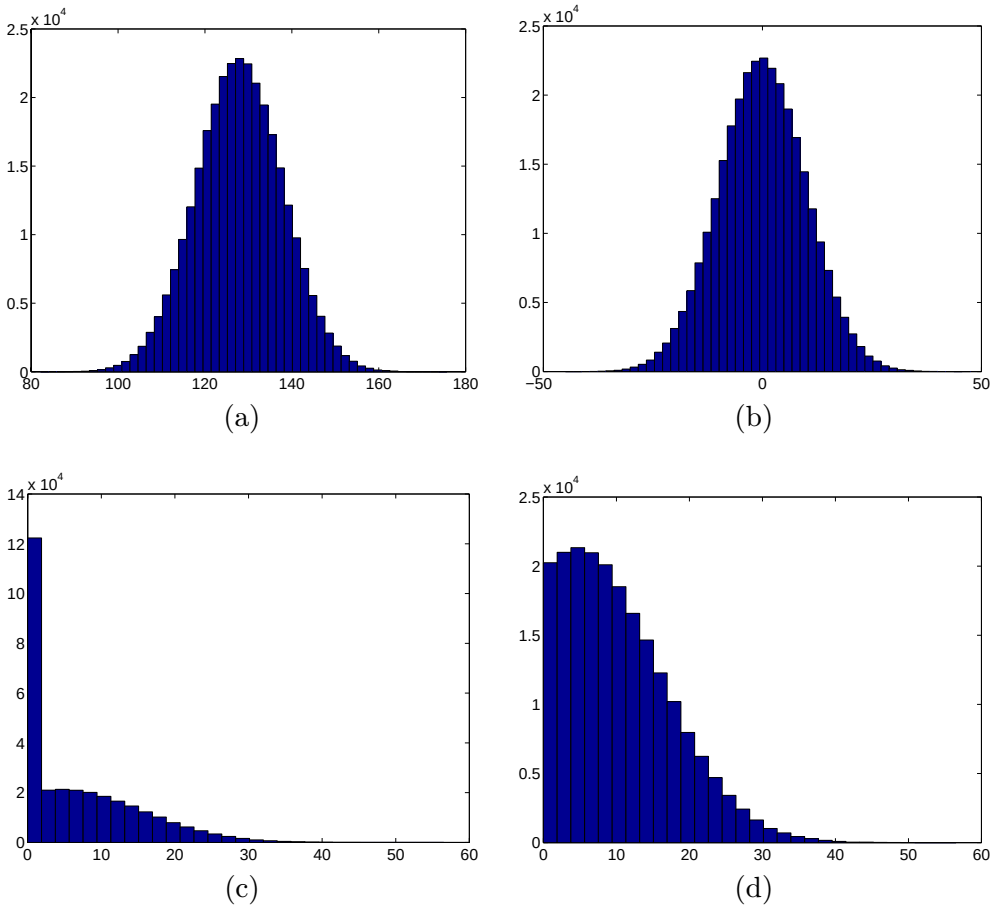


Fig. 6.1. Statistics of low-stop and morphological filters: Histograms of: (a) Input image with Gaussian noise. (b) Output of low-stop filter. (c) Output of morphological filter. (d) Output of morphological filter after removing impulse at zero value.

Table 6.1. Statistical parameters of low-stop and morphological filters derived from simulations

Parameter	Value
η	10.0
$m_l = \mu_l$	-6.6e-08 $\simeq$ 0.0
$s_l = \sigma_l$	9.9815 $\simeq$ 10.0
m_m	7.0539
s_m	7.8352
μ_m	4.6293
σ_m	10.8423

filter, the parameters μ_l and σ_l coincide with the distribution mean and variance m_l and s_l^2, respectively, and are approximately equal to the signal amplitude θ and the input noise standard deviation η, respectively, corresponding to $g_l \simeq 1$ and $f_l \simeq 1$. For the morphological filter, the actual parameters μ_m and σ_m of the underlying normal distribution were calculated from the m_l and s_l^2 using the simultaneous equations (6.8). It was observed that the parameter σ_m is approximately equal to the noise intensity η, and does not change much with the signal amplitude θ. However, the parameter μ_m increases non-linearly with θ. It has a positive value at $\theta = 0$ – i.e., noise-only condition – and increases with a lower rate than the corresponding low-stop filter parameter μ_l. Figure 6.2 shows the plots of the parameters μ_l and μ_m against the signal amplitude θ.

The output of the morphological filter is invariant to the constant background level β. Furthermore, it also suppresses the clutter. Hence, the effective 'noise' intensity for the morphological filter would be different from that for the low-stop filter in case of cluttered scenario, and is denoted by η_m. However, in the case of the above simulations it is the same as the original noise intensity η. If η_m as well as the signal amplitude θ are scaled by a constant factor, σ_m and μ_m will get scaled by the same factor. Hence, outputs of the morphological filter for any general η_m can be written as:

$$\sigma_m = \eta_m \, f_m \, , \; \mu_m = \eta_m \, g_m \left(\frac{\theta}{\eta_m} \right) = \frac{\sigma_m}{f_m} \, g_m \left(\frac{f_m \theta}{\sigma_m} \right) \tag{6.9}$$

where f_m is the gain in standard deviation ($f_m \simeq 1$), neglecting the dependency on the target SNR, and $g_m(\cdot)$ is the gain in mean, depending on the target SNR θ/η_m. The

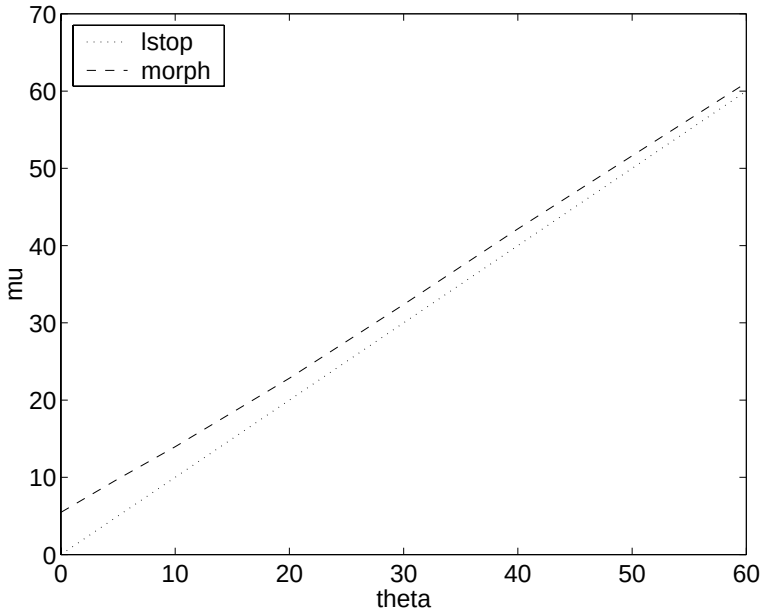


Fig. 6.2. Plot of parameters μ_l and μ_m against signal amplitude θ for $\eta = 10$.

function f_m can be obtained from the experimentally determined values of μ_m and σ_m for $\eta = 10$, plotted in Figure 6.2.

It was also observed that there is a correlation between the outputs of the low-stop and the morphological filters. Hence, the joint distribution of the two outputs is modeled as a normal distribution, truncated for the morphological filter. Assuming a hypothetical random variable $\xi = (\xi_l, \xi_m)^t$ which is normally distributed, the actual output vector z can be expressed as:

$$z = (z_l, z_m)^t = (\xi_l, \max(\xi_m, 0))^t \tag{6.10}$$

The parameters of distribution of z are:

$$\mu = \begin{bmatrix} \mu_l \\ \mu_m \end{bmatrix}, \Sigma = \begin{bmatrix} \sigma_l^2 & \rho\sigma_l\sigma_m \\ \rho\sigma_l\sigma_m & \sigma_m^2 \end{bmatrix} \tag{6.11}$$

where ρ is the correlation coefficient, and σ_l^2 and σ_m^2 are the individual variances of z_l and z_m, respectively. The distribution mean and covariance matrix are given by:

$$m = \begin{bmatrix} m_l \\ m_m \end{bmatrix}, S = \begin{bmatrix} s_l^2 & \rho' s_l s_m \\ \rho' s_l s_m & s_m^2 \end{bmatrix} \tag{6.12}$$

Note that due to the linearity of the low-stop filter, we have $m_l = \mu_l, s_l = \sigma_l$.

However, using these relations, it is analytically difficult to calculate the actual correlation coefficient parameter ρ from the observed correlation coefficient ρ'. Furthermore, such a computation would have to be repeated for every pixel, which is highly inefficient. Hence, the value of $\rho = \rho'$ is currently being used.

Using the above models of low-stop and morphological filter outputs, the distribution of z for $z_m > 0$ is given by:

$$\begin{aligned} p(z|H_0) &= |2\pi\Sigma|^{-1/2} \exp\left[(z-\mu_0)^t \Sigma^{-1} (z-\mu_0)/2\right] \\ p(z|H_1) &= |2\pi\Sigma|^{-1/2} \exp\left[(z-\mu_\theta)^t \Sigma^{-1} (z-\mu_\theta)/2\right] \end{aligned} \tag{6.13}$$

For $z_m < 0$, $p(z|H_i) = 0$ Also, there is an impulse function at $z_m = 0$, so that the integral of p becomes unity.

6.3 Bayesian fusion of multiple filters

The combined likelihood ratio of the two filters is given by:

$$L_{H,C}(z) = \frac{p(z|H_1, C)}{p(z|H_0, C)} \simeq \frac{N(\mu_\theta, \Sigma_C)}{N(\mu_0, \Sigma_C)} \tag{6.14}$$

where μ_θ and μ_0 are 2-D vectors denoting the mean outputs of the algorithms in presence and absence of target. The covariance matrix Σ_C, which depends on the clutter level, can be estimated using the image, and will be denoted by Σ for brevity. The same covariance is used for the presence and absence of the target, since it is experimentally observed that there is not much difference between the respective covariances.

Using equation (6.13), the log likelihood ratio (LLR) is given by:

$$\begin{aligned} l(z) &= \log L_{H,C}(z) = -\frac{1}{2}(z-\mu_\theta)^t\Sigma^{-1}(z-\mu_\theta) + \frac{1}{2}(z-\mu_0)^t\Sigma^{-1}(z-\mu_0) \\ &= (\mu_\theta-\mu_0)^t\Sigma^{-1}(z-\mu_0) - \frac{1}{2}(\mu_\theta-\mu_0)^t\Sigma^{-1}(\mu_\theta-\mu_0) \end{aligned} \tag{6.15}$$

The parameters of the LLR in absence of target – i.e., $E[z|H_0] = \mu_0$, $V[z|H_0] = \Sigma$ – can be computed as:

$$\begin{aligned} E[l(z)|H_0] &= (\mu_\theta-\mu_0)^t\Sigma^{-1}E[z-\mu_0|H_0] - \frac{1}{2}(\mu_\theta-\mu_0)^t\Sigma^{-1}(\mu_\theta-\mu_0) = -\frac{1}{2}d^2 \\ V[l(z)|H_0] &= (\mu_\theta-\mu_0)^t\Sigma^{-1}V[z-\mu_0|H_0]\Sigma^{-1}(\mu_\theta-\mu_0) = d^2 \end{aligned} \tag{6.16}$$

where d known as the deflection coefficient [37] is the generalization of the signal to noise ratio for multiple dimensions.

$$d = \sqrt{(\mu_\theta-\mu_0)^t\Sigma^{-1}(\mu_\theta-\mu_0)} \tag{6.17}$$

When the target of any strength is present, the variance parameter still remains the same but the mean parameter changes. For the target strength such that $E[z|H_1] = \mu_\theta$, the LLR parameters are given by:

$$\begin{aligned} E[l(z)|H_1] &= (\mu_\theta-\mu_0)^t\Sigma^{-1}E[z-\mu_0|H_1] - \frac{1}{2}(\mu_\theta-\mu_0)^t\Sigma^{-1}(\mu_\theta-\mu_0) = \frac{1}{2}d^2 \\ V[l(z)|H_1] &= (\mu_\theta-\mu_0)^t\Sigma^{-1}V[z-\mu_\theta|H_1]\Sigma^{-1}(\mu_\theta-\mu_0) = d^2 \end{aligned} \tag{6.18}$$

It is seen that the mean and variance of the LLR are dependent on the mean and variance parameters of the filter outputs. Due to this, the probability of false alarm

and mis-detection also depends on these parameters. Accordingly, two approaches of obtaining a detector are shown below.

6.3.1 Constant False Alarm Rate (CFAR) detector

To get a constant false alarm rate irrespective of the local variance, the LLR is normalized so that it would have a zero mean and unit variance in absence of the target. The resulting function is given by:

$$D(z) = \frac{l(z) - E[l(z)|H_0]}{\sqrt{V[l(z)|H_0]}} = \frac{(\mu_\theta - \mu_0)^t \Sigma^{-1}(z - \mu_0)}{\sqrt{(\mu_\theta - \mu_0)^t \Sigma^{-1}(\mu_\theta - \mu_0)}} \tag{6.19}$$

This is a matched filter, which matches the 2-D outputs from low-stop and morphological filters, to the expected outputs of these filters. Since $D(z|H_0) \sim N(0,1)$, if a threshold τ is applied, the false alarm rate is given by:

$$P_{FA} = 1 - \Phi\left(\frac{\tau - E[D(z)|H_0]}{V[D(z)|H_0]}\right) = 1 - \Phi(\tau) \tag{6.20}$$

where $\Phi(\cdot)$ denotes the cumulative distribution of a standard normal variable. Note that this is now independent of any parameters. In presence of a target so that $E[z|H_1] = \mu_\theta$, it can be easily seen that $D(z) \sim N(d,1)$. Hence, the mis-detection rate is given by:

$$P_{MD} = \Phi\left(\frac{\tau - E[D(z)|H_1]}{V[D(z)|H_1]}\right) = \Phi(\tau - d) \tag{6.21}$$

The CFAR approach attempts to maintain a constant false alarm rate all over the image, irrespective of the local variance. Hence, it would be useful if a constant false alarm rate is required in all parts of the image, for example, if the parts are processed separately on parallel processors. To check the conditions under which this filter is optimal, the log likelihood ratio $l(z)$ is written in terms of the discriminant function $D(z)$ as:

$$l(z) = d\,D(z) - d^2/2 \tag{6.22}$$

It can be seen that, $l(z)$ and $D(z)$ are monotonic to each other when the deflection coefficient d, given by equation (6.17) remains constant. Under such conditions, thresholding $D(z)$ is equivalent to thresholding $l(z)$, the latter being the Bayesian optimum. The deflection coefficient is dependent on the covariance of the noise, as well as the target strength, and is the generalization of SNR for multiple dimensions. Thus, if the variance

parameters of the individual filter outputs, as well the target amplitudes, are constant across the image, this approach is optimal in terms of the false alarms and mis-detection rates. However, in practice, the parameters (especially the low-stop filter output variance) do depend on the clutter level. In such a case, if the target amplitude is constant throughout the image, the CFAR approach is not optimal. However, if the criterion for good detection is to detect targets having a particular SNR – i.e., stronger targets in cluttered regions but weaker targets in uncluttered regions – the CFAR approach can be considered optimal.

It can be seen that $D(z)$ is dependent on the target amplitude θ through $\mu_\theta - \mu_0$, as well as d. If $\mu_\theta - \mu_0$ is a linear function of the target amplitude θ, it would cancel out in equation (6.19) and $D(z)$ would become independent of the signal amplitude θ. However, if $\mu_\theta - \mu_0$ is non-linear, the filter would be optimal only under specific conditions.

The false alarm rate is determined by the threshold τ, whereas the mis-detection rate is also determined by the deflection coefficient d. Consider optimizing the matched filter for a particular d, in an environment with clutter covariance Σ. If θ is the signal amplitude, equations (6.5) and (6.9) yield:

$$\begin{aligned} \mu_\theta - \mu_0 &= \begin{bmatrix} \mu_{l\theta} - \mu_{l0} \\ \mu_{m\theta} - \mu_{m0} \end{bmatrix} = \begin{bmatrix} g_l\theta - 0 \\ \left(g_m(\frac{\theta}{\eta_m}) - g_m(0) \right) \eta_m \end{bmatrix} \\ &= \begin{bmatrix} g_l\theta \\ \frac{\sigma_m}{f_m} \left(g_m(\frac{f_m\theta}{\sigma_m}) - g_m(0) \right) \end{bmatrix} \end{aligned} \tag{6.23}$$

Using this expression, the following equation should be numerically solved for θ by evaluating $\mu_\theta - \mu_0$ using equation (6.23) with the particular d.

$$(\mu_\theta - \mu_0)^t \Sigma^{-1} (\mu_\theta - \mu_0) = d^2 \tag{6.24}$$

However, if the covariance matrix Σ varies throughout the image, this procedure would have to be carried out for all pixels, which would be highly inefficient. Furthermore, the procedure optimizes only for a particular value of d.

Alternatively, if one assumes that d and θ are small, one can optimize the fusion using a Locally Most-Powerful (LMP) test [37, 49]. For small value of θ, we have:

$$\mu_\theta - \mu_0 \simeq \left(\frac{\partial \mu}{\partial \theta} \right)_{\theta=0} \cdot \theta = \begin{bmatrix} g_l \\ g'_m(0) \end{bmatrix} \theta = s\,\theta \tag{6.25}$$

where s is 2-D vector independent of θ. The expression is now linear in θ, and the discriminant function $D(z)$ becomes independent of θ.

$$D(z) = \frac{s^t \Sigma^{-1}(z - \mu_0)}{\sqrt{s^t \Sigma^{-1} s}} \tag{6.26}$$

with

$$s = \left[\begin{array}{cc} g_l & g'_m(0) \end{array} \right]^t \tag{6.27}$$

6.3.2 Direct thresholding of Log Likelihood Ratio (LLR)

As shown in the previous section, if the amplitude of the signal to be detected is fixed irrespective of the local variance, the overall mis-detection rate for a given overall false alarm rate is not minimized by the CFAR approach. In fact, there cannot be a single optimal detector for all amplitudes. Hence, the fusion should be optimized for a particular amplitude. A criterion for choosing this amplitude is described below.

Suppose that some particular minimum rates of false alarms as well as mis-detections are required for the algorithm. The amplitude corresponding to the minimum possible variance – i.e., the variance of the camera noise without clutter – can be used to tune the fusion. If the actual amplitude is smaller than this amplitude, even an optimal detector tailored to that amplitude will not give the required false alarm and mis-detection rates. On the other hand, since the performance of the detector increases monotonically with the amplitude, a larger amplitude yields a better performance, though it may not be optimal.

Suppose the LLR threshold is τ. Using the mean and the variance of the LLR in absence and presence of the target, given by equations (6.16) and (6.18), the false alarm and mis-detection rates can be computed as:

$$P_{FA} = 1 - \Phi\left(\frac{\tau + d^2/2}{d}\right) \, , \; P_{MD} = \Phi\left(\frac{\tau - d^2/2}{d}\right) = 1 - \Phi\left(\frac{d^2/2 - \tau}{d}\right) \tag{6.28}$$

If one denotes:

$$\phi_0 = \Phi^{-1}(1 - P_{FA}) = \frac{\tau + d^2/2}{d} \, , \; \phi_1 = \Phi^{-1}(1 - P_{MD}) = \frac{d^2/2 - \tau}{d} \tag{6.29}$$

then τ can be eliminated to obtain:

$$\phi_0 - \phi_1 = d = \sqrt{(\mu_\theta - \mu_0)^t \Sigma^{-1} (\mu_\theta - \mu_0)} \tag{6.30}$$

The target amplitude can be chosen such that μ_θ corresponding to it satisfies this equation, using Σ^{-1} under noise only conditions.

6.4 Application on images

To apply this procedure on images, the statistical parameters are computed in an annular 31×31 window around each pixel, where an 11×11 window immediately around the pixel is excluded to reduce the biasing of parameters when the target is present at the pixel. There is a trade-off between using larger sized window giving more reliable estimates, and smaller sized window giving better localization in case of space varying clutter intensity. The window size used here was arbitrary. However, use of different window sizes can be explored to find the optimum window size.

Efficient methods are used to estimate the distribution mean m and the covariance S at each pixel of the low-stop and morphological output images. From these, the estimates of μ and Σ are calculated and stored as images. However, in some experiments, fixed values of μ_m and σ_m^2 were used for the morphological filter, since the estimates are less reliable, but do not change much over the image (unlike low-stop filter, where these parameters heavily depend on the clutter). The template signal for the matched filter is calculated using equation (6.27), and the matched filter is applied separately to each pixel.

6.5 Results

The algorithm fusion approach was evaluated using the performance characterization approach of Chapter 3. Background images obtained from digital and analog cameras shown in Figures 6.3 (a) and (b), respectively, were used for false alarm analysis. For mis-detection analysis, a number of targets of size 2×2 were added to these images. Low-stop and morphological filters were applied to these images. The outputs of these filters were fused using the two approaches described above. The local variance of the low-stop filter output, which is a measure of clutter, is shown in Figures 6.3 (c) and (d). The histogram of the local variance is shown in Figures 6.3 (e) and (f). It is seen that the analog camera image has a much higher clutter level than the digital camera image.

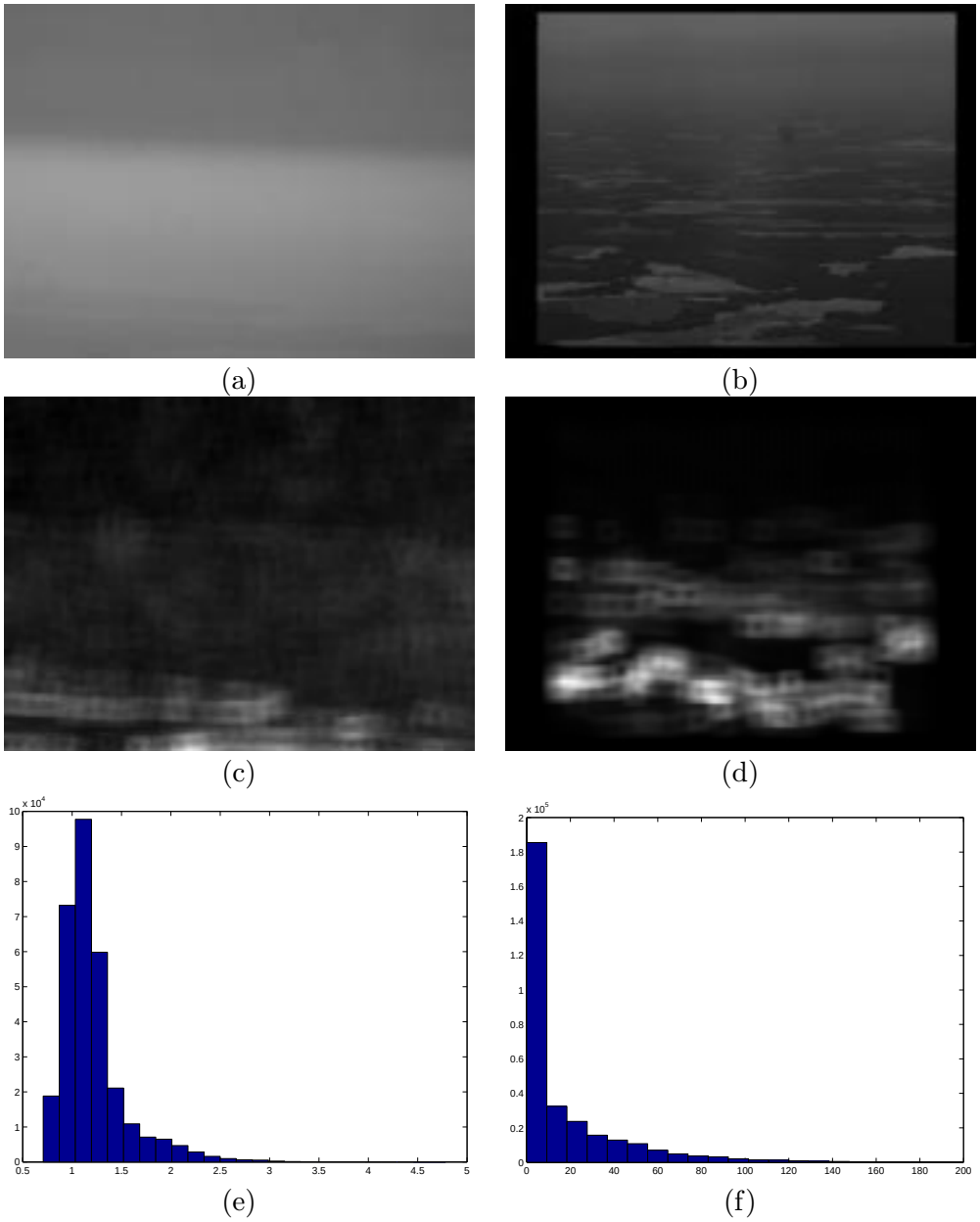


Fig. 6.3. Images from (a) digital (b) analog camera with partly cluttered background. Image of the local variance of low-stop filter output, which is the measure of clutter for images from (c) digital (d) analog camera. Histogram of the local variance of low-stop filter output for images from (e) digital (f) analog camera.

For the Constant False Alarm Rate (CFAR) fusion, the Locally Most Powerful (LMP) test was used. This gave the matched filter template as:

$$s = (g_l, g'_m(0))^t = (1.0, 0.8623)^t$$

gives a slightly lower weight to the morphological filter when the level of noise is same for both the filter outputs. The plots of the mis-detections against the false alarms for the digital camera images are shown in Figure 6.4 (a) and (b), These use the assumption that the outputs of the low-stop and morphological filters are correlated. Algorithm fusion was also performed assuming independence between filters – i.e., $\rho = 0$. The independence assumption gave a slightly better performance for the fused filter as shown in Figure 6.4 (c) and (d), possibly because the correlation between filters may not have been adequately modeled. Similar plots using analog camera images are shown in Figure 6.5.

In both the cases, it is seen that the fused output does not give optimal performance for all the rates of false alarms. However, it can be observed that the fused output does give larger weight to the filter which has a better performance in the particular case. For example, in the case of digital camera images having relatively low clutter, (Figure 6.4), the better performing low-stop filter is given higher a weight. On the other hand, for analog camera images (Figure 6.5) with severe background clutter, the morphological filter which performs better is given a higher weight. Since the individual filter which would actually perform better in a particular case would not be known a-priori, the fusion at least serves the purpose of selecting the better filter.

To explore the reasons for the non-optimality of the CFAR approach, the method of thresholding the log likelihood ratio (LLR) was first used in place of the CFAR fusion. The results of thresholding likelihood ratio are shown in Figure 6.6. The outputs of the individual detectors, the likelihood ratio detector using each filter, and the fused likelihood ratio detector are shown for amplitudes of 6.0 and 8.0. The amplitude used for computing the likelihood ratio was of 6.0, which gave the signal template as:

$$\mu_\theta - \mu_0 = (6.0, 5.5603)^t = 6.0(1.0, 0.9267)^t$$

which is only slightly different from the LMP template (scaled), due to the non-linearity of the morphological filter.

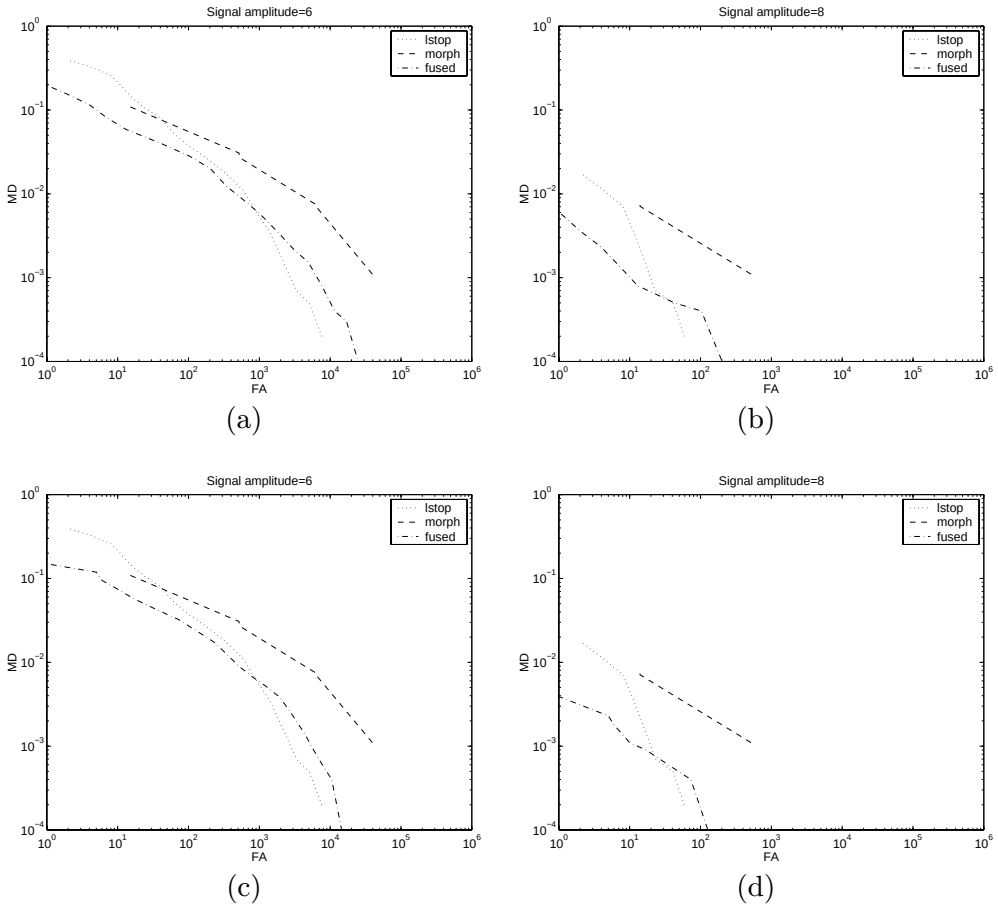


Fig. 6.4. Operating curves for digital camera image using CFAR fusion: Assuming correlation between filters with target amplitudes of: (a) 6.0 (b) 8.0 Assuming independence between filters with target amplitudes of: (a) 6.0 (b) 8.0

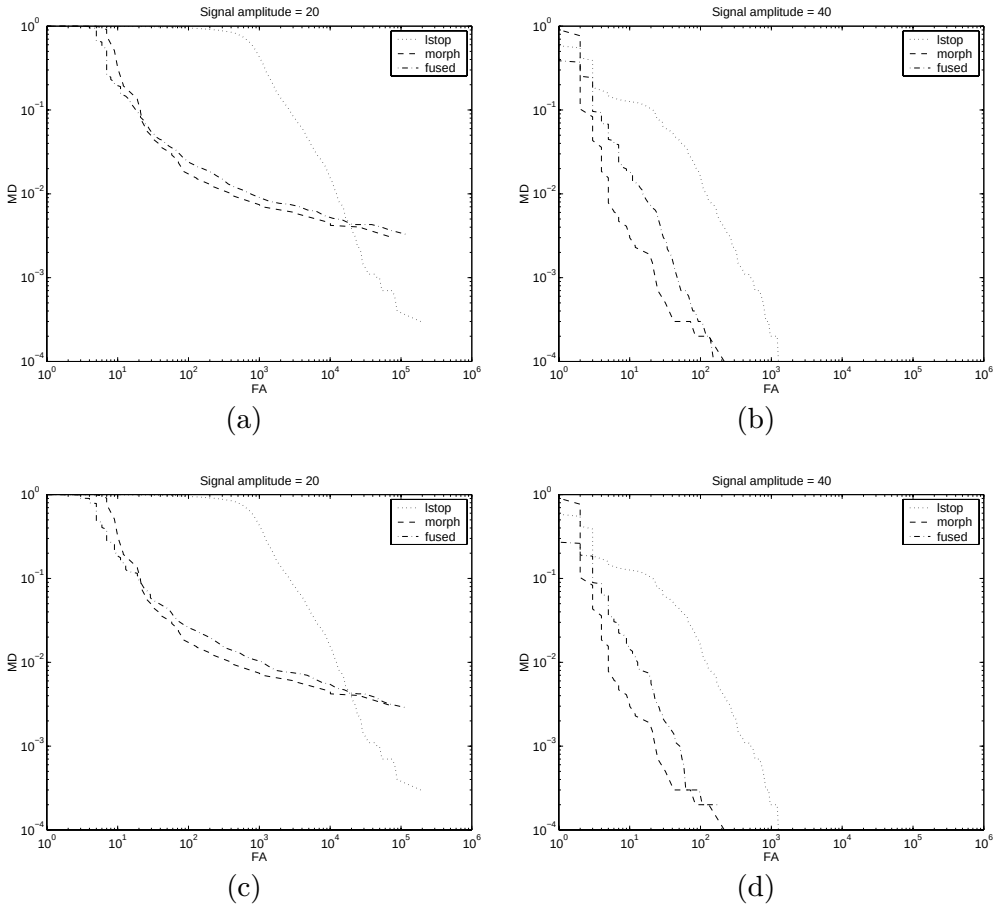


Fig. 6.5. Operating curves for analog camera image using CFAR fusion: Assuming correlation between filters with target amplitudes of: (a) 20.0 (b) 40.0 Assuming independence between filters with target amplitudes of: (c) 20.0 (d) 40.0

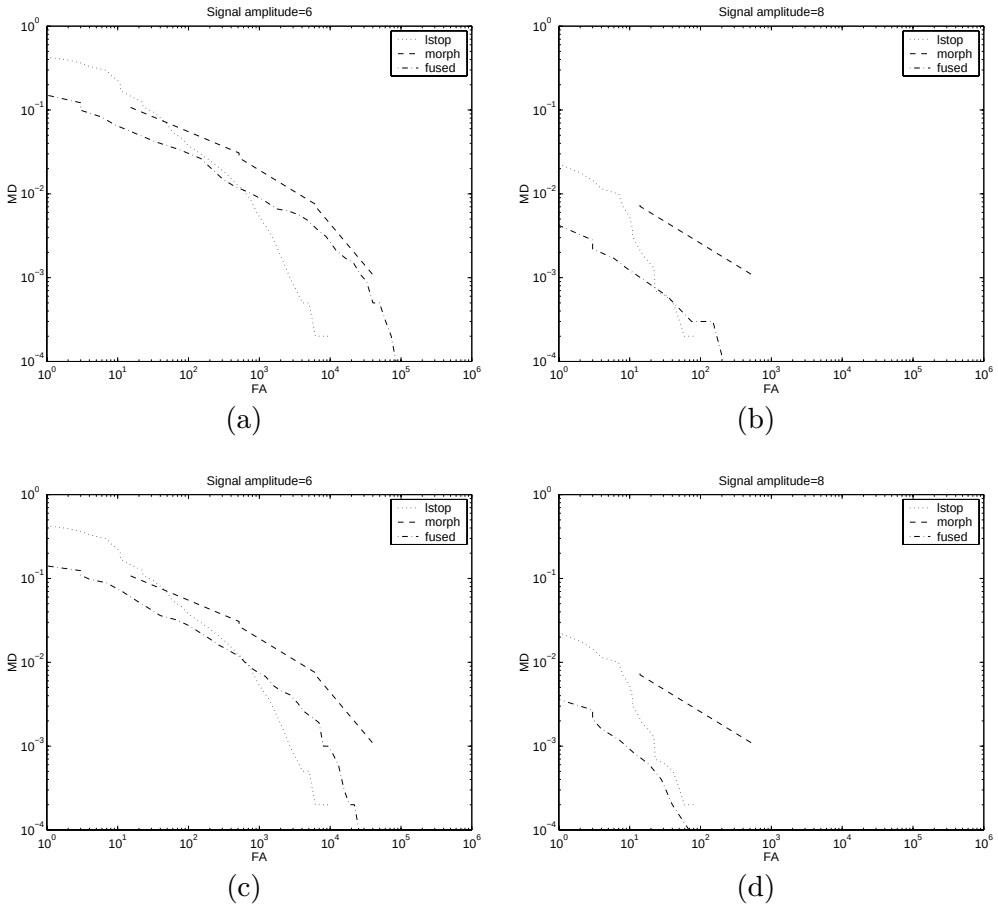


Fig. 6.6. Operating curves for digital camera image using LLR thresholding: Using correlation between filters with target amplitudes of: (a) 6.0 (b) 8.0 Assuming independence between filters with target amplitudes of: (c) 6.0 (d) 8.0

However, it was seen that for the matched signal strength of 6.0, it still did not give desirable performance. Hence, another reason for this non-optimality was explored. It was observed that the variance parameter σ_m^2 of the morphological filter output was underestimated from the images. This unreliability of was because the estimation was performed using small windows around every point in the image. Furthermore, there was quantization error, since the noise in the images was of the same order as the gray level resolution of the real images. However, since the morphological filter is comparatively insensitive to clutter, the value of σ_m^2 remains approximately same throughout the image. Hence, the entire background image from the digital camera (without adding targets) was used to pre-compute the parameter value as $\sigma_m^2 = 1.5$. The low-stop filter parameter σ_l^2 was estimated as before, since its value *does* depend on the local clutter level. The correlation coefficient was assumed to be zero. The results obtained using these parameters are much better, and shown in Figure 6.7.

Hence, it can be concluded that the performance of CFAR approach was poor due to the following reasons:

1. CFAR fusion is not optimal under the condition of constant target amplitude.
2. The morphological filter parameters are not reliably estimated from small sized windows.

However, as shown before, the CFAR approach is theoretically optimal, when the target amplitude is not constant, but is adjusted so that the deflection coefficient d given by equation (6.17) remains constant. To check the optimality of the CFAR approach for this condition, another set of experiments was performed. The statistical parameters of the low-stop filter were estimated at every pixel using the background image without the addition of targets. The morphological filter parameters were estimated for the entire image (instead of individual pixels). Using the parameters of the low-stop and morphological filters, the deflection coefficient d_1 for a unit amplitude of the signal was computed at every pixel, and stored as a separate image. False alarm rate was also estimated using this image as before. For estimating mis-detection rates, targets were added to the background image. The amplitude of the target at a particular pixel was given by d/d_1 where d_1 is the function of the pixel coordinates and d is constant. The mis-detection rate was then estimated from a number of such images. The LMP template was used for fusing the outputs of the individual filters. The plots of the mis-detection rate against false alarm rate are shown in Figure 6.8. It can be seen that the fusion output is better or as good as the individual filter outputs, within experimental error.

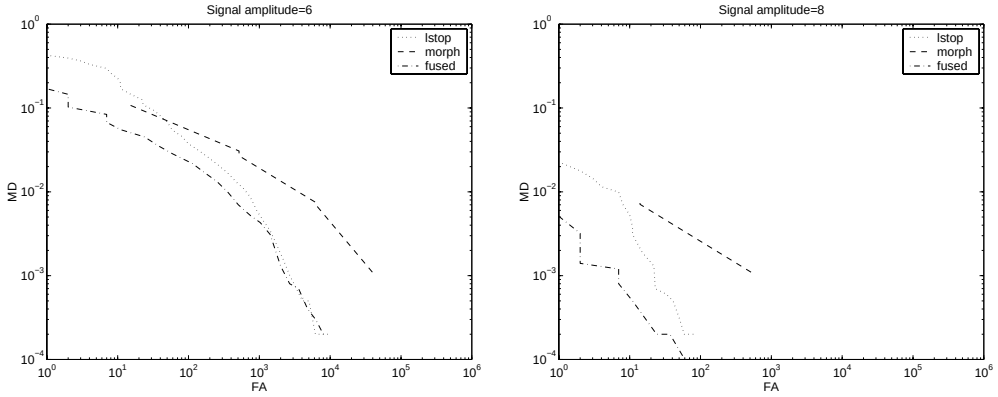


Fig. 6.7. Operating curves for digital camera image using LLR thresholding, and fixed value of morphological variance parameter $\sigma_m^2 = 1.5$, and assuming independence between the filters, for target amplitudes: (a) 6.0 (b) 8.0

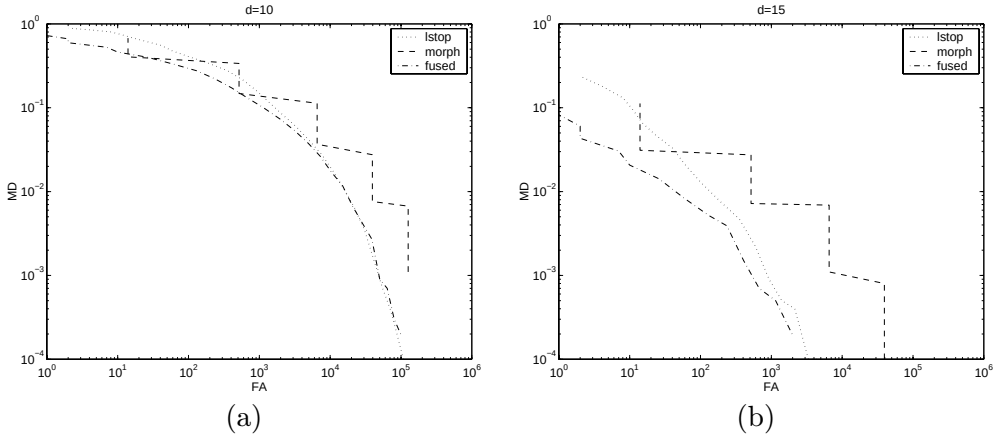


Fig. 6.8. Operating curves for digital camera image using CFAR approach with condition of optimality, and fixed value of morphological variance parameter $\sigma_m^2 = 1.5$, assuming independence between the filters, where the targets have amplitude such that the deflection coefficient d is constant equal to: (a) 10.0 (b) 15.0

Chapter 7

Detection of Translating Objects

In addition to the detection of objects on a collision course, it is useful to monitor the objects which are crossing the aircraft. For this purpose, a system was designed to specifically detect objects having a translational motion in the image. To distinguish translating objects from ground or cloud clutter, the following criteria were used:

1. The object should have sufficient signal strength.
2. The object should have an image velocity greater than a threshold.
3. The object should have a consistent motion – i.e., its velocity must not change abruptly.

The system to detect translating objects has been implemented on the pipelined image processing system, the DataCube MaxPCI described in Section 2.8 to obtain real time performance. The system was mounted on the Air Force Total In-Flight Simulator (TIFS/NC1314) aircraft, and flight tests were conducted by NASA with another aircraft flying in front of it. The detection and tracking of the target aircraft were demonstrated during the flight test.

This system is divided into two stages, an image processing stage and a tracking stage. The first stage consists of image processing steps which remove most of the clutter, and isolate potential features which could be translating objects. This stage involves repetitive image operations such as convolution, pointwise operations, histograms, etc. which are suitable for a pipelined architecture, and can be performed in integer format. Hence, these steps are implemented on the DataCube machine. The output of this stage is a list of image features which are likely to contain the target objects, including their positions and the signal strengths. However, the list may also contain features corresponding to background clutter, which are not separated by the simple image processing steps of the first stage. The second stage tracks these features to distinguish the genuine translating objects from background clutter using the criteria mentioned above. Since the first stage has reduced the volume of data to be operated on, more complicated target tracking algorithms can be implemented even on the host PC associated with the

DataCube. The threshold used in the first stage is adjusted dynamically to give a nearly constant number of features for the second stage so that they can be processed in real time using the slower host. This matching of the output rate of one stage to the input rate of the next stage is known as the rate constraint criterion [10].

7.1 Image processing stage

This stage performs the basic image processing steps to suppress clutter and extract features which could potentially be translating targets.

1. Resolution Reduction: The resolution of the image is reduced so that the system is capable of operation in real time. The image is convolved with the following low-pass filter mask and then down-sampled by two in both horizontal and vertical directions.

$$M_0 = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 24 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 24 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

   Low-pass filtering suppresses high frequencies, which would otherwise have been aliased to low frequencies by the down-sampler. Although the image resolution is reduced, the signal to noise ratio is actually enhanced. This is because the target size is usually greater than 2 pixels, leading to spatial integration of the target contrast.

2. Low-stop filtering: A low-stop filter is applied to the reduced image to suppress background clutter. The filter is implemented by convolving the image with the following masks, one after the other:

$$M_1 = \frac{1}{64}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 2 & 8 & 12 & 8 & 2 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}, \; M_2 = -\frac{1}{128}\begin{bmatrix} 0 & 2 & 3 & 2 & 0 \\ 2 & 8 & 12 & 8 & 2 \\ 3 & 12 & -108 & 12 & 3 \\ 2 & 8 & 12 & 8 & 2 \\ 0 & 2 & 3 & 2 & 0 \end{bmatrix}$$

   The mask M_1 is a smoothing mask, which performs spatial integration for large targets. A rectangular mask is used since the targets are expected to have a greater

width than height. Application of the mask M_2 is equivalent to subtracting a smoothed image from the input image. The overall result of the two convolutions is the subtraction of a low-pass filter output with a larger mask from a low-pass filter output with a smaller mask. Hence, this step suppresses uniform background intensity and weak clutter corresponding to low frequencies, and also performs spatial integration for larger objects.

3. Image differencing: Image differencing is performed on the low-stop filtered images by subtracting consecutive frames. This is equivalent to a low-stop filter in temporal direction. Since the object is assumed to be translating, image differencing suppresses stationary objects corresponding to background clutter. It should be noted that steps 1 to 3 are theoretically interchangeable, since they are all linear filters. However, since these operations are performed with integer arithmetic of limited precision, the particular order of the steps is used to reduce the truncation error.

4. Non-maximal suppression: Directly using the output of the previous step would give rise to a large number of features for an extended target. Non-maximal suppression is performed to get a single feature (or sometimes a small number of features) for the entire target. Pixels can have both positive or negative values corresponding to bright and dark targets, respectively. Hence, an absolute value image is first formed, and every pixel which is not a local maximum in its 3×3 neighborhood is marked. The marked pixels are set to zero in the *original image* – i.e., the image before taking the absolute values.

5. Histogram formation: To extract candidate features, the output from the above steps should be thresholded. Furthermore, the threshold should be chosen so that the number of features neither overloads the tracking stage, nor keeps it unnecessarily idle. Hence, the threshold is selected so that the number of pixels exceeding the threshold is less than or equal to a fixed rate which matches the operation speed of the tracking stage. For this purpose, a histogram of the image is constructed. The threshold then is determined as the smallest pixel value for which the number of elements in the histogram bins above this value does not exceed the fixed rate. Applying this value as the threshold would then ensure that the number of features remains bounded.

6. Thresholding and feature output: Pixels in the image with the output value greater than the threshold are separated as features, and their positions as well as the amplitudes are transmitted to the tracking stage.

7.2 Tracking stage

This stage maintains a list of tracks containing the frame number, unique ID, position, velocity, and amplitude. The list is empty in the beginning. The following steps are repeated for every frame for which the list of features is received from the image processing stage:

1. Track update: For each track in the list of tracks, the list of features is scanned to obtain features in a neighborhood window around the track position. If one or more such features are found, the one with the largest amplitude is selected as the continuation of the track. Using the coordinates (z_1, z_2) of this feature, as well as the current track position (x_1, x_2) and velocity (u_1, u_2), the expected position and velocity for the next frame is estimated using a Kalman filter. The filter is applied separately for horizontal $(i = 1)$ and vertical $(i = 2)$ directions. For each direction, the state vector is given by $X_i = \begin{bmatrix} x_i & u_i \end{bmatrix}^t$, and the observation is the feature coordinate z_i. The track life n of the track is the number of frames in which the target has been observed, with adjustments made in the frames where the target is not observed. The measurement update is given by:

$$\begin{aligned} x_i^+(n) &= x_i(n) + K_1(n)\,(z_i - x_i) \\ u_i^+(n) &= u_i(n) + K_2(n)\,(z_i - x_i) \end{aligned} \tag{7.1}$$

The state update is given by:

$$\begin{aligned} x_i(n+1) &= x_i^+(n) + u_i^+(n) \\ u_i(n+1) &= u_i^+(n) \end{aligned} \tag{7.2}$$

The Kalman filter matrix $K(n) = \begin{bmatrix} K_1(n) & K_2(n) \end{bmatrix}^t$ is pre-computed using the inverse covariance formulation of the Kalman filter. The computation is performed for a number of $n = 1 \ldots N$, where N is large enough so that $K(N)$ does not change significantly with N.

The track amplitude is updated using recursive averaging according to the following equation:

$$F(n+1) = f(n) + \alpha F(n) \tag{7.3}$$

where $F(n)$ and $F(n+1)$ are the track amplitudes for the current and next frames, $f(n)$ is the feature amplitude, and α is the forgetting factor. The track life n is incremented by one.

If no feature satisfying the above conditions is found in the neighborhood of the track, the position and velocity are extrapolated using only the state update. Theoretically, this would mean that the values of the Kalman filter matrix would have to be recomputed. To avoid such a computation, the value of the track life n is reduced by a factor to approximately simulate the effect of having 'lost track' of the feature. The feature amplitude is updated using $f(n) = 0$ in equation (7.3).

2. Formation of new tracks: After all the current tracks are updated, features in the feature list are used to check for new tracks. For every feature, the list of tracks is scanned to see if a track is already there in its neighborhood. If not, a track is created out of the feature with its track life $n = 1$. Its position (x_1, x_2) will be the same as feature position (z_1, z_2), whereas velocity (u_1, u_2) is initialized to zero. The actual velocity will be computed only in the next frame.

3. Pruning the list of tracks: If the number of tracks is too large, the stage can get overloaded and fail to operate in real time. To eliminate this possibility, if the number of tracks are greater than a particular number, the weakest tracks are deleted.

4. Merging similar tracks: It may happen that two or more tracks may be formed corresponding to the same object. Hence, tracks which are very close to each other and have nearly the same velocity are merged, retaining the one with the larger track amplitude.

5. Output: Tracks which satisfy the criteria of the object, including having an amplitude larger than a threshold, as well as a significant and consistent motion are outputted as potential objects.

7.3 Results

The real-time image capturing, recording, and processing system were demonstrated by the flight tests conducted by NASA. During the first set of flight tests, image sequences were captured and recorded successfully at the rate of 30 frames per second. The tracking algorithms were designed and fine-tuned using these image sequences. During the next set of flight tests, in addition to the real-time capturing and recording, the translating target tracking algorithm was executed concurrently at the rate of 15 frames per second. Several image sequences with the target aircraft crossing the host aircraft were obtained. It was observed that the system successfully detected and tracked the translating object during the flight tests. Figure 7.1 shows a trace of the tracking algorithm applied on an image sequence with the target aircraft translating from right to left at a distance of 3 nautical miles.

Table 7.1 summarizes the performance of the translating target tracking algorithm with different distances between the host and the target aircraft, during the first set of flight tests. The false alarm rate is measured as the ratio of the total number of false alarms throughout the sequence to the number of image frames in the sequence. The mis-detection rate is measured as the ratio of the number of frames in which the target was missed to the total number of frames. The false alarm rate depends on the amount and motion of clutter in the images, whereas the mis-detection rate depends on the target size and contrast, and therefore increases with the target distance in most cases. Since false alarms can be very annoying to the pilots, a low false alarm rate was more desirable than a low mis-detection rate. Hence, the parameters of the algorithms were selected to reduce the false alarm rate, and were same for all the scenarios. It is possible to get a better performance by adjusting parameters according to the characteristics (such as the clutter level) of each scenario.

The performance was relatively poor in the cases where the host aircraft rotated about its own axes, resulting in large image motion of background features. To improve the performance, the image motion due to aircraft rotation should be compensated using the aircraft navigation data. If this data is unavailable, the background motion should be modeled to separate independent object motion. For example, Irani and Anandan [28] separated the scene motion into planar and parallax components, and identified independently moving objects having a significant parallax. However, since the DataCube architecture is capable only of simple image processing operations, any such procedure would have to be performed on the host machine, using a feature based approach.

Fig. 7.1. Tracking algorithm applied on an image sequence with the target aircraft translating from right to left at a distance of 3 nautical miles. The target aircraft is located at the end of the track in this image.

Table 7.1. The performance of the translating target detection algorithm for a number of target distances. The false alarm rate is the ratio of the total number of false alarms throughout the sequence to the number of image frames in the sequence. The mis-detection rate is the ratio of the number of frames in which the target was missed to the total number of frames.

Distance (nmi)	Mis-detection rate	False alarm rate
1.5	0.061	0.000
1.8	0.113	0.000
2.0	0.394	0.000
2.4	0.059	0.000
3.0	0.056	0.000
4.7	0.335	0.183
5.0	0.803	0.147
5.4	0.643	0.000

Chapter 8

Runway Object Detection

In recent years, considerable effort has been put into evaluating the feasibility of computer vision algorithms for the tasks of navigation and runway obstacle detection using visible light images. This chapter briefly describes the work performed for detection of obstacles on the runway for the NASA Synthetic Vision System (SVS) project, citing appropriate references for details.

The detection of runway lines has been attempted in [27, 43] using properties specific to the runway (e.g., the use of parallel lines). Work described in [21] is based on a model of the runway where the camera image is compared with the expected runway image obtained by generating an image using the known position of the camera and the airport runway model database. On-board INS and GPS can provide reasonably good position estimation during flight, but they are not considered accurate enough to permit automatic landing [54]. Image-based features such as points, lines, etc., are used to determine the position of a sensor in [5, 39, 40].

In principle, the problem of obstacle detection with a moving camera can be completely solved if either the range map – i.e., the 3-D object distance corresponding to each pixel in the image – is supplied for all points in front of the camera, or the image acquired using the sensor is segmented and recognized into constituent parts – i.e., sky, runways, buildings, etc. Sridhar et al. [55] developed a recursive estimation procedure for estimating range to feature points in the image. Image regions of size 11×11 pixels with high variance are detected as features. The initial range of these features is computed by triangulation using motion. The feature positions in the subsequent frames are predicted, and the estimates of ranges to the feature points are refined using a Kalman filter which combines the observed position and velocity parameters with those predicted from previous frames, to obtain an optimal estimate of the parameters. The algorithm is tested and the results are reported for both indoor images and images acquired from the actual flight test. In this work, all obstacles are assumed to be stationary. The algorithm has also been modified to handle stereo images [52].

If the camera were stationary, moving objects could be detected by taking pixel-wise difference between consecutive image frames [24, 30, 31]. However, since the camera is also moving, there is image motion everywhere in the image plane. Hence, simple image differencing approaches are not useful for identifying independently moving objects. Instead, the motion in the entire image should be estimated and analyzed.

Under normal conditions, the apparent image motion, known as optical flow can be estimated by observing spatial and temporal changes in the image intensity in the neighborhood of a pixel. The optical flow can then be used to identify independently moving objects from the apparently moving background. However, if the motion of the camera or the 3-D structure (range map) of the scene are unknown, the problem becomes more difficult. Thompson and Pong [59] have shown by example that different motion and structure situations can result in an identical optical flow distribution. Hence, it is impossible to derive motion or structure information uniquely, given only the image and the optical flow field.

Tang et al. [58, 57] use a simple incremental weighted least squares method for estimating the position of stationary objects using known camera state parameters. This algorithm extends the epipolar plane image analysis of Bolles and Baker [12, 6] to general camera motion by assuming the camera motion to be piecewise linear. For detecting independently moving objects, an optical flow-based approach is used [58]. Image regions corresponding to the independently moving objects are segmented from the background by applying the constraint filtering originally proposed by Nelson [45] on the optical flow computed from the initial few frames of the sequence.

Gandhi et al. [23, 35] use warping to compensate the ego-motion of stationary objects on the runway in order to separate objects that are moving or at a height above the runway, as described in Section 8.1. Parameters of camera motion and the runway plane are assumed to be known in this work. However, this approach requires the knowledge of the motion and plane parameters. To eliminate the need for these parameters, Devadiga [20] used a planar motion model to estimate the image motion of the runway plane. Section 8.2 describes the work done by Devadiga [20], in which the knowledge of the motion and plane parameters is not required. Instead, a planar motion model is used to separate the obstacles from the runway, which is assumed to be piecewise planar.

8.1 Use of warping to compensate ego-motion

In the work by Gandhi et al. [23, 35], it is assumed that the camera motion is approximately known, and the runway is a planar surface with approximately known parameters. Similar to the approach used by Sull et al. [56], the residual flow is computed by compensating for the camera motion using image warping. Warping parameters are computed by using the known camera motion and plane parameters. Obstacles are detected by thresholding the residual optical flow computed using the warped images. The approach of image warping has also been used by Carlsson et al. [16] where warping parameters are computed by matching image windows in two frames.

The system block diagram is shown in Figure 8.1. The input to the system is a sequence of images captured from the camera on board, position and velocity (both linear and angular) of the aircraft obtained from sources, such as the GPS and INS, referred here as the Inertial Navigation Unit (INU), and knowledge about the parameters of the runway plane. Using these parameters, a transformation to map features from one frame to another is obtained. Since the image motion can be reliably estimated only in regions resembling corners, the optical flow is calculated only at corner-like features detected using the corner detector proposed by Smith and Brady [53]. The features are then warped using the transformation obtained from INU and runway plane parameters. The optical flow is obtained from the warped features in multiple frames of the image sequence. Optical flow is computed using Lucas and Kanade's algorithm [41] with Simoncelli's [51] modification to compute the covariance of the estimated optical flow. Due to warping, the ego-motion of the features on the runway is compensated as much as possible and residual flow is obtained. Since the ego-motion is quite large, compensating the ego-motion makes it much easier to detect independently (slower) moving obstacles as well as the obstacles with (relatively small) height. Once the optical flow is obtained, features are tracked from one frame to the other, and velocities of the features are smoothed using moving average filtering. The smoothed estimate is added to the warped features, so that in the subsequent frames, even this flow is compensated. The estimated residual velocity can then be thresholded in order to check which features are moving or are at a height above the runway plane. The details of this method are described in [23, 35].

The method described above was applied to a video image sequence provided by NASA Ames Research Center. The sequence was obtained using a camera mounted on-board an aircraft with a truck moving across the runway from right to left. A zoomed part

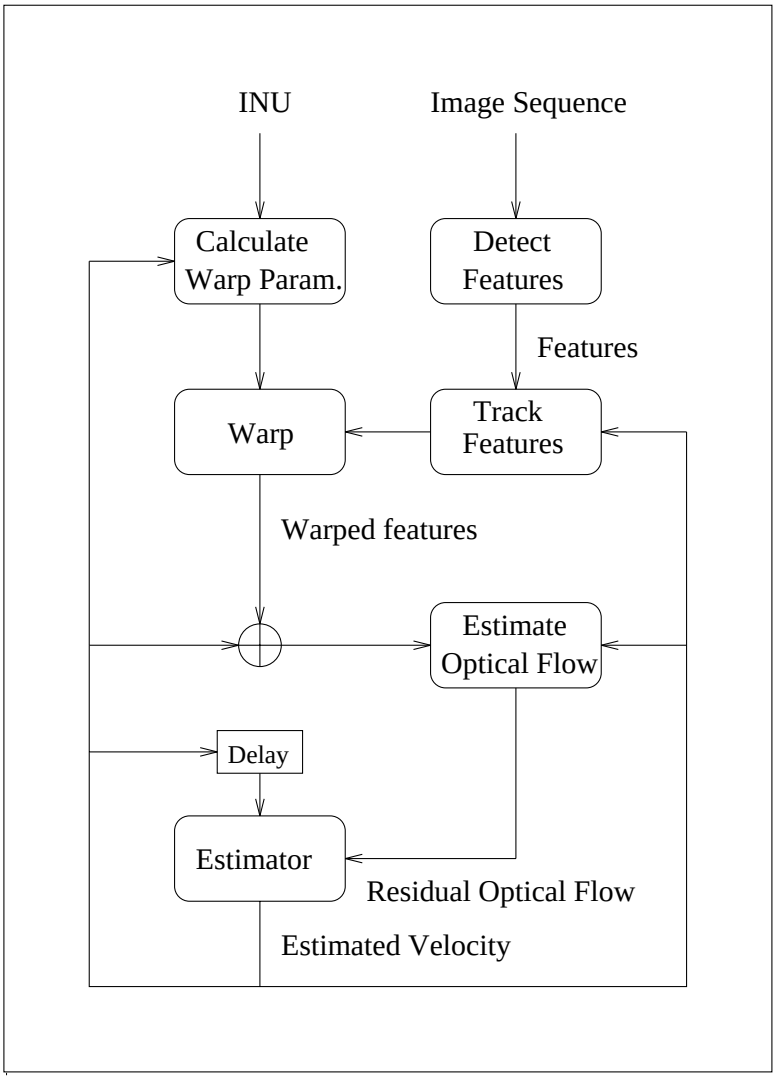


Fig. 8.1. Use of warping to compensate ego-motion: System block diagram.

of frame 50 from the image sequence is shown in Figure 8.2 (a). It is seen that the image contains numerous extraneous features, such as tire-marks in addition to the moving truck, making the task more difficult. The tracked features, and their image velocity estimates after motion compensation are shown in Figure 8.2 (b) and (c) respectively. The ellipses shown in Figure 8.2 (c) are proportional to the estimated covariances of the velocity estimates. Thresholding the velocity separates the moving truck from the extraneous features due to tire marks as shown in Figure 8.2 (d).

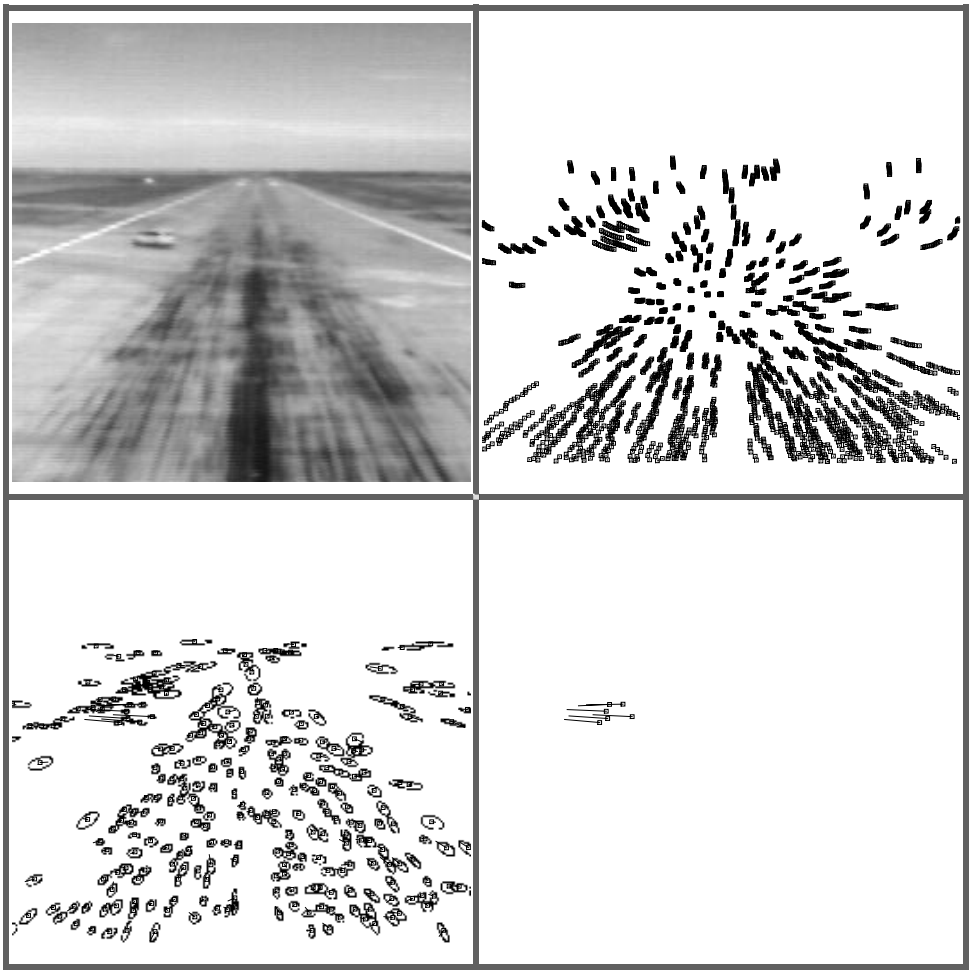
8.2 Use of planar motion model

In the work by Devadiga [20], the camera motion as well as the runway parameters are assumed to be unknown. The only assumption is that the background – i.e., the runway – is planar or is piecewise planar. The assumption is applicable even in cases where the background is not planar, as long as the ratio of distance to the scene point from the camera to the variation in the depth of scene points is large.

The block diagram of the system is shown in Figure 8.3. Similar to the approach used by Gandhi et al. [23, 35], the optical flow is computed at selected feature points detected using the corner detector proposed by Smith and Brady [53]. Optical flow is computed using Lucas and Kanade's algorithm [41] with Simoncelli's [51] modification to compute the covariance of the estimated optical flow. However, since compensation of ego-motion is not performed, the optical flow can be very large in some regions. Hence, a hierarchical computation method [9] using Gaussian pyramids [15, 50] was used to avoid errors due to temporal aliasing.

In the ideal situation where a single planar surface fits all the points in the scene, a planar motion model consisting of eight parameters can be fitted to the optical flow vectors using least squares method. However, obstacles having independent motion, or a significant height above the planar surface do not satisfy this model. They are expected to have a large error of fit, and are classified as outliers. However, if the total error for fitting a single planar surface to the entire image is large, it is hypothesized that the scene is piecewise planar. A recursive motion-based segmentation algorithm is then used to segment the image into regions corresponding to planar surfaces, as well as outliers which could correspond to independently moving objects. After removing the outliers, least squares is used to estimate the motion parameters of each of these surfaces. The details of optical flow computation method, as well as the planar motion model and the segmentation algorithm to segment single and multiple motion are described in [20].

(a) (b)



(c) (d)

Fig. 8.2. Detection and tracking of an obstacle on a runway: (a) A typical frame in an image sequence of a truck crossing a runway. (b) Tracked features in the sequence of images. (c) Estimate of residual feature velocities. (d) Detection of moving truck.

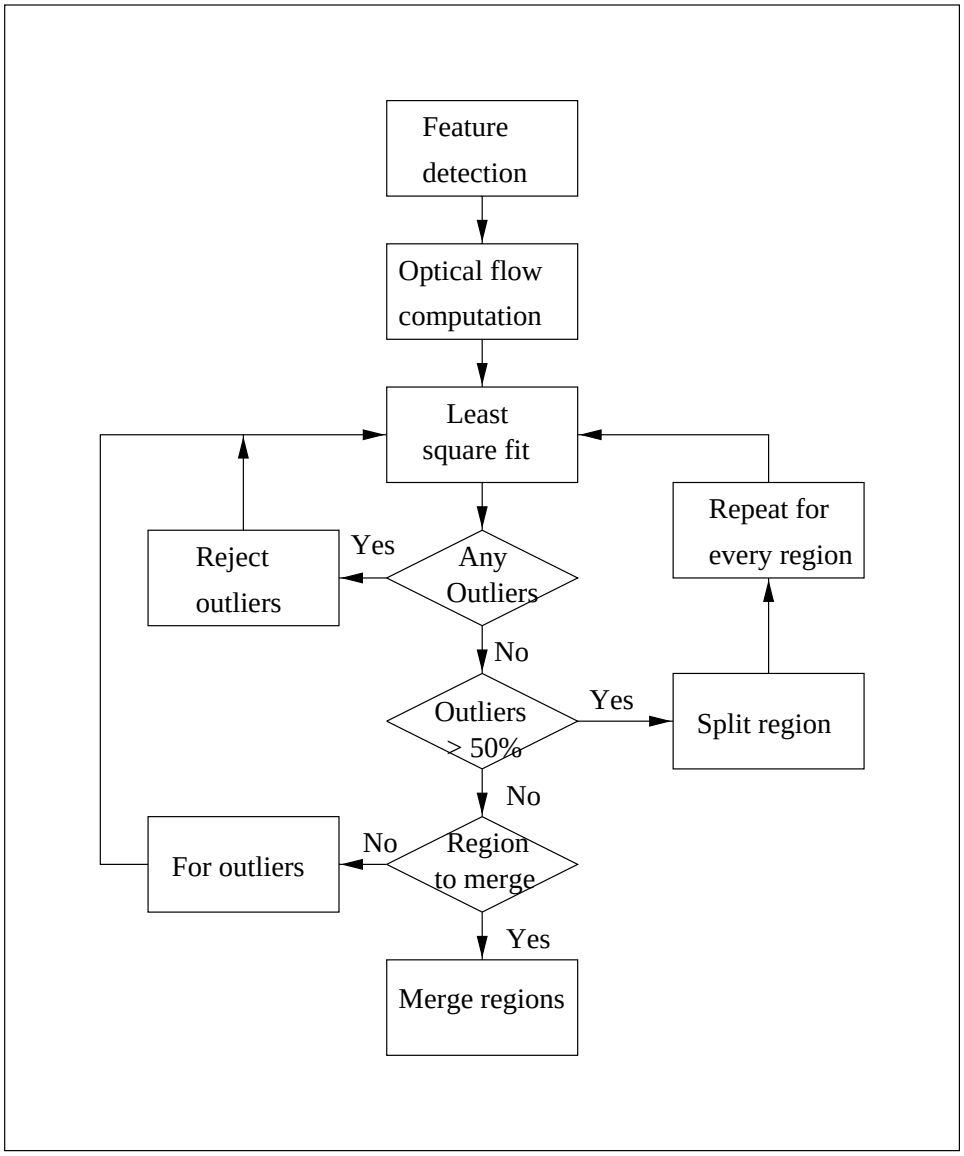


Fig. 8.3. Use of planar motion model: System block diagram.

An alternative approach for motion segmentation was proposed by Adiv [1]. In this approach, an affine motion model with six parameters is used to describe the optical flow at a pixel in the image plane. A six dimensional Hough space is formed. Each flow vector votes to one or more bins of this Hough space, depending on the error between the model flow (computed using the affine motion model) and the actual flow (computed at the pixel using an optical flow algorithm). However, the success of the Hough method relies on prior knowledge about the range of parameter values, and the accuracy relies on the resolution of the Hough space. Unlike the Hough method, the method described above does not have to rely on knowledge of the range of the model parameters. Instead, the parameters are estimated from the available data, and refined recursively by removing the outliers.

Various stages of the motion-based segmentation algorithm were tested using the same image sequence containing the truck crossing the runway. Figure 8.4 (a) shows frame 50 of the sequence. Features detected in this image are shown in Figure 8.4 (b). Figure 8.4 (c) shows the computed optical flow using our optical flow algorithm, and Figure 8.4 (d) shows the set of feature vectors identified as obstacles.

Similar results for a sequence of synthetic images are shown in in Figure 8.5. It can be seen that the planar motion model is fit to the runway features such as the text characters, and the features corresponding to the obstacle aircraft which is moving and at a height above the runway are separated out as outliers. In the next chapter, another application of this approach is explored, where the planar surfaces which could potentially contain such text characters are identified, and perspective correction is applied to these features using the calculated parameters of the planar surfaces.

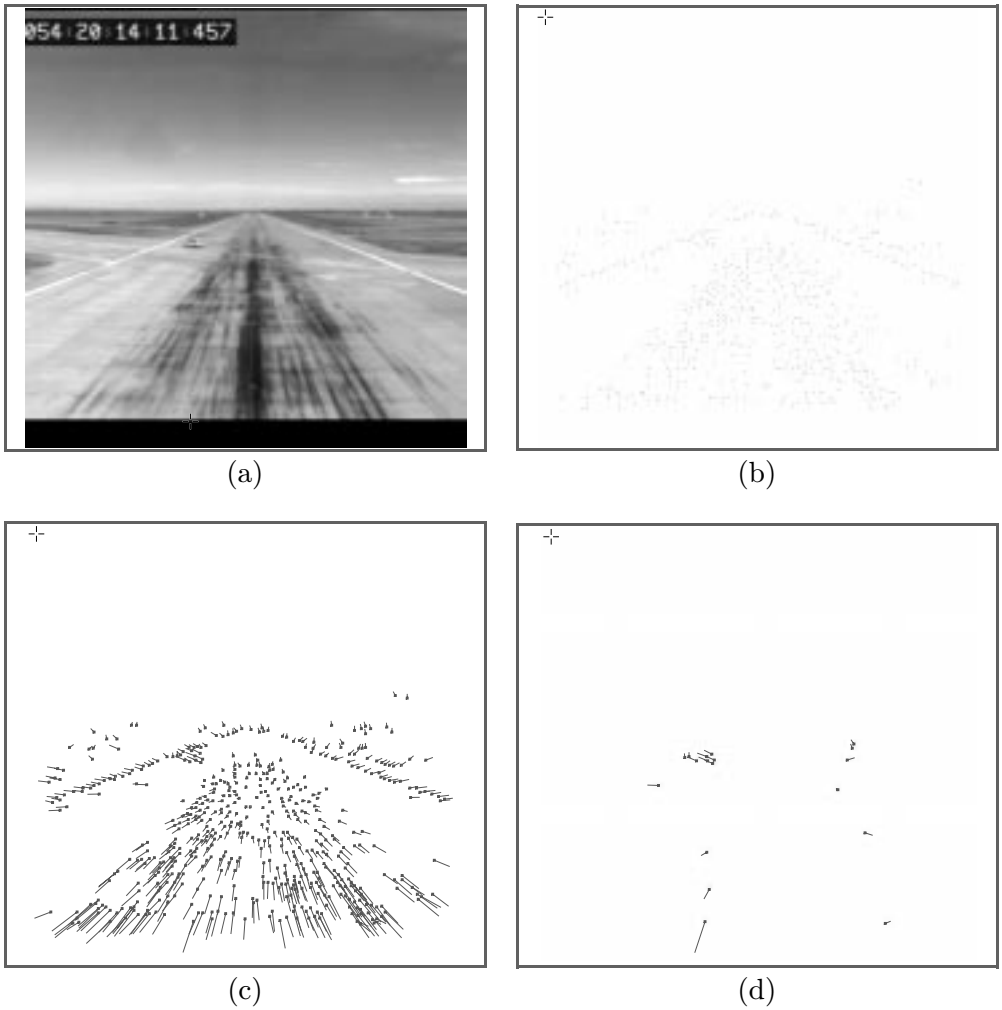


Fig. 8.4. Results obtained for a real image sequence in which a truck is crossing the runway: (a) Frame 60 of the image sequence (b) Detected Features (c) Computed optical flow (d) Optical flow vectors detected as due to obstacle

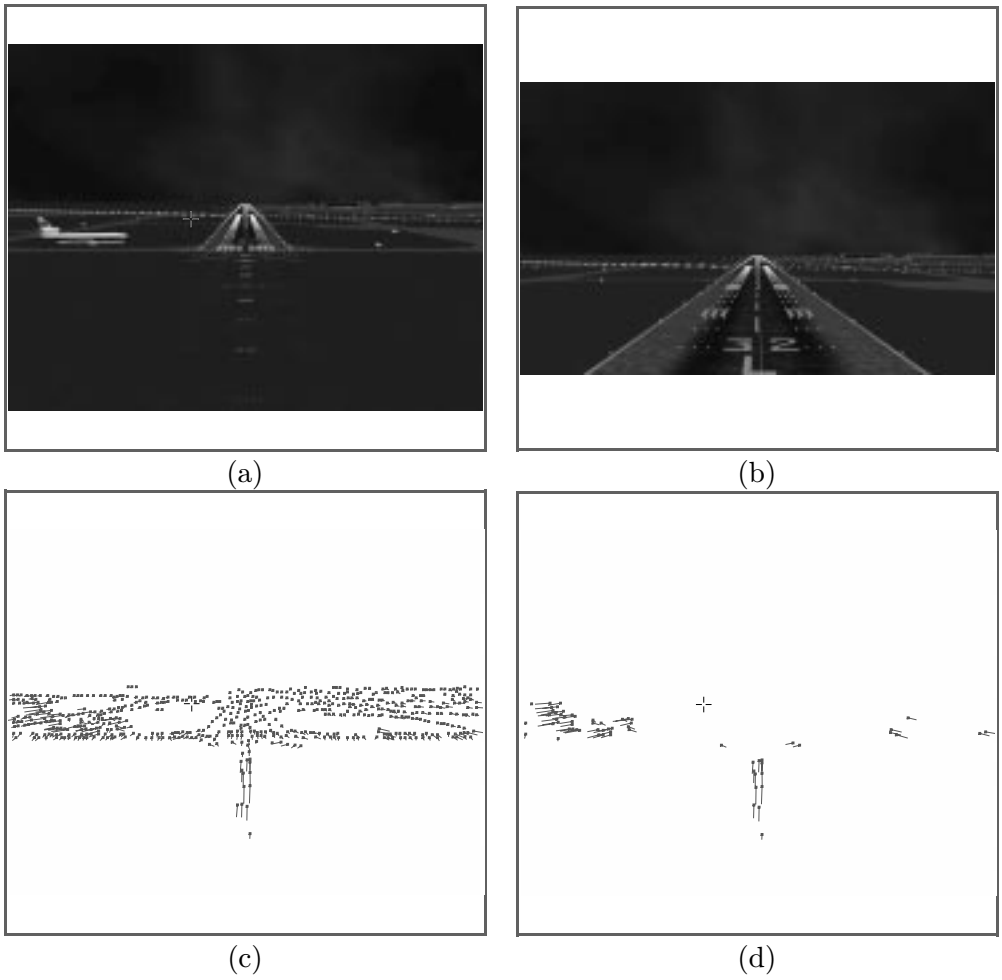


Fig. 8.5. Results obtained for a simulated image sequence: (a), (b) Frames 3 and 150 of the image sequence. (c) Computed image motion for all the features in frame 3. (d) Image motion vectors detected after subtracting the features belonging to the planar runway model.

Chapter 9

Motion-Based Segmentation for Text Extraction

The principle of using planar motion model, used by Devadiga [20] for image segmentation could also be useful in other applications, such as detection and localization of text in a video image sequence. There is a considerable amount of text occurring in video that is a useful source of information. The text that occurs naturally in the 3-D scene being imaged is called scene text. The scene text can have any orientation, and its image will be distorted by perspective projection in addition to being subject to the illumination conditions of the scene and susceptible to partial occlusion by other objects.

There has been very little research on recognizing scene text in video per se. The research that resembles this work the most is on recognition of vehicle license plates, which shares some characteristics of scene text [18, 19]. However, this work makes restrictive assumptions about the capture process, such as the vehicle and camera positions leading to near normal projection and near horizontal orientation. In contrast, a general-purpose scene text detection algorithm must handle text in all orientations. Cui and Huang [19] also use information from multiple frames, and correct for perspective projection distortion using the fact that the characters lie on a plane. Other research in scene text extraction is done by Ohya et al. [48] for high contrast scenes, and Winger et al. [63] for low-contrast scenes, both using adaptive thresholding.

Scene text typically exists on a planar surface in a 3-D scene. As the camera or the object moves, the motion of the text features should satisfy planar motion in 3-D. This property can be exploited to separate text features from features due to other objects, which are likely to be at different random depths, and thus do not satisfy the planar constraint. A sequence of images can be used to segment different planar surfaces in the image, determine their parameters, and remove outliers corresponding to clutter which do not fit any such surface, or is in motion with respect to these surface.

Once the parameters of the plane are estimated, the perspective effect of the camera on the characters can be compensated. Furthermore, a number of image frames can be registered using the planar motion parameters. The redundancy of information in the images can be used to generate a single image with a higher resolution than the

original images by a process known as super-resolution [29, 42]. Perspective correction as well as super-resolution could improve the accuracy of Optical Character Recognition (OCR).

The following sections describe in detail the planar motion model, motion segmentation using this model as performed in Devadiga's work [20], and the adaptation of the procedure for scene text segmentation. Using this procedure, the motion model parameters of the planar surfaces are estimated along with their covariances. These parameters are used to determine the camera motion in terms of the linear and angular velocity, and the scene structure in terms of the plane normal vectors. Since the camera motion parameters are the same for all planar surfaces, these parameters, as well as the plane normals of multiple planar surfaces are combined by using linear and non-linear methods. The plane normal vectors are then used for performing perspective correction on the planar surfaces. Results on simulated and real image sequences are described.

9.1 Planar motion model

Let $X = (X_0, X_1, X_2)^t$ be the 3-D coordinates of a point in the camera coordinate system, in which the X_0 axis passes through the optical axis of the sensor, and the $X_1 - X_2$ axes form the image plane. The perspective projection of the point in the image plane is given by:

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{X_0} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{9.1}$$

The image pixel coordinates (z_1, z_2) are related to the camera plane coordinates (x_1, x_2) by the following equation involving the internal parameters of the camera:

$$\begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ o_1 & \alpha_1 & 0 \\ o_2 & s_1 & \alpha_2 \end{bmatrix} \begin{bmatrix} 1 \\ z_1 \\ z_2 \end{bmatrix} \tag{9.2}$$

Here, (o_1, o_2) represents the pixel origin coordinates in terms of the camera plane coordinates, α_1 and α_2 are the scales of the pixel coordinate system, and s_1 is the skew parameter to deal with pixel axes which are not exactly perpendicular. In the simple case of square shaped pixels, $s_1 = 0$ and $f = \alpha_1^{-1} = \alpha_2^{-1}$ is the focal length of the camera in pixels.

Let the relative motion between the camera and the scene be modeled by a translational velocity of $V = (V_0, V_1, V_2)^t$ and a rotational velocity of $W = (W_0, W_1, W_2)^t$.

Due to this relative motion, the image point corresponding to X will also move. Differentiating equation (9.1) with respect to time, the image motion is given by:

$$\begin{aligned}\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \frac{1}{X_0^2}\begin{bmatrix} X_0\dot{X}_1 - X_1\dot{X}_0 \\ X_0\dot{X}_2 - X_2\dot{X}_0 \end{bmatrix} \\ &= \frac{1}{X_0}\begin{bmatrix} \dot{X}_1 - x_1\dot{X}_0 \\ \dot{X}_2 - x_2\dot{X}_0 \end{bmatrix} = \frac{1}{X_0}\begin{bmatrix} -x_1 & 1 & 0 \\ -x_2 & 0 & 1 \end{bmatrix}\begin{bmatrix} \dot{X}_0 \\ \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} \end{aligned} \tag{9.3}$$

The apparent motion of the 3-D point X due to the camera translation V and rotation W is given by the Coriolis equation:

$$\dot{X} = -V - W \times X = -V + X \times W \tag{9.4}$$

or

$$\begin{aligned}\begin{bmatrix} \dot{X}_0 \\ \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} &= -\begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} 0 & -X_2 & X_1 \\ X_2 & 0 & -X_0 \\ -X_1 & X_0 & 0 \end{bmatrix}\begin{bmatrix} W_0 \\ W_1 \\ W_2 \end{bmatrix} \\ &= -\begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix} + X_0\begin{bmatrix} 0 & -x_2 & x_1 \\ x_2 & 0 & -1 \\ -x_1 & 1 & 0 \end{bmatrix}\begin{bmatrix} W_0 \\ W_1 \\ W_2 \end{bmatrix} \end{aligned} \tag{9.5}$$

Substituting this in equation (9.3), we get the image motion:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \frac{1}{X_0}\begin{bmatrix} x_1 & -1 & 0 \\ x_2 & 0 & -1 \end{bmatrix}\begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix} + \begin{bmatrix} x_2 & x_1x_2 & -(1+x_1^2) \\ -x_1 & 1+x_2^2 & -x_1x_2 \end{bmatrix}\begin{bmatrix} W_0 \\ W_1 \\ W_2 \end{bmatrix} \tag{9.6}$$

where the two terms represent the translational and the rotational components of the image motion. It can be seen that only the translational component of the image motion is a function of the depth X_0, which cannot be directly determined from the image. However, if the point lies on a planar surface with the normal vector $K = (K_0, K_1, K_2)^t$ such that the equation of the plane in the camera coordinate system is:

$$K^tX = K_0X_0 + K_1X_1 + K_2X_2 = 1 \tag{9.7}$$

Then, dividing equation (9.7) by X_0 and substituting from equation (9.1) gives the depth X_0 as:

$$(X_0)^{-1} = K_0 + K_1 x_1 + K_2 x_2 \tag{9.8}$$

Let the image motion of such a point be denoted by $\hat{u} = (\hat{u}_1, \hat{u}_2)^t$. Then, by substituting the value of $(X_0)^{-1}$ in equation (9.6), we get:

$$\begin{aligned} \hat{u} = \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \end{bmatrix} &= (K_0 + K_1 x_1 + K_2 x_2) \begin{bmatrix} x_1 & -1 & 0 \\ x_2 & 0 & -1 \end{bmatrix} \begin{bmatrix} V_0 \\ V_1 \\ V_2 \end{bmatrix} \\ &+ \begin{bmatrix} x_2 & x_1 x_2 & -(1+x_1^2) \\ -x_1 & 1+x_2^2 & -x_1 x_2 \end{bmatrix} \begin{bmatrix} W_0 \\ W_1 \\ W_2 \end{bmatrix} \end{aligned} \tag{9.9}$$

Using the above equation, the theoretical image motion for any point on a planar surface can be written as a quadratic function of image coordinates.

$$\begin{aligned} \hat{u}_1 &= a_1 + a_3 x_1 + a_5 x_2 + a_7 x_1^2 + a_8 x_1 x_2 \\ \hat{u}_2 &= a_2 + a_4 x_1 + a_6 x_2 + a_7 x_1 x_2 + a_8 x_2^2 \end{aligned} \tag{9.10}$$

where the vector of eight coefficients $\vec{a} = (a_1 \ldots a_8)^t$ called the planar motion model parameters (or simply model parameters) can be expressed in terms of the relative motion parameters (V and W), as well as the structure parameter (K) of the planar surface as:

$$\vec{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} = \begin{bmatrix} -V_1 K_0 - W_2 \\ -V_2 K_0 + W_1 \\ V_0 K_0 - V_1 K_1 \\ -V_2 K_1 - W_0 \\ -V_1 K_2 + W_0 \\ V_0 K_0 - V_2 K_2 \\ V_0 K_1 - W_2 \\ V_0 K_2 + W_1 \end{bmatrix} \tag{9.11}$$

It should be noted that even if the internal parameters of the camera are not known, the planar motion can be expressed in terms of the pixel coordinates (z_1, z_2) by a applying a suitable transformation on $\vec{a}$. The form of equation (9.10) remains the same, and the transformation can be easily derived using equation (9.2). Hence, motion

segmentation can be performed even without the knowledge of the internal parameters of the camera. However, in that case one cannot compute the motion and structure parameters V, W, and K from the model.

If the image motion at a number of points on the planar surface is known, each point results in two equations containing eight unknown parameters. At least four points are required to solve these equations for model parameters, assuming that all four points belong to the same planar motion. If more than four points are available, then a least squares fit can be used to compute the best model which fits all the points. The relative motion parameters V and W between the camera and the scene need not be known in advance to solve these equations.

The parameter vector $\vec{a} = (a_1, a_2 \ldots a_8)^t$ can be estimated using a least-squares minimization of the error between the observed image motion $u^i = (u_1^i, u_2^i)^t$ and the parametric image motion $\hat{u}^i = (\hat{u}_1^i, \hat{u}_2^i)^t$ for the i^{th} pixel given by:

$$\sum_i \|\hat{u}^i - u^i\|^2 = \sum_i \left[(\hat{u}_1^i - u_1^i)^2 + (\hat{u}_2^i - u_2^i)^2\right] = [J\vec{a} - e]^t \, \Sigma_u^{-1} \, [J\vec{a} - e] \tag{9.12}$$

where Σ_u is the covariance matrix of the time gradient, and

$$J = \begin{bmatrix} J^1 \\ J^2 \\ \vdots \\ J^n \end{bmatrix}, \; e = \begin{bmatrix} e^1 \\ e^2 \\ \vdots \\ e^n \end{bmatrix} \tag{9.13}$$

$$J^i = \begin{bmatrix} 1 & 0 & x_1^i & 0 & x_2^i & 0 & (x_1^i)^2 & x_1^i x_2^i \\ 0 & 1 & 0 & x_1^i & 0 & x_2^i & x_1^i x_2^i & (x_2^i)^2 \end{bmatrix}, \; e^i = \begin{bmatrix} u_1^i \\ u_2^i \end{bmatrix} \tag{9.14}$$

The least squares estimate $\hat{a}$ and its covariance Σ_a are given by:

$$\hat{a} = \left[J^t \Sigma_u^{-1} J\right]^{-1} \left[J^t \Sigma_u^{-1} e\right] \tag{9.15}$$

$$\Sigma_a = \left[J^t \Sigma_u^{-1} J\right]^{-1} \tag{9.16}$$

The 8-parameter equation given above is sufficient to represent the motion of a planar surface. However, the estimation of the quadratic parameters (a_7, a_8) may not be robust if small parts of the image are used. Hence, other models may be used in intermediate stages of segmentation. For example, the affine motion model with 6

parameters is frequently used for motion segmentation.

$$\begin{aligned} \hat{u}_1 &= a_1 + a_3 x_1 + a_5 x_2 \\ \hat{u}_2 &= a_2 + a_4 x_1 + a_6 x_2 \end{aligned} \tag{9.17}$$

9.2 Approaches for estimation of image motion

As seen above, if image motion at a number of points on the planar surface is available, it is possible to estimate the model parameters. This image motion may be estimated by observing the apparent motion of a discrete set of features or brightness patterns in the images. Two distinct approaches have been developed for the computation of motion from image sequences.

The first approach, known as the *feature-based approach*, is based on extracting a set of relatively sparse, but highly discriminatory, two-dimensional features in the images corresponding to three-dimensional object features, such as corners, lines corresponding to edges demarcating the surfaces of the object in 3-D, etc. Such points and/or lines are extracted from each image, and inter-frame correspondence is then established between these features to obtain an estimate of motion of the features from one frame to another.

The other approach involves computing the two-dimensional field of instantaneous velocities of brightness values in the image plane, known as optical flow, by using spatial and temporal derivatives of the image brightness. Under favorable conditions of image brightness being constant over a short period of time, the temporal gradient g_0 and the spatial gradients g_1 and g_2 satisfy the following optical flow constraint equation:

$$g_0 + g_1 u_1 + g_2 u_2 = 0 \tag{9.18}$$

Using this equation, there is only one equation between two unknowns (u_1, u_2) for a single pixel. Due to this, only the normal flow – i.e., flow in the direction of the gradient – can be determined using a single point. This is an intrinsic ambiguity, known as the aperture effect, which is seen clearly in Figure 9.1 (a). To solve this problem, Lucas and Kanade [41] assumed that the image motion is approximately constant in a small window around every point. Using this constraint, more equations are obtained using the neighboring points, and the full optical flow can be estimated. Such an estimate is reliable when the window has points with gradients in different directions. This is possible if a corner is located inside the window, where the aperture problem is avoided as shown in Figure 9.1 (b).

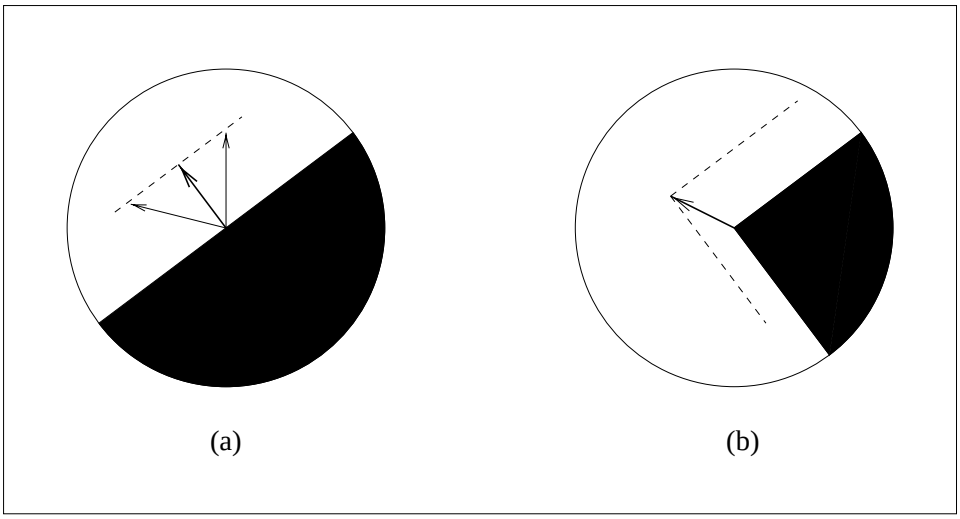


Fig. 9.1. Aperture problem: (a) In the case of an edge, only the component of motion normal to the edge can be determined. (b) In the case of a corner, the aperture problem is avoided, and the motion can be uniquely determined.

Alternatively, image gradients can be used directly to estimate the model parameters as done by Black [11]. The optical flow constraint in equation (9.18) can then be expressed as a minimization problem. Least squares approach can be used to minimize the following objective function:

$$\begin{aligned} & \sum_i \left[g_0^i + g_1^i \hat{u}_1^i + g_2^i \hat{u}_2^i \right]^2 \\ = & \sum_i \left[g_0^i + g_1^i (a_1 + a_3 x_1^i + a_5 x_2^i \ldots) + g_2^i (a_2 + a_4 x_1^i + a_6 x_2^i \ldots) \right]^2 \\ = & \left[J_g \vec{a} - e_g \right]^t \Sigma^{-1} \left[J_g \vec{a} - e_g \right] \end{aligned} \tag{9.19}$$

where Σ is the covariance matrix of the time gradient, and

$$J_g = \begin{bmatrix} J_g^1 \\ J_g^2 \\ \vdots \\ J_g^n \end{bmatrix}, \; e_g = - \begin{bmatrix} g_0^1 \\ g_0^2 \\ \vdots \\ g_0^n \end{bmatrix} \tag{9.20}$$

$$J_g^i = \begin{bmatrix} g_1^i & g_2^i \end{bmatrix} J^i = \begin{bmatrix} g_1^i & g_2^i \end{bmatrix} \begin{bmatrix} 1 & 0 & x_1^i & 0 & x_2^i & 0 & (x_1^i)^2 & x_1^i x_2^i \\ 0 & 1 & 0 & x_1^i & 0 & x_2^i & x_1^i x_2^i & (x_2^i)^2 \end{bmatrix}_i \tag{9.21}$$

The least squares estimate $\hat{a}$ and its covariance Σ_a are given by:

$$\hat{a} = \left[J_g^t \Sigma^{-1} J_g \right]^{-1} \left[J_g^t \Sigma^{-1} e_g \right] \tag{9.22}$$

$$\Sigma_a = \left[J_g^t \Sigma^{-1} J_g \right]^{-1} \tag{9.23}$$

Direct application of the gradient based methods yield accurate results only when the image motion of the pixels is less than 1 to 2 pixels per frame. To deal with larger image velocities, pyramid approach [11] is used, as shown in Figure 9.2. A pyramid of images is formed by reducing the original image to lower resolutions. The image gradients are first computed at the coarsest resolution, and the image velocities or model parameters are estimated using these gradients. These are used to warp the images at the next finer resolution, and the gradients are computed using the warped images. The process is repeated until the finest resolution is reached.

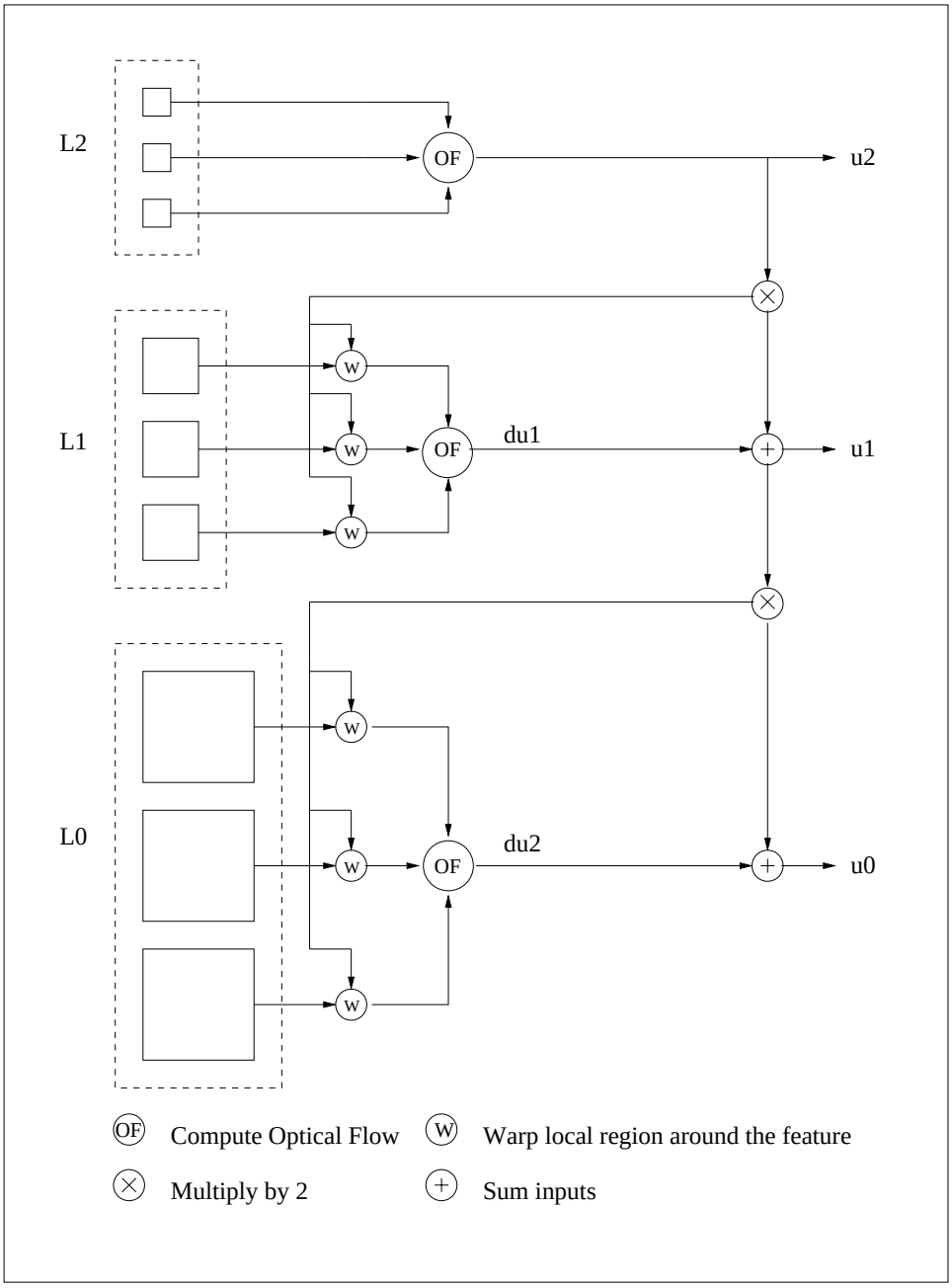


Fig. 9.2. Estimation of planar motion parameters ('a' parameters) using the pyramid estimation framework.

9.3 Multiple motion segmentation and estimation

Under ideal situations, where all image motion vectors result from a single planar surface motion, a least squares fit approach can be easily used to recover the model parameters. If noise in the image motion is small, the least squares solution provides a good estimate of the parameter vector. However, if a few data points have large errors, then these points can have a significant effect on the accuracy of the recovered model parameters. Such data points, called outliers which usually do not belong to the planar surface, should be detected and removed before reliable parameter estimates can be obtained. The problem is worse when the image contains multiple planar surfaces; the least squares approach if used directly, yields an estimate which is an 'average' of the motion parameters of all the surfaces. For obtaining reliable estimates of parameters, the image should be segmented into parts before applying the least squares. However, this is a chicken and egg problem, since the segmentation is what we want to obtain by analyzing the image motion.

A considerable amount of work has been done on segmenting a scene into planar surfaces. Adiv [1] used Hough transform to map the image motion vectors into bins corresponding to the affine motion model. The bins with a large number of points, corresponding to the parts of the image satisfying the same motion model are grouped together. Wang et al. [62] divides the image into non-overlapping regions and fits the affine motion model to these regions. A k-means algorithm is used to cluster the affine parameters of all the regions, using a distance measure between a pair of parameters. All image points are reclassified using the affine parameters of the clusters, and the parameters are recomputed using the new classification. The segmentation obtained is used as initial segmentation for next frames. Borshukov et al. [13] observe that the adaptive k-means used in [62] gives an 'average fit' model for each layer. They claim that a better result is obtained by replacing the clustering step with a merge step, which in effect picks the model having the smallest residual for each layer – i.e., the 'best fit' model. Also, they observe that labeling the classes of all motion vectors in a single step produces unsatisfactory results. Hence, they use a multistage strategy, which automatically decides for the best number of motion classes. Another approach by Bouthemy and Francois [14] uses a two-term energy function, and a relaxation algorithm partitions the image into regions with different motion models. The first term is the prior energy term, depending on the consistency of region labels in the neighborhood of each pixel whereas the second term is the observation label interaction energy term which

depends on the error of fit between the motion model and observation using spatial and temporal gradients. A deterministic relaxation method (ICM) is used to assign and modify labels of pixels so that the energy is minimized.

For the runway obstacle detection, Devadiga [20] assumed a piecewise planar model for the scene, and a used a split and merge approach to segment the scene. The scene is first hypothesized to be perfectly planar, and planar motion model is applied to the whole image. A robust method can be used to check if a single planar surface fits all or most of the points. If it does not, it is concluded that the initial hypothesis of a single planar surface has failed, and the scene is hypothesized to be piecewise planar. The scene is split into parts, and the procedure is recursively performed by separately applying the planar motion models to each of the pieces. On the other hand, at any stage, if two neighboring pieces satisfy a similar model, they are merged into one, and the model parameters recomputed using points in both the pieces. This process of splitting and merging is continued until most of the points are fitted on planar surfaces. The points which cannot be fit are classified as outliers.

At present, the segmentation is interactive so that features of some of the above methods can be implemented and explored. The basic framework is similar to the approach used by Devadiga [20]. However, instead of first computing the image motion at selected feature points and using these to compute the motion parameters, the image gradients are directly used for motion parameter computation as in [11]. This enables the use of more information from the image and is expected to yield better estimates of the parameters.

To start the segmentation, the user specifies the initial regions, which could also be the entire scene. The algorithm then calculates the parametric image motion in each of these regions, separating the pixels which it suspects to be outliers. A new region label can be interactively assigned to these outliers, and the process repeated. Outliers can also be reclassified according to the best-fitting regions. A region can be split into a number of parts interactively, if the motion model fit is unsatisfactory. For outlier removal, an approach similar that of Bouthemy and Francois [14] is used. This results in smoother boundaries between regions and discourages fragmentation into small regions.

9.4 Structure and motion parameters from a single planar surface

The structure parameter (K), and the motion parameters (V, W) can be computed up to a scale factor and an ambiguity, from the planar motion parameter vector ($\vec{a}$) using

the following method [32]. It should be noted that the internal parameters of the camera are required for determining the structure and motion parameters.

The planar motion parameters $\vec{a}$ are expressed in terms of the camera motion parameters V and W, and the plane normal K parameters by the equation (9.11). The following notation can also be used for that expression:

$$M = \begin{bmatrix} 0 & -a_7 & -a_8 \\ a_1 & a_3 & a_5 \\ a_2 & a_4 & a_6 \end{bmatrix}, W_\times = \begin{bmatrix} 0 & -W_2 & W_1 \\ W_2 & 0 & -W_0 \\ -W_1 & W_0 & 0 \end{bmatrix} \tag{9.24}$$

Then,

$$VK^t + W_\times = \begin{bmatrix} V_0K_0 & V_0K_1 - W_2 & V_0K_2 + W_1 \\ V_1K_0 + W_2 & V_1K_1 & V_1K_2 - W_0 \\ V_2K_0 - W_1 & V_2K_1 + W_0 & V_2K_2 \end{bmatrix} = -M + V_0K_0I \tag{9.25}$$

To extract V, W, and K from $\vec{a}$ (or equivalently M), the following steps are used:

1. Calculate $\tilde{M}$ which differs from M only by a scalar multiple of the identity matrix, and whose trace is zero.
$$\tilde{M} = M - \mathrm{tr}(M/3)I \tag{9.26}$$

2. The anti-symmetric part of $\tilde{M}$ has only 3 independent elements which can be written as a vector:
$$m = \frac{1}{2}\begin{bmatrix} \tilde{M}_{32} - \tilde{M}_{23} & \tilde{M}_{13} - \tilde{M}_{31} & \tilde{M}_{21} - \tilde{M}_{12} \end{bmatrix}^t \tag{9.27}$$
If $m = 0$, the motion is a pure rotation with
$$V = 0\,,\; W = m \tag{9.28}$$
The plane normal K is indeterminate in this case. Otherwise, apply the following steps:

3. Compute eigenvalues $e_1 \geq e_2 \geq e_3$ and the corresponding eigenvectors v_1, v_2, v_3 of the symmetric part of $\tilde{M}$ given by:
$$\frac{1}{2}\left(\tilde{M} + \tilde{M}^t\right) = e_1v_1v_1^t + e_2v_2v_2^t + e_3v_3v_3^t \tag{9.29}$$

4. The two linearly independent sets of solutions for K and V are given up to an arbitrary scale factor γ as:

$$\begin{aligned} \gamma K &= (v_3\sqrt{e_2 - e_3} \pm v_1\sqrt{e_1 - e_2})\sqrt{e_1 - e_3} \\ \gamma^{-1}V &= (v_3\sqrt{e_2 - e_3} \mp v_1\sqrt{e_1 - e_2})/\sqrt{e_1 - e_3} \end{aligned} \tag{9.30}$$

5. The angular velocity W (independent of γ) for each set of solutions is given by:

$$W = m - \frac{K \times V}{2} \tag{9.31}$$

The scale factor γ reflects the fact that a perspective camera does not differentiate between a distant surface with large camera velocity from a nearer surface with correspondingly smaller camera velocity. In each set of solutions, the sign of γ is determined from the fact one of the two cases – $\gamma > 0$ or $\gamma < 0$ – corresponds to the solutions behind the camera, which are physically invalid. The magnitude of γ can be calculated for normalized solutions in which V is unit vector, and the plane normal K can be scaled appropriately. The ambiguity due to the two linearly independent sets of solutions cannot be resolved if there is only one planar surface. Hence, both solutions should be checked when performing perspective correction using the plane normal. However, at present, the 'correct' solution is chosen manually.

9.5 Structure and motion parameters from multiple planar surfaces

In the case of the scene containing multiple planar surfaces, the linear and angular velocities of the camera are identical for all the surfaces. If these parameters are determined by the above method, the constraint is not utilized, and each planar surface would give rise to different estimates of the camera motion. Furthermore, a single set of motion model parameters give rise to two sets of solutions for structure and motion parameters. Hence, a novel method is proposed to combine the estimates of all planar surfaces to increase the accuracy of the structure and motion parameters, and to remove the ambiguity due to multiple solutions.

Use of image motion to directly estimate the structure and motion parameters have been previously proposed so that the constraints on the camera motion are utilized. Horn [26] describes the method to determine the camera motion as well as depth

of all the scene points using the optical flow of the points for a general scene. Factorization methods [60, 44] are also used to determine the structure and motion parameters from multiple frames, especially for orthographic and para-perspective motion models. However, it is noted that these methods require the knowledge of the full image motion. The direct use of image gradients instead of full optical flow makes the problem under-constrained, requiring smoothness constraints.

In this work, a piecewise planar model is assumed instead of a general scene to get the required smoothness constraint. The input to this algorithm are the motion model parameters of all the planar surfaces, computed independently after motion segmentation. The algorithm combines these parameters to increase the accuracy of the structure and motion parameters, and removes the ambiguity due to multiple solutions.

Let the planar motion parameters of each planar surface be given by $\vec{a}^i$, and the respective plane normal vectors be K^i. These, as well as the camera motion parameters V and W can be stacked as follows:

$$\vec{A} = \begin{bmatrix} \vec{a}^1 \\ \vec{a}^2 \\ \vdots \\ \vec{a}^n \end{bmatrix}, \vec{K} = \begin{bmatrix} K^1 \\ K^2 \\ \vdots \\ K^n \end{bmatrix}, \vec{L} = \begin{bmatrix} W \\ V \end{bmatrix}, \vec{P} = \begin{bmatrix} \vec{K} \\ \vec{L} \end{bmatrix} \tag{9.32}$$

Using equation (9.11), the relations between these parameters can be written in two forms for all planar surfaces $i = 1 \ldots n$:

$$\begin{bmatrix} G_2 & G_3(K^i) \end{bmatrix} \vec{L} = \vec{a}^i \tag{9.33}$$

$$G_1(\vec{L})\, K^i = \vec{a}^i - \begin{bmatrix} G_2 & O_{8\times 3} \end{bmatrix} \vec{L} \tag{9.34}$$

where $O_{8\times 3}$ is a zero matrix and

$$G_1 = \begin{bmatrix} -V_1 & 0 & 0 \\ -V_2 & 0 & 0 \\ V_0 & -V_1 & 0 \\ 0 & -V_2 & 0 \\ 0 & 0 & -V_1 \\ V_0 & 0 & -V_2 \\ 0 & V_0 & 0 \\ 0 & 0 & V_0 \end{bmatrix}, G_2 = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}, G_3(K^i) = \begin{bmatrix} 0 & -K_0^i & 0 \\ 0 & 0 & -K_0^i \\ K_0^i & -K_1^i & 0 \\ 0 & 0 & -K_1^i \\ 0 & -K_2^i & 0 \\ K_0^i & 0 & -K_2^i \\ K_1^i & 0 & 0 \\ K_2^i & 0 & 0 \end{bmatrix} \tag{9.35}$$

These equations can be iteratively solved using linear least squares to get solutions for K^i and $\vec{L}$. However, the objective function which is optimized by this procedure does not correspond to the physical objective function to be minimized which is:

$$\left[\vec{A} - f(\vec{P})\right]^t \Sigma_A^{-1} \left[\vec{A} - f(\vec{P})\right] \tag{9.36}$$

subject to:

$$c(\vec{P}) = \frac{1}{2}\left(V_0^2 + V_1^2 + V_2^2 - 1\right) = 0 \tag{9.37}$$

where f denotes the function derived from equation (9.11), which computes $\vec{A}$ parameters from the structure and motion parameters combined in the vector $\vec{P}$. These equations can be solved using non-linear least squares with constraints. For this, Lagrange parameter λ is added to obtain:

$$\left[\vec{A} - f(\vec{P})\right]^t \Sigma_A^{-1} \left[\vec{A} - f(\vec{P})\right] + \left[c(\vec{P})\right]^t \lambda \tag{9.38}$$

where c and λ are column vectors in general, describing all the constraints on $\vec{L}$. However, in this particular case, they are scalars since there is only one constraint. The above expression is minimized by iteratively computing:

$$\Delta\vec{P}_+ = \begin{bmatrix} \Delta\vec{P} \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} F^t\Sigma_A^{-1}F & C^t \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} F^t\Sigma_A^{-1}\Delta\vec{A} \\ 0 \end{bmatrix} \tag{9.39}$$

where $\vec{P}_+$, formed by stacking $\vec{P}$ and λ is incremented by $\Delta\vec{P}_+$ at every iteration. F and C denote the Jacobians of the functions f and c, respectively, and are given by:

$$F = \frac{\partial A}{\partial P} = \begin{bmatrix} \frac{\partial A^1}{\partial K^1} & \cdots & \frac{\partial A^1}{\partial K^n} & \frac{\partial A^1}{\partial W} & \frac{\partial A^1}{\partial V} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \frac{\partial A^n}{\partial K^n} & \cdots & \frac{\partial A^n}{\partial K^n} & \frac{\partial A^n}{\partial W} & \frac{\partial A^n}{\partial V} \end{bmatrix} = \begin{bmatrix} G_1 & \ldots & 0 & G_2 & G_3^1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \ldots & G_1 & G_2 & G_3^n \end{bmatrix} \tag{9.40}$$

$$C = \frac{\partial C}{\partial P} = \begin{bmatrix} \frac{\partial C}{\partial K^1} & \cdots & \frac{\partial C}{\partial K^n} & \frac{\partial C}{\partial W} & \frac{\partial C}{\partial V} \end{bmatrix} = \begin{bmatrix} 0 & \ldots & 0 & 0 & V \end{bmatrix} \tag{9.41}$$

where G_1, G_2, and G_3^i are given by equation (9.35).

In actual practice, the non-linear iterations do not improve the accuracy significantly. However, the concept is useful for estimating the sensitivity of the parameters in $\vec{P}$. If the vector $\vec{A}$ has an error $\Delta\vec{A}$, the optimal solution changes by $\Delta\vec{P}$ to the first order approximation. From equation (9.39), using $E[\Delta\vec{A}\Delta\vec{A}^t] = \Sigma_A$, the covariance of

$\vec{P}_+$ is approximately given by:

$$\Sigma_{P_+} = E[\Delta\vec{P}_+\Delta\vec{P}_+^t] = \begin{bmatrix} Q & C^t \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Q & C^t \\ C & 0 \end{bmatrix}^{-1} \tag{9.42}$$

where $Q = F^t\Sigma_A^{-1}F$. The covariance of $\vec{P}$ is obtained by deleting the last row and column of Σ_{P_+} corresponding to the Lagrange parameter λ. The estimation of parameters from a single planar surface can be considered as a special case, and its sensitivity can be calculated using the above equations with $n = 1$.

In practice, the linear and non-linear estimation can be applied one after the other as follows:

- Determine K^i separately from a^i for each planar surface using the algorithm in Section 9.4.
- Use the above K^i's to form K as the initial guess. Repeat the following steps until convergence:
  1. Solve the system given by equation (9.33) for L with $i = 1 \ldots n$ using least squares with covariance.
  2. Normalize the velocity component of the solution to make it a unit vector. If $\tilde{L}$ and $\hat{L}$ denote the original and modified solutions, then
  $$\hat{L} = \begin{bmatrix} \hat{W} \\ \hat{V} \end{bmatrix} = \begin{bmatrix} \tilde{W} \\ \tilde{V}/\|\tilde{V}\| \end{bmatrix} \tag{9.43}$$
  3. Solve equations given by (9.34) separately for K^i with $i = 1 \ldots n$ using least squares with covariance.
- Refine the estimate using the following non-linear iterations:
  1. Compute the Jacobians F and C using equations (9.40) and (9.41), respectively. Also compute the residual error $\Delta\vec{A} = \vec{A} - f(\vec{P})$.
  2. Compute $\Delta\vec{P}$ using equation (9.39). Increment $\vec{P}$ by $\Delta\vec{P}$.
  3. Normalize the velocity component $\tilde{V}$ of the solution $\tilde{P}$. Appropriately multiply the plane normals in $\vec{K}$ by the norm of the $\tilde{V}$.
  $$\hat{P} = \begin{bmatrix} \hat{K} \\ \hat{W} \\ \hat{V} \end{bmatrix} = \begin{bmatrix} \hat{K}\|\tilde{V}\| \\ \tilde{W} \\ \tilde{V}/\|\tilde{V}\| \end{bmatrix} \tag{9.44}$$

- Estimate the covariance of $\vec{P}$ using equation (9.42).

9.6 Correction of perspective distortion

Once the plane normal parameter for each surface is known, the surfaces can be rotated so as to face the camera axis. The rotation is performed around the axis perpendicular to the camera axis as well as the plane normal. If the plane normal K is along the axis X_0, no perspective correction is required. Otherwise, if k is the unit vector in the direction of K, the rotation matrix is then given by:

$$R(K) = R(k) = \begin{bmatrix} k_0 & k_1 & k_2 \\ -k_1 & \frac{k_2^2+k_0k_1^2}{k_1^2+k_2^2} & -\frac{k_1k_2(1-k_0)}{k_1^2+k_2^2} \\ -k_2 & -\frac{k_1k_2(1-k_0)}{k_1^2+k_2^2} & \frac{k_1^2+k_0k_2^2}{k_1^2+k_2^2} \end{bmatrix} \tag{9.45}$$

This compensates for the perspective distortion. However, the rotation around the camera axis cannot be compensated using this method. The rotated coordinates, corresponding to the coordinate system (t_1, t_2) of the planar surface are then given by:

$$\begin{bmatrix} \alpha \\ t_1 \\ t_2 \end{bmatrix} \propto R(K) \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \tag{9.46}$$

where α is the zoom factor which determines the size of the compensated image. For warping the image using the rotation matrix, the inverse transformation is used. For each (t_1, t_2) on a discrete grid, the corresponding image coordinate (x_1, x_2) is calculated. Since (x_1, x_2) is not an integer in general, the pixel value is obtained by interpolating from the neighboring pixels using bilinear transformation.

9.7 Results

The application of the planar fit was tested on a number of simulated and real image sequences containing text or other patterns on planar surfaces. In the case of simulated image sequences, the motion parameters of the camera, the initial position and orientation of the camera, and the equations of the planar surfaces were pre-specified.

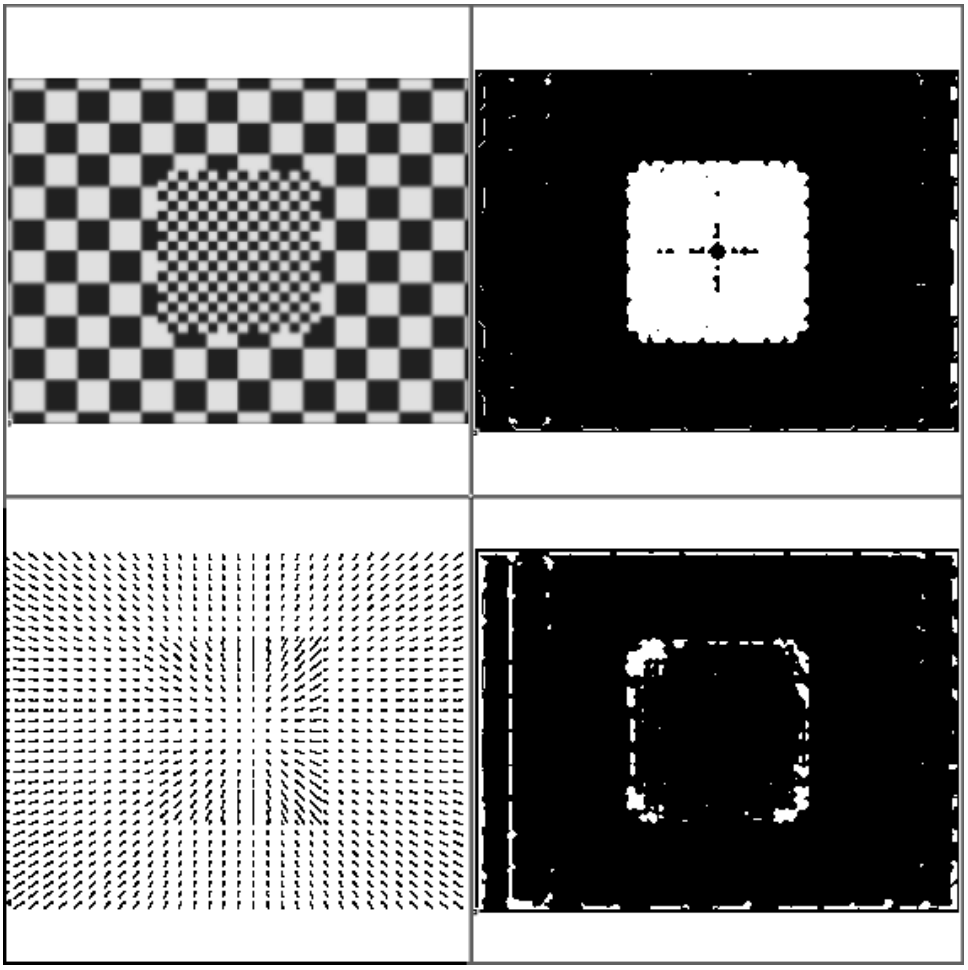
Figure 9.3 (a) shows an image from a simulated image sequence, containing two planar surfaces, with one occluded by another. The camera is moving towards these

surfaces, resulting in a radial pattern of image motion for each surface. The inner region which is nearer to the camera has a larger image motion. Using the image sequence, parameters of motion were first computed assuming a single planar motion, and the outliers were systematically removed. The outliers were assigned another region label, and a planar surface was fit to these. Outliers obtained by fitting using these two region models were then reclassified according to the model which they fit the best. Figure 9.3 (b) shows the segmentation into two regions. Figure 9.3 (c) shows the image motion field obtained using the parametric models of the two regions and Figure 9.3 (d) shows the pixels classified as outliers in any of the regions. The segmentation is quite good in this example, except for the central region of the image where pixels are misclassified. This is because the parametric image motion in the central region is very small, irrespective of the region label assigned to it. This problem can be corrected by performing connected component analysis to remove small regions, and merge these with larger regions in their neighborhood.

The segmentation procedure was also applied to real image sequences. A sample image from a sequence of an indoor scene is shown in Figure 9.4 (a). The scene contains a poster with text characters, pasted on a window with the camera moving towards the poster. The background outside the window is at a large distance. Since the images from the video camera were interlaced, a super-sampling procedure was applied to obtain two non-interlaced images from one interlaced image. In this scene, the segmentation picks up the poster as the dominant motion. In the next phase of segmentation, background is separated as another layer. However, background due to a perpendicular wall on the left is not separated, since it does not have enough features. The window bars are classified with the poster, since it is almost at the same depth as the poster. The segmentation is shown in Figure 9.4 (b). The image motion shown in Figure 9.4 (c) shows that the poster (except near the focus of expansion) as well as the window have larger motion than the distant background. Figure 9.4 (d) shows the detected outliers in each region which could make the parameter estimation unreliable.

Figure 9.5 shows the corresponding results for another outdoor scene captured from a moving bus. The scene contains a "STOP" sign with a distant background. In this case, the segmentation picks up the background as the dominant motion, and separates the traffic sign in the next step. However, some pixels in the other parts of the image are mis-classified to belong to the planar surface, possibly because they may be lying close to the plane of the traffic sign. However, these pixels are isolated, and can be easily separated by performing connected component analysis and reclassifying

(a) (b)



(c) (d)

Fig. 9.3. Segmentation of a simulated image sequence: (a) A frame in a simulated image sequence containing two planar surfaces, one occluding the other. (b) Segmentation obtained using the algorithm. (c) Estimate of image motion throughout the image. Note the discontinuity in the image motion at the occluding boundary. (d) Outliers in each region which could make the parameter estimation unreliable.

small components. Also, the uniform areas of the sign are not classified properly, since the image motion in these regions cannot be estimated.

The following experiment shows the result of estimation of plane normal, and its use in correcting the perspective distortion of the planar surfaces. The scene consists of two planar surfaces on both sides of the camera. The camera axis is parallel to these surfaces, and the camera is moving with a uniform linear velocity along its axis. Figure 9.6 (a) and (c) show the text surfaces on two sides of a camera, and an image frame obtained after projection is shown in Figure 9.6 (b). Using the sequence of images, the planar motion parameters were obtained separately for the two halves of the image. The plane normal vectors were extracted from these parameters. The error in the plane normal estimate in terms of the angle between the estimated and actual normal vector is around 3.5° and 4.7° in the left and right planes respectively. The two parts of the image warped separately to correct for the perspective distortion. The corrected images are shown in Figure 9.6 (d) and (e). Although the error in the planar normal is small, the correction is not very satisfactory. This happens when the normal from the camera center to the plane is very far from the part of the plane that is imaged by the camera. The reason for this problem will be explained in Chapter 10.

The experiment was performed by having the planes rotated 75° about the camera axis instead of 90° as done in the above experiment. The results of this experiment are shown in Figure 9.7. The perspective correction is somewhat better in this case, since the normal from the camera center to the plane is not as far from the part of the plane that is imaged by the camera.

Figure 9.8 (a) and (b) shows the images obtained after the perspective correction of the text segments in the real image sequences of Figures 9.4 and 9.5, respectively. Focal length of the camera, given in the instruction manual was used for performing these corrections. In the case of the indoor scene, the correction is somewhat imperfect. The possible reasons could be the large distance of the plane normal through the camera from the field of view, or an error in the focal length. The outdoor scene shows a satisfactory correction of the perspective distortion.

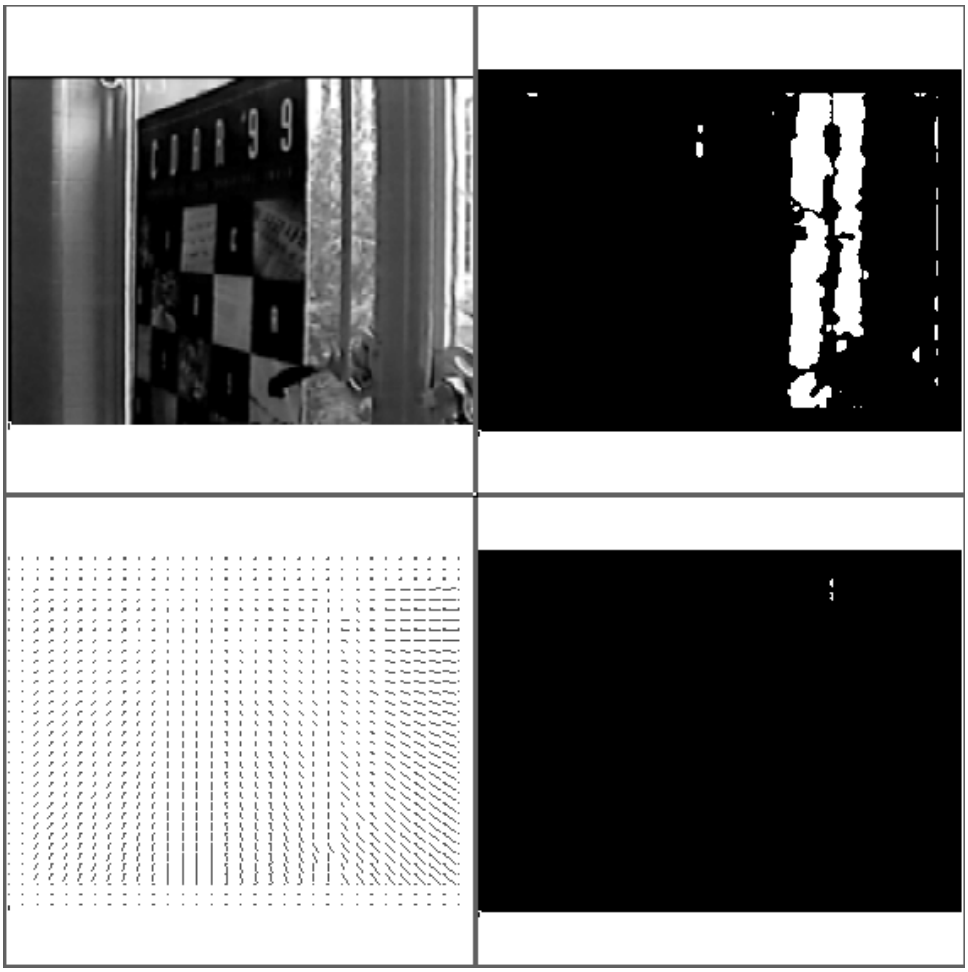
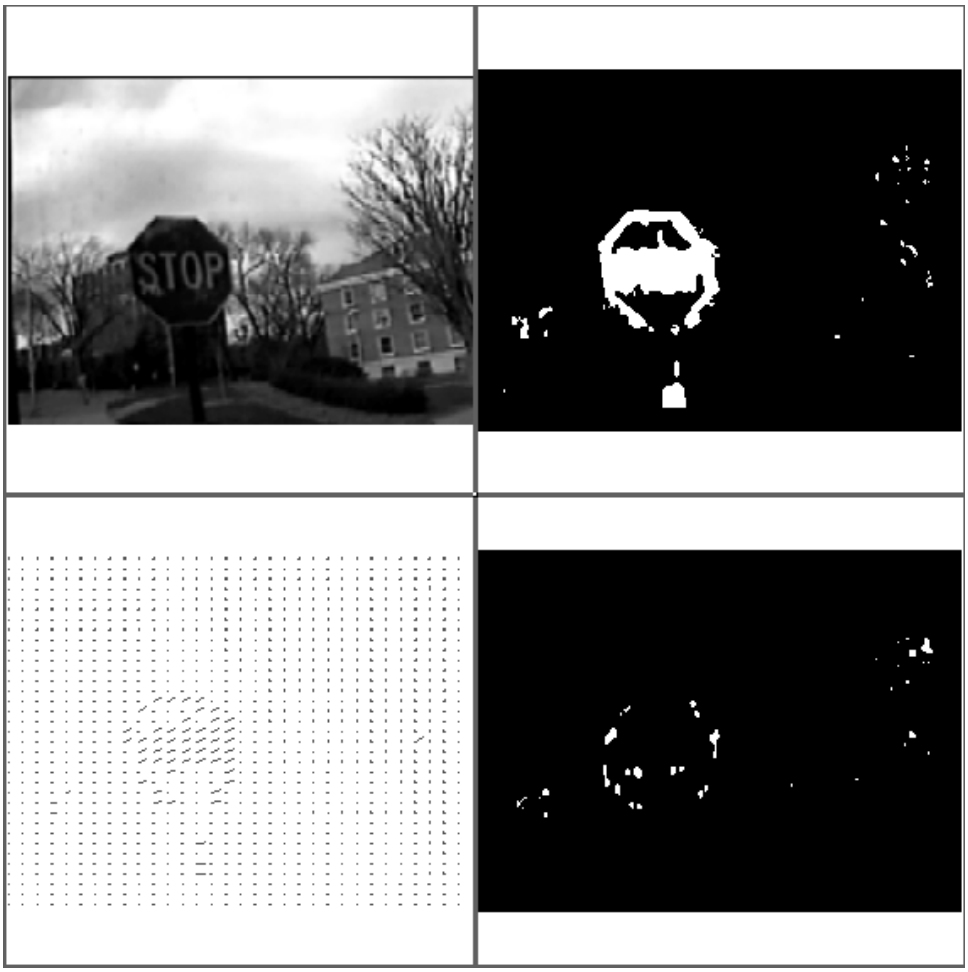


Fig. 9.4. Segmentation of a real image sequence from an indoor scene: (a) Frame 30 in a real image sequence with a poster containing text. (b) Segmentation obtained using the algorithm in 2 steps. (c) Estimates of image motion throughout the image. (d) Detected outliers in each region which could make the parameter estimation unreliable.

(a) (b)



(c) (d)

Fig. 9.5. Segmentation of a real image sequence from an outdoor scene: (a) Frame 40 in a real image sequence containing a "STOP" sign in distant background. (b) Segmentation obtained using the algorithm in 2 steps. (c) Estimates of image motion throughout the image. (d) Detected outliers in each region which could make the parameter estimation unreliable.

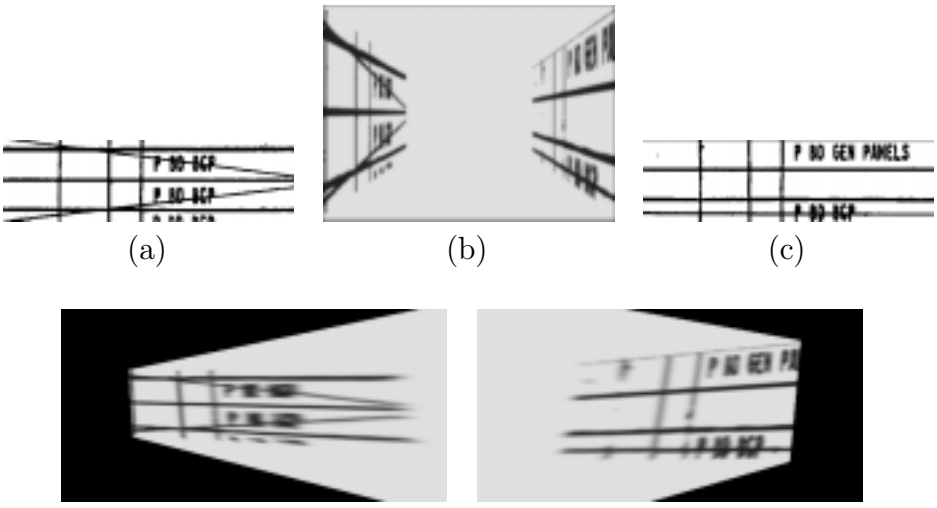
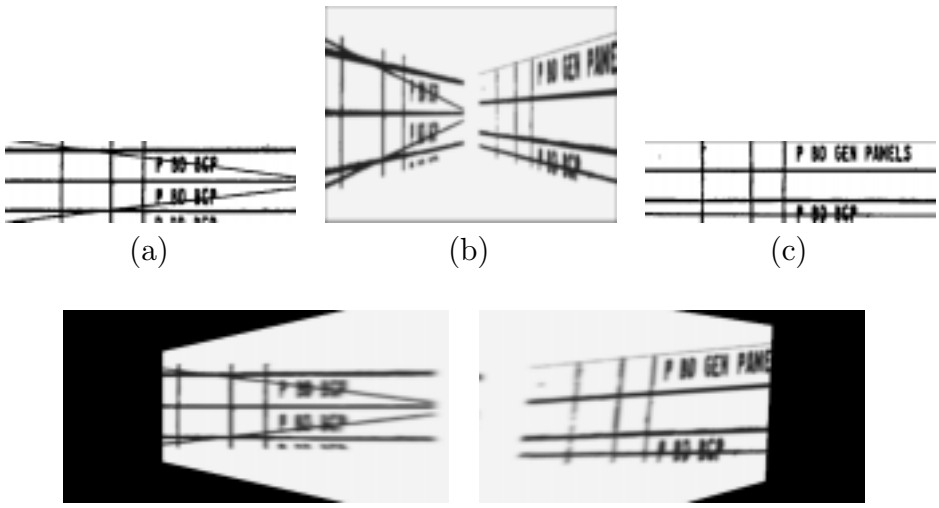


Fig. 9.6. Perspective correction for a simulated image sequence: (a) Original text image for left side. (b) The text images projected on the left and right side of the camera in a typical image from a sequence. (c) Original text image for right side. (d) Corrected image from the left half. (e) Corrected image from the right half.



(a) (b) (c)

(d) (e)

Fig. 9.7. Perspective correction for a simulated image sequence with the projected planes rotated 75° about the camera axis. The perspective correction is somewhat better in this case: (a) Original text image for left side. (b) The text images projected on the left and right side of the camera in a typical image from a sequence. (c) Original text image for right side. (d) Corrected image from the left half. (e) Corrected image from the right half.

(a) (b)

Fig. 9.8. Perspective correction for the text segments of the (a) indoor and (b) outdoor image sequences captured using a camera in motion.

Chapter 10

Sensitivity Analysis of Planar Motion Estimation

Due to the presence of image noise, and the inexact nature of the image motion computation method, the estimated planar motion parameters are expected to have an error which can result in degraded performance of segmentation algorithms. This error is dependent on various conditions, such as the field of view of the camera, orientation of plane normal, and the amount of image motion [2]. Furthermore, the estimates of the plane normal parameter would also be inaccurate. If these parameters are used to compensate the perspective distortion, the compensation would not be complete. It was seen in Chapter 9 that even if the plane normal was accurately determined, the perspective correction was not always satisfactory. Hence, the error propagation from one stage to the other should be studied, and conditions under which the sensitivity to error is large should be identified. This chapter describes the propagation of error in these stages to examine the conditions under which the parameter estimates as well as the perspective distortion compensation are accurate. During the experiments, all the error propagation was done numerically through the covariance matrices.

10.1 Image motion to planar motion parameters

As shown in Chapter 9, if Σ_u is the covariance of the estimated image motion of a pixel, the inverse covariance of the planar motion parameters is given by:

$$\Sigma_a^{-1} = J^t \Sigma_u^{-1} J \tag{10.1}$$

where J is given by equations (9.13) and (9.14). The analytical treatment of the general case involves very long expressions. Hence, the following particular case is considered. It is assumed that the error in the image motion estimates is isotropic, identical and independent of each other (which may not be exactly true). In such a case, $\Sigma_u = \sigma_u^2 I$, and one can write:

$$\Sigma_a^{-1} = \sigma_u^{-2} \sum_i \left[(J^i)^t (J^i) \right] \tag{10.2}$$

If the image motion estimates are taken at relatively small intervals on a grid of resolution $\Delta x_1 \times \Delta x_2$, the summation can be approximated as integration:

$$\Sigma_a^{-1} \simeq \frac{\sigma_u^{-2}}{\Delta x_1 \Delta x_2} \int\int \left[J^t(x_1, x_2) J(x_1, x_2) \right] dx_1 dx_2 \tag{10.3}$$

where and $J(x_1, x_2)$ is 2×8 matrix function corresponding to J^i in the summation expression. For a block of size $c \times d$ in the image, centered at the origin – i.e., $x_1 = -c/2 \ldots c/2$ and $x_2 = -d/2 \ldots d/2$ – the matrix Σ_a is given by:

$$\begin{aligned}
\frac{cd}{\sigma^2 \Delta x_1 \Delta x_2} \Sigma_a &= \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & \frac{c^2}{12} & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & \frac{d^2}{12} \\
0 & 0 & \frac{c^2}{12} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{c^2}{12} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{d^2}{12} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{d^2}{12} & 0 & 0 \\
\frac{c^2}{12} & 0 & 0 & 0 & 0 & 0 & \frac{c^2}{720}(9c^2 + 5d^2) & 0 \\
0 & \frac{d^2}{12} & 0 & 0 & 0 & 0 & 0 & \frac{d^2}{720}(5c^2 + 9d^2)
\end{bmatrix}^{-1} \\
&= \begin{bmatrix}
\frac{9c^2+5d^2}{(4c^2+5d^2)} & 0 & 0 & 0 & 0 & 0 & -\frac{60}{(4c^2+5d^2)} & 0 \\
0 & \frac{5c^2+9d^2}{(5c^2+4d^2)} & 0 & 0 & 0 & 0 & 0 & -\frac{60}{(5c^2+4d^2)} \\
0 & 0 & \frac{12}{c^2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{12}{c^2} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{12}{d^2} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{12}{d^2} & 0 & 0 \\
-\frac{60}{(4c^2+5d^2)} & 0 & 0 & 0 & 0 & 0 & \frac{720}{c^2(4c^2+5d^2)} & 0 \\
0 & -\frac{60}{(5c^2+4d^2)} & 0 & 0 & 0 & 0 & 0 & \frac{720}{d^2(5c^2+4d^2)}
\end{bmatrix}
\end{aligned} \tag{10.4}$$

The variance of each element of $\vec{a}$ is given by the diagonal elements of Σ_a. These are bounded by the following:

$$\mathrm{diag}(\Sigma_a) \leq \frac{\sigma^2 \Delta x_1 \Delta x_2}{cd} \left[\begin{array}{cccccccc} \frac{9}{4} & \frac{9}{4} & \frac{12}{c^2} & \frac{12}{c^2} & \frac{12}{d^2} & \frac{12}{d^2} & \frac{180}{c^2(c^2+d^2)} & \frac{180}{c^2(c^2+d^2)} \end{array} \right] \tag{10.5}$$

From this equation, it can be seen that the accuracy degrades when the dimensions of the planar surface (c, d) are small. The degradation is most severe for the quadratic

parameters (a_7, a_8), followed by the linear $(a_3 \ \ldots \ a_6)$ and constant (a_1, a_2) parameters, respectively.

The above analysis gives an order of magnitude in the accuracy of the planar motion parameters. However, the actual covariance is estimated numerically at the time of estimation, directly from the image gradients as described in Chapter 9.

10.2 Planar motion parameters to plane normal vector

The expressions for covariance of planar motion parameters K, V, and W were derived in Chapter 9 for multiple planar surfaces, with single planar surface being a special case. However, when perspective correction is performed, the plane normal vector K is normalized to give a unit vector k, whose sensitivity can be different from that of K. To keep the relationship between $\vec{a}$ and the new parameters same as before, V is also scaled, but W is kept the same. The new parameters are grouped as:

$$\vec{p} = \begin{bmatrix} k \\ w \\ v \end{bmatrix} = \begin{bmatrix} K/\|K\| \\ W \\ V\|K\| \end{bmatrix} \tag{10.6}$$

and the plane parameters expressed as:

$$\vec{a} = \tilde{f}(\vec{p}) \tag{10.7}$$

For small changes in $\vec{p}$, we have:

$$\tilde{F}\Delta\vec{p} \simeq \Delta\vec{a} \tag{10.8}$$

where $\tilde{F}$ is the Jacobian of $\vec{a}$ w.r.t. $\vec{p}$, which is an 8×9 matrix and therefore not invertible. However, by using the constraint:

$$\tilde{c}(\vec{p}) = \frac{1}{2}\left[(k_0)^2 + (k_1)^2 + (k_2)^2 - 1\right] = 0 \tag{10.9}$$

its Jacobian denoted by $\tilde{C}$ satisfies:

$$\tilde{C}\Delta\vec{p} \simeq \Delta\tilde{c} = 0 \tag{10.10}$$

Hence, one can write:

$$\begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix} \Delta\vec{p} \simeq \begin{bmatrix} \Delta\vec{a} \\ \Delta\tilde{c} \end{bmatrix} = \begin{bmatrix} \Delta\vec{a} \\ 0 \end{bmatrix} \tag{10.11}$$

The above equation can be inverted to give the change in $\vec{p}$ for a small change in $\vec{a}$ as follows:

$$\Delta\vec{p} \simeq \begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix}^{-1} \begin{bmatrix} \Delta\vec{a} \\ 0 \end{bmatrix} \tag{10.12}$$

where

$$\begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix} = \begin{bmatrix} -v_1 & 0 & 0 & 0 & 0 & -1 & 0 & -k_0 & 0 \\ -v_2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -k_0 \\ v_0 & -v_1 & 0 & 0 & 0 & 0 & k_0 & -k_1 & 0 \\ 0 & -v_2 & 0 & -1 & 0 & 0 & 0 & 0 & -k_1 \\ 0 & 0 & -v_1 & 1 & 0 & 0 & 0 & -k_2 & 0 \\ v_0 & 0 & -v_2 & 0 & 0 & 0 & k_0 & 0 & -k_2 \\ 0 & v_0 & 0 & 0 & 0 & -1 & k_1 & 0 & 0 \\ 0 & 0 & v_0 & 0 & 1 & 0 & k_2 & 0 & 0 \\ k_0 & k_1 & k_2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10.13}$$

The determinant of the above matrix can be shown to be equal to:

$$\|v \times k\|^2 = (v_1k_2 - v_2k_1)^2 + (v_2k_3 - v_3k_2)^2 + (v_3k_1 - v_1k_3)^2 \tag{10.14}$$

If v and k are nearly parallel the determinant is very small, and the estimate of k is highly sensitive. Furthermore, it can be also shown that the elements of rows 1 to 3 and columns 1 to 8 which determine k from $\vec{a}$ are bounded by:

$$\begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix}_{ij} \leq \frac{3\|v\|}{\|v \times k\|^2} = \frac{3}{\|K\|\|k \times V\|^2} \tag{10.15}$$

where k and V are both unit vectors. It can be seen that the sensitivity of the estimate is inversely proportional to $\|K\|$ and therefore decreases as the planar surface comes nearer so that $\|K\|$ becomes larger. Also, increase in camera velocity v is equivalent to bringing the planar surface nearer by the same factor, and therefore decreases the sensitivity of the estimate.

Also, if $\Sigma_a = E[\Delta\vec{a}\Delta\vec{a}^t]$ is the covariance of $\vec{a}$, the covariance of $\vec{p}$ is given by:

$$\Sigma_p = E[\Delta\vec{p}\Delta\vec{p}^t] = \begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix}^{-1} \begin{bmatrix} \Sigma_a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{F} \\ \tilde{C} \end{bmatrix}^{-t} \tag{10.16}$$

If multiple planar surfaces are used, and the results are integrated, it can be expected that the surfaces which have lower sensitivity would dominate the computation. Hence, even if one of the surfaces is not nearly normal to the velocity vector, a good solution can be expected.

10.3 Plane normal vector to perspective correction

Suppose that after applying perspective correction by rotating the coordinate system using the unit normal vector k, the residual error is Δk. This error Δk is equivalent to rotating the perfectly corrected image around the axis parallel to Δk by angle equal to its magnitude. If (t_1, t_2) are the coordinates of the actual planar surface, and the coordinates on the corrected surface (t_1', t_2') are given by rotating the homogeneous coordinates by $R(\Delta k)$ as:

$$\begin{bmatrix} \alpha \\ t_1' \\ t_2' \end{bmatrix} \propto R(\Delta k) \begin{bmatrix} \alpha \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} R_{00}\alpha + R_{01}t_1 + R_{02}t_2 \\ R_{10}\alpha + R_{11}t_1 + R_{12}t_2 \\ R_{20}\alpha + R_{21}t_1 + R_{22}t_2 \end{bmatrix} \tag{10.17}$$

where α is the zooming factor. For small Δk, $R(\Delta k)$ is given by:

$$R(\Delta k) = \begin{bmatrix} 1 & -\Delta k_2 & \Delta k_1 \\ \Delta k_2 & 1 & -\Delta k_0 \\ -\Delta k_1 & \Delta k_0 & 1 \end{bmatrix} \tag{10.18}$$

It should be noted that since $\|k\| = 1$, there are only two degrees of freedom in Δk, which approximately satisfies:

$$k^t \Delta k = k_0\Delta k_0 + k_1\Delta k_1 + k_2\Delta k_2 \simeq 0 \tag{10.19}$$

A square grid of the perfect projection would be distorted as shown in Figure 10.1 (a). The two vanishing points, obtained by letting t_1 and t_2 respectively approach infinity,

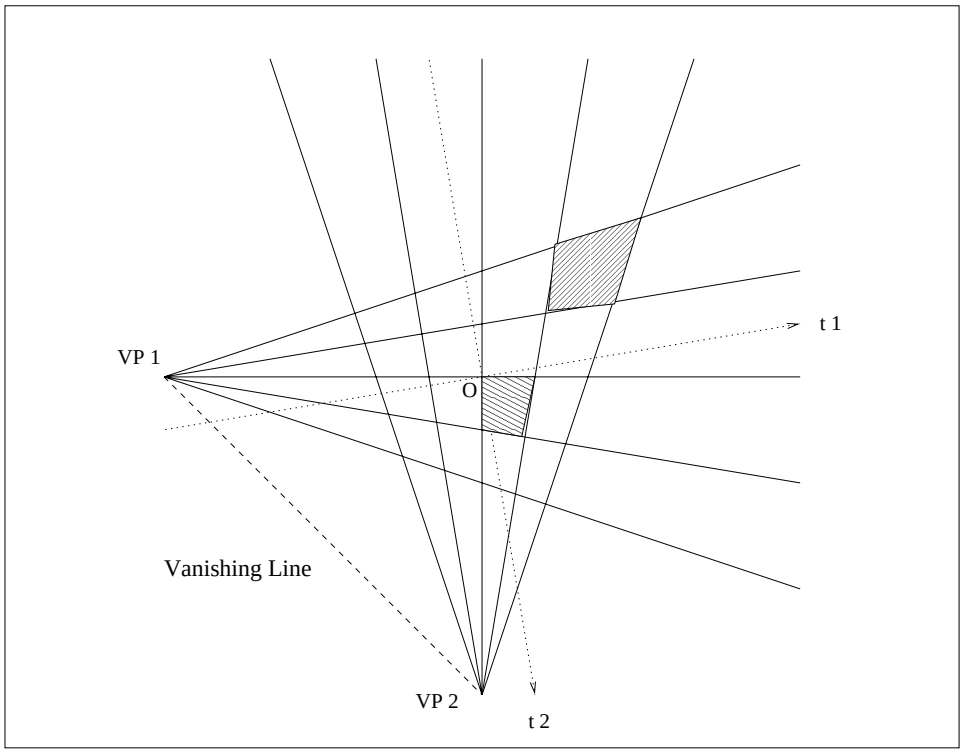
are given by:

$$\begin{aligned} VP_1 &= \frac{\alpha}{R_{01}} \begin{bmatrix} R_{11} \\ R_{21} \end{bmatrix} \simeq \frac{-\alpha}{\Delta k_2} \begin{bmatrix} 1 \\ \Delta k_0 \end{bmatrix} \\ VP_2 &= \frac{\alpha}{R_{02}} \begin{bmatrix} R_{12} \\ R_{22} \end{bmatrix} \simeq \frac{\alpha}{\Delta k_1} \begin{bmatrix} -\Delta k_0 \\ 1 \end{bmatrix} \end{aligned} \tag{10.20}$$

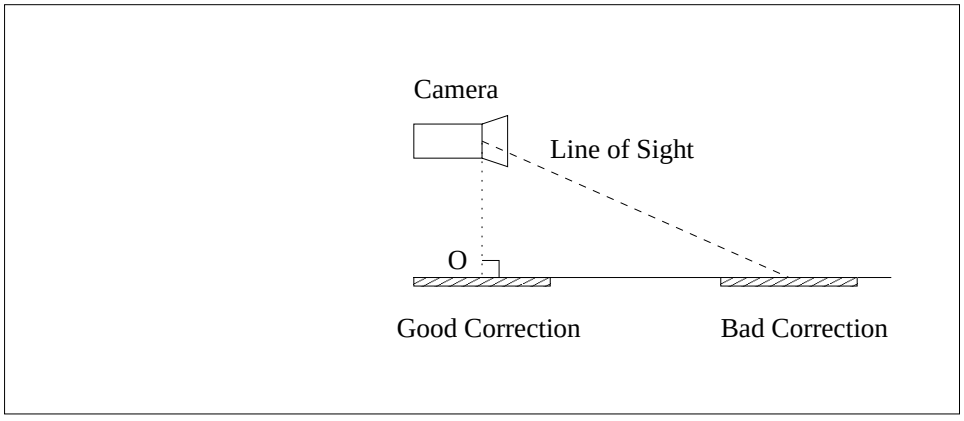
If $\Delta k_0 = 0$, the vanishing points lie on the t_1 and t_2 axes respectively. Otherwise, they get rotated by an angle of $\tan^{-1} \Delta k_0$. The vanishing points are joined by the vanishing line representing all the points lying at infinity in the $t_1 - t_2$ system. It can be shown that the equation of the vanishing line is given by:

$$\alpha R_{00} + R_{10} t_1 + R_{20} t_2 = 0 \tag{10.21}$$

The distances of the vanishing points from the planar origin is approximately inversely proportional to the errors in the components of Δk. The origin of the plane is the point on the plane from which the normal to the plane passes through the camera center. It can be seen that if the imaged area is very far from the planar origin, i.e., $t_1 \gg \alpha$ or $t_2 \gg \alpha$, the perspective projection is inaccurate. This happens especially for a planar surface whose normal is perpendicular to the camera axis. In such a case, the field of view of the camera usually lies far from the planar origin as shown in Figure 10.1 (b). The correction in such areas is therefore unsatisfactory.



(a)



(b)

Fig. 10.1. Sensitivity of perspective correction: (a) The distortion in perspective correction of a square grid due to an error in estimating the plane normal parameter. The origin (O) is actually the point on the plane from which the normal to the plane passes through the camera center. (b) Positions where the perspective distortion is expected to be satisfactory or unsatisfactory for a particular camera and scene configuration.

Chapter 11

Conclusion

This research was focused on designing and implementing algorithms for detection of obstacles in the flight path of the aircraft using the image sequences obtained from the on board cameras. Some of the principles used for obstacle detection were applied to the application of separating scene text from image sequences. The main contributions of this research and the possible avenues for future work are described below.

11.1 Contributions of this research

- Basic algorithms performing signal enhancement were tested for detecting flying objects using the image sequences provided by NASA. Performance characterization of these algorithms was conducted using simulated and real image sequences. It was observed that the algorithms performed well on images with little or no clutter, but their performance degraded in presence of clutter.

- To distinguish the objects on a collision course from the background clutter, the difference in the behavior of their image translation and expansion were studied. Conditions under which these criteria are useful were derived. Novel methods for estimating the rates of image translation and expansion over long image sequences were designed and tested on the image sequence with a large amount of background clutter. The approach successfully separated the obstacle from the clutter.

- Algorithm fusion to overcome limitations of algorithms was studied, and it was observed that under proper conditions, a combination of algorithms performed better than the individual algorithms.

- A real-time system using pipelined image-processing hardware was designed to detect objects crossing the aircraft. The tracking algorithm to separate background clutter from crossing objects was developed and implemented on the host machine associated with the system.

- Principles used for obstacle detection on runway were applied to separation of scene text from image sequences with relative motion between the scene and the camera. Since scene text usually occurs on planar surfaces, a motion segmentation algorithm that was developed previously for runway obstacle detection was used for separating planar surfaces in the scene. Parameters of the planar surfaces were obtained using the planar motion model for each surface, and these parameters were used for correction of perspective distortion in scene text. A novel method for combining the structure and motion estimates from multiple planar surfaces was developed to improve the robustness of the estimates. Sensitivity analysis of various steps in this procedure was also performed.

11.2 Future work

- Many of the research ideas, such as the use of translation and expansion, algorithm fusion, etc. were tested individually. The future goal would be to combine these into an integrated system for obstacle detection. Performance characterization of this system could be done with more real image sequences.

- During the estimation of image translation to discriminate a hazardous object from background clutter, the compensation of aircraft rotation was performed using the navigation system information. Use of background clutter to model the aircraft motion could be explored, so that the compensation could be performed even without the navigation system information.

- False expansion occurring due to the rotation of the target aircraft can be studied. This expansion takes place only in a particular direction, resulting in deformation of the object in the image. Methods to distinguish this deformation from uniform expansion can be studied.

- Gaussian models were used for studying the behavior of individual algorithms to perform algorithm fusion. Better models could be developed, especially in presence of clutter, where the Gaussian models would not be as robust.

- To improve the performance of crossing object detection, the image motion due to aircraft rotation should be compensated. This could be done either using the navigation data from the aircraft, or by modeling the image motion separate independent object motion. Since the DataCube architecture is capable only of simple

image processing operations, such a procedure should be performed on the host machine, using a feature based approach.

- Segmentation algorithm used for scene text segmentation is interactive at present. The next step would be to automate the algorithm, test it on more real image sequences, and optimize the parameters of the algorithm. Estimation of the equations of planar surfaces, and perspective correction using these parameters was performed separately. These could be combined to form an integrated system for scene text detection.

- The planar motion parameters obtained from the motion segmentation procedure for scene text extraction can be used to register multiple image frames. The redundancy of information in these images can then be used to generate a single image with a higher resolution than the original images. This process, known as super-resolution could improve the accuracy of Optical Character Recognition (OCR) on the scene text.

References

[1] G. Adiv. Determining three-dimensional motion and structure from optical flow generated by several moving objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(4):384–401, 1985.

[2] G. Adiv. Inherent ambiguities in recovering 3-D motion and structure from a noisy flow field. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):477–489, 1989.

[3] N. Ancona and T. Poggio. Optical flow from 1-D correlation: Application to a simple time-to-crash detector. *International Journal of Computer Vision*, 14:131–146, 1995.

[4] J. Arnold, S. Shaw, and H. Pasternack. Efficient target tracking using dynamic programming. *IEEE Trans. on Aerospace and Electronic Systems*, 29(1):44–56, January 1993.

[5] S. Atiya and G. D. Hager. Real-time vision-based robot localization. *IEEE Transaction on Robotics and Automation*, 9(6):785–800, December 1993.

[6] H. H. Baker and R. C. Boles. Generalizing epipolar-plane image analysis on the spatiotemporal surface. *International Journal of Computer Vision*, 3(1):33–49, 1989.

[7] Y. Baram and Y. Barniv. Obstacle detection by recognizing binary expansion patterns. *IEEE Trans. on Aerospace and Electronic Systems*, 32(1):191–197, January 1996.

[8] Y. Barniv. Dynamic programming solution for detecting dim moving targets. *IEEE Trans. on Aerospace and Electronic Systems*, 21(1):144–156, January 1985.

[9] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Higorani. Hierarchical model-based motion estimation. In *Proc. of European Conference on Computer Vision*, pages 237–252, 1992.

[10] J. S. Bird and M. M. Goulding. Rate-constrained target detection. *IEEE Trans. on Aerospace and Electronic Systems*, 28(2):491–503, April 1992.

[11] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[12] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7–56, 1987.

[13] G. D. Borshukov, G. Bozdagi, Y. Altunbasak, and A. M. Tekalp. Motion segmentation by multistage affine classification. *IEEE Trans. on Image Processing*, 6(11):1591–1594, November 1997.

[14] P. Bouthemy and E. Francois. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *International Journal of Computer Vision*, 10(2):157–182, 1993.

[15] P. J. Burt. Fast filter transforms for image processing. *Computer Graphics and Image Processing*, 16(1):20–51, 1981.

[16] S. Carlsson. Object detection using model-based prediction and motion parallax. In *Proc. of European Conference on Computer Vision*, pages 297–306, 1990.

[17] D. Casasent and A. Ye. Detection filters and algorithm fusion for ATR. *IEEE Trans. on Image Processing*, 6(1):114–125, January 1997.

[18] P. Comelli, P. Ferragina, M. N. Granieri, and F. Stabile. Optical recognition of motor vehicle license plates. *IEEE Trans. on Vehicular Technology*, 44(4):790–799, November 1995.

[19] Y. Cui and Q. Huang. Character extraction of license plates from video. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 502–507, 1997.

[20] S. Devadiga. *Detection of Obstacles in Monocular Image Sequences*. PhD thesis, The Pennsylvania State University, August 1997.

[21] E. D. Dickmanns and F. R. Schell. Autonomous landing of airplanes by dynamic vision. In *Proc. of IEEE Workshop on Applications of Computer Vision*, pages 172–179, December 1992.

[22] E. Francois and P. Bouthemy. Derivation of qualitative information in motion analysis. *Image and Vision Computing*, 8(4):279–288, November 1990.

[23] T. Gandhi, S. Devadiga, R. Kasturi, and O. Camps. Detection of obstacles on runway by ego-motion compensation and tracking of significant features. In *Proc. of IEEE Workshop on Applications of Computer Vision*, pages 168–173, 1996.

[24] S. M. Haynes and R. Jain. Detection of moving edges. *Computer Vision, Graphics and Image Processing*, 21(3):345–367, 1983.

[25] G. C. Holst. *CCD Arrays, Cameras and Displays*. JCD Publishing, Winter Park, FL, 1996.

[26] B. K. P. Horn. *Robot Vision*. The MIT Electrical Engineering and Computer Science Series. The MIT Press, Cambridge, MA, 1986.

[27] A. Huertas, W. Cole, and R. Nevetia. Detecting runways in complex airport scenes. *Computer Vision Graphics and Image Processing*, 51(2):107–145, 1990.

[28] M. Irani and P. Anandan. A unified approach to moving object detection in 2D and 3D scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(6):577–589, June 1998.

[29] M. Irani and S. Peleg. Improving resolution by image registration. *CVGIP: Graphical Models and Image Processing*, 53(3):231–239, May 1991.

[30] R. Jain, D. Militzer, and H.-H. Nagel. Separating non-stationary scene components in a sequence of real world TV images. In *Proc. of International Joint Conference on Artificial Intelligence*, pages 425–428, 1977.

[31] R. Jain and H.-H. Nagel. On the analysis of accumulative difference pictures from image sequences of real world scenes. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 1(2):206–214, 1979.

[32] K. Kanatani. *Statistical Optimization for Geometric Computation: Theory and Practice*. Elsevier Science, Amsterdam, The Netherlands, 1996.

[33] T. Kanungo, M. Y. Jaisimha, J. Palmer, and R. M. Haralick. A methodology for quantitative performance evaluation of detection algorithms. *IEEE Trans. on Image Processing*, 4(12):1667–1674, December 1995.

[34] R. Kasturi, O. Camps, L. Coraor, K. Hartman, T. Gandhi, and M.-T. Yang. Performance characterization of target detection algorithms for aircraft navigation. Technical report, Dept. of Computer Science and Engineering, The Pennsylvania State University, October 1998.

[35] R. Kasturi, O. Camps, T. Gandhi, and S. Devadiga. Detection of obstacles on runway using ego-motion compensation and tracking of significant features. Technical Report CSE-96-045, Dept. of Computer Science and Engineering, The Pennsylvania State University, June 1996.

[36] R. Kasturi, Y.-L. Tang, and S. Devadiga. A model-based approach for detection of runways and other objects in image sequences acquired using an on-board camera. Technical Report CSE-94-051, Department of Computer Science and Engineering, Penn State University, August 1994.

[37] S. M. Kay. *Fundamentals of Statistical Signal Processing, Volume II: Detection Theory.* Prentice Hall, Upper Saddle River, NJ, 1993.

[38] S. S. Krause. *Avoiding Mid-Air Collisions.* TAB books, Mc Graw Hill Inc., Blue Ridge Summit, PA, 1995.

[39] D. J. Kriegman, E. Trendel, and T. O. Binford. Stereo vision and navigation in buildings for mobile robots. *IEEE Transaction on Robotics and Automation*, 5(6):792–803, December 1989.

[40] Y. Liu, T. S. Huang, and O. D. Faugeras. Determination of camera location from 2-D to 3-D line and point correspondences. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12(1):28–37, January 1990.

[41] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of DARPA Image Understanding Workshop*, pages 121–130, 1981.

[42] S. Mann and R. W. Picard. Virtual bellows: Constructing high quality stills from video. In *Proc. IEEE International Conference on Image Processing*, pages I:363–367, November 1994.

[43] G. Medioni and R. Nevetia. Matching images using linear features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):675–685, 1984.

[44] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(8):858–867, August 1997.

[45] R. C. Nelson. Qualitative detection of motion by a moving observer. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 173–178, 1991.

[46] R. C. Nelson and J. Y. Aloimonos. Obstacle avoidance using flow field divergence. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(10):1102–1106, 1989.

[47] K. Nishiguchi, M. Kobayashi, and A. Ichikawa. Small target detection from image sequences using recursive max filter. In *Proc. SPIE*, volume 2561, pages 153–166, July 1995.

[48] J. Ohya, A. Shio, and S. Akamatsu. Recognizing characters in scene images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16:214–220, February 1994.

[49] H. V. Poor. *An Introduction to Signal Detection and Estimation*. Springer-Verlag, New York, NY, 2nd edition, 1994.

[50] A. Rosenfeld, editor. *Multiresolution Image Processing and Analysis*, chapter The Pyramid as a Structure for Efficient Computation. Springer-Verlag, 1984.

[51] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger. Probability distributions of optical flow. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pages 310–315, 1991.

[52] P. N. Smith, B. Sridhar, and R. E. Suorsa. Multiple-camera/motion stereoscopy for range estimation in helicopter flight. In *Proc. of American Control Conference*, pages 1339–1343, June 1993.

[53] S. M. Smith and J. M. Brady. SUSAN - a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997.

[54] S. Snyder, B. Schipper, L. Vallot, N. Parker, and G. Spitzer. Differential GPS/inertial navigation approach/landing flight test results. *IEEE PLANS*, March 1992.

[55] B. Sridhar, R. Suorsa, and B. Hussien. Passive range estimation for rotor-craft low-altitude flight. *Machine Vision and Applications*, 6(1):10–24, 1993.

[56] S. Sull and B. Sridhar. Runway obstacle detection by controlled spatiotemporal image flow disparity. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 385–390, June 1996.

[57] Y. L. Tang. *An Airborne Vision System for Runway Recognition and Obstacle Detection.* PhD thesis, The Pennsylvania State University, August 1994.

[58] Y. L. Tang and R. Kasturi. Tracking moving objects during low altitude flight. *Journal of Machine Vision and Applications*, 9(1):20–31, 1996.

[59] W. B. Thompson and T. C. Pong. Detecting moving objects. *International Journal of Computer Vision*, 4(1):39–58, 1990.

[60] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[61] S. M. Tonissen and R. J. Evans. Performance of dynamic programming techniques for track-before-detect. *IEEE Trans. on Aerospace and Electronic Systems*, 32(4):1440–1451, October 1996.

[62] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Trans. on Image Processing*, 3(5):625–638, September 1994.

[63] L. L. Winger, M. E. Jernigan, and J. A. Robinson. Character segmentation and thresholding in low-contrast scene images. In *Proc. SPIE*, volume 2660, pages 286–296, 1996.

[64] D. Wood. *Jane's World Aircraft Recognition Handbook.* Jane's Information Group Ltd., Coulsdon, UK, 1992.

Vita

Tarak Gandhi was born on May 19, 1970 in Mumbai (Bombay), India. He received his Bachelor of Technology degree in Computer Science and Engineering at the Indian Institute of Technology, Bombay, India in 1991. In 1993, he joined the Pennsylvania State University, University Park, Pennsylvania, and recieved his Master of Science degree in Computer Engineering in 1995.

Tarak Gandhi's interests incllude image processing, computer vision, motion analysis, target detection, and pattern recognition.