Algorithms for Detection of Objects in Image Sequences Captured from an Airborne Imaging System

R. Kasturi, O. Camps, Y-L. Tang, S. Devadiga, T. Gandhi

Department of Computer Science and Engineering
Technical Report
CSE-95-026

October 10, 1995

PENNSTATE

1855

Algorithms for Detection of Objects in Image Sequences Captured from an Airborne Imaging System

R. Kasturi, O. Camps, Y-L. Tang, S. Devadiga, T. Gandhi

Department of Computer Science and Engineering
Technical Report
CSE-95-026

October 10, 1995

PENNSTATE

Algorithms for Detection of Objects in Image Sequences Captured from an Airborne Imaging System

Final Technical Report for NASA Grant NCC 2-853
"Algorithms for Detection of Objects in Image Sequences Captured from an Airborne Imaging System"
Period of the Grant: March 1, 1994 to August 31, 1995.

Submitted to
NASA Ames Research Center
Technical Officer: Banavar Sridhar
Grant Officer: Beatrice Morales

by

Rangachar Kasturi
Principal Investigator
Professor of Computer Science and Engineering and Electrical Engineering
Tel: (814)863-4254 Fax: (814)865-3176
E-Mail: kasturi@cse.psu.edu

Octavia Camps
Co-Investigator
Assistant Professor of Electrical Engineering and Computer Science and Engineering
Tel: (814)863-1267 Fax: (814)865-7065
E-Mail: camps@whale.ece.psu.edu

Graduate Students:
Yuan-Liang Tang
Sadashiva Devadiga
Tarak Gandhi

Department of Computer Science and Engineering
The Pennsylvania State University
220 Pond Laboratory
University Park, PA 16802-6106

Algorithms for Detection of Objects in Image Sequences Captured from an Airborne Imaging System

Abstract

This research was initiated as a part of the effort at the NASA Ames Research Center to design a computer vision based system that can enhance the safety of navigation by aiding the pilots in detecting various obstacles on the runway during critical section of the flight such as a landing maneuver. The primary goal of the research described in this report is the development of algorithms for detection of moving objects from a sequence of images obtained from an on-board video camera.

Image regions corresponding to the independently moving objects are segmented from the background by applying constraint filtering on the optical flow computed from the initial few frames of the sequence. These detected regions are tracked over subsequent frames using a model based tracking algorithm. Position and velocity of the moving objects in the world coordinate is estimated using an extended Kalman filter. The algorithms are tested using the NASA line image sequence with six static trucks and a simulated moving truck and experimental results are described. Various limitations of the currently implemented version of the above algorithm are identified and possible solutions to build a practical working system are investigated.

Table of Contents

List of Publications and Reports

1. Tang Y. L., "An Airborne Vision System for Runway Recognition and Obstacle Detection", Ph. D. Thesis, Department of Computer Science and Engineering, Penn State University, December 1994.

2. Tang Y. L. and R. Kasturi, "Tracking Moving Objects during Low Altitude Flight", to appear in Machine Vision and Applications.

3. Tang Y. L., R. Kasturi, "An airborne Vision System for Runway Recognition and Obstacle Detection", Technical Report CSE 94-045, Department of Computer Science and Engineering, Penn State University, August 1994.

4. Kasturi R., O. Camps, Y. L. Tang and S. Devadiga, "An Airborne Vision System for Tracking Moving Object", Technical Report CSE 94-052, Department of Computer Science and Engineering, Penn State University, August 1994.

1. Introduction

Because of the heavy workload demands that are imposed upon the pilots and crew during low-altitude flight, there is a significant need for an automatic obstacle detection system on-board an aircraft or a rotorcraft. Such a system can relieve the pilots from tiring, monotonous flight control tasks so that they can concentrate more on flight planning. In addition, despite the enforced regulations on the movement of vehicles and aircraft on the ground, runway incursion is still a serious problem which jeopardizes the safety of aircraft landing. NASA with the help of other institutions has been developing various vision algorithms for obstacle detection in an image sequence captured from on-board camera [49, 50, 53-61]. The objective of the intended research is to design a system that can detect stationary and moving objects on the flight path, estimate the range to the stationary objects and estimate the position and the velocity of moving objects [49, 50, 53-61].

In autonomous navigation, it is essential to obtain a three-dimensional description of the static environment in which the vehicle is traveling. For rotorcraft conducting low-altitude flight, this description is particularly useful to detect and avoid obstacles in the intended flight path. This technique is generally referred to as *structure from motion* [31, 52, 63, 64, 67], where the 3D structure of the static scene is estimated, using more than two image frames captured at different camera locations and/or orientations. Many approaches have used line or point correspondences among two to four images to compute the camera motion and the scene structure [30, 33, 35]. Results showed that the solutions from these methods are very noisy. Other approaches overcome this problem by integrating information from a long sequence of images and Kalman filtering is a common choice to obtain a smoothed estimate. The Kalman filtering technique is popular because of its elegant way of handling uncertainty and providing incremental processing. Broida and Chellappa [9] used Kalman filtering as a recursive means to estimate object motion parameters.

Matthies et. al. [34] built a framework which gives depth estimates for every pixel in the image. In their experiments, the side-viewing camera is assumed and the camera motion is only translational in the vertical direction. Under such conditions, feature tracks will follow the vertical image scan lines, and feature matching becomes much simpler. Several approaches for implementing passive range estimation have been investigated at NASA Ames [53]. In [50, 54] Sridhar et. al. have described feature based range estimation algorithm for recursive estimation of range using a Kalman filter. Their algorithm uses monocular sequence of images, along with the knowledge of the camera's motion. Results from helicopter flight experiment were presented. The above algorithm was also implemented on various parallel machines and their performance was evaluated [55].

Bolles et. al. [7] used *Epipolar Plane Image (EPI) Analysis* for static scene analysis. In their approach, the camera is moving only laterally and the camera viewing direction is always orthogonal to the motion path. In such a simple case, the feature motion analysis becomes merely a line fitting process on the EPI, and the 3D location of the tracked feature can be determined by the parameters of the fitted line. In the case of forward linear camera motion, however, the feature tracks will be hyperbolas and curve fitting becomes necessary. Sawhney et al. [38] reported that curve fitting is much more difficult and noisy, making this approach less robust. In another paper by Baker and Bolles [3], the Kalman filtering technique is used to estimate the range in the case of forward linear camera motion. Also dealt with was the nonlinear camera motion case where the nonlinear motion path is restricted to be on a horizontal plane and where only *one* of the EPIs in the spatiotemporal data, parallel to the motion plane, can be analyzed. This restriction makes their approach less useful in practice. In [57, 58] EPI analysis was modified for general 3D camera motion by assuming that camera motion is piecewise linear and results for the NASA line and arc sequence were presented.

In the course of navigation, the robot or the vehicle has to estimate the range from an obstacle to the camera in order to avoid it by changing its nominal path. Several methods for range estimation have been investigated at NASA Ames Research Center [5, 11, 42, 50, 53, 54]. Their main approach is also to use Kalman filtering to recursively refine the estimated range. The range estimation procedure described in [57, 58] uses a simple incremental weighted least squares method for stationary object position estimation using known camera state parameters.

Obstacle detection in case of a moving camera, moving object situation has been an active research topic for years. Speorri and Ullman [51] describes methods requiring only local computations for detecting motion boundaries in a scene containing several objects without prior knowledge of their shapes and motions. Meygret and Thonnat [36] computes the optical flow associated with each contour chain points and groups the chain points based on the spatial proximity and coherency of the apparent movement. Thompson and Pong [62] and Nelson [38] detects moving objects on the basis of simple flow clustering or inconsistency with the background flow. Adiv [1] segments the optical flow based on a fit to an affine model. Adiv further groups the resulting regions to fit a model of a planar surface undergoing 3D motions in perspective projection. Debrunner et. al. [14], given dense temporal sequence of intensity image of multiple moving objects, will separate images into regions showing distinct objects by segmenting the trajectories into subsets corresponding to different objects with the determination of motion and structure of the objects. Balck [6] describes an approach that formulates a model of surface patches in terms of constraints on intensity and motion while accounting for discontinuities. An incremental minimization scheme is used to segment the scene over a sequence of images. Bouthemy and Lalande [8] describes a framework based on a statistical model namely the spatiotemporal morkov fields to detect moving objects when the camera is static. Rognone et al. [43] finds homogeneous regions by analyzing local linear approximations of optical flow over

patches of the image plane, which determine a list of possibly viewed motions, and, finally by applying a technique of stochastic relaxation.

In Section 2 of this report, we describe an optical flow based approach for detecting independently moving objects from a sequence of monocular images. Knowledge of the camera motion is used to detect image regions violating the flow constraint in the initial frame. Model based object tracking method is used to track these regions in the subsequent frames. Position and velocity of these objects in the world coordinate system is computed using an extended Kalman filter. Experimental results for the NASA line image sequence with a simulated truck is presented. In section 3, various limitations of the currently implemented version of the above algorithm are identified and proposed enhancements to the algorithm to build a practical working system are described.

2. Moving Object Detection and Estimation

In order to maintain flight safety, the vision system should be capable of tracking both the stationary and the moving objects. Detecting and tracking moving objects, present a challenging problem because of the following factors. First, since the objects are moving in the scene, their 2D projections on the images do not follow the epipolar constraint unlike stationary objects. Hence, feature matching is more difficult. Second, the motion dynamics of the object may not be constant during the process of tracking. It is thus difficult to model and estimate object motion. Third, fast moving objects will be difficult to track, while slowly moving objects may not be detected at all.

It is well known that moving object detection using visual information alone is quite difficult, particularly when the camera is also moving. In such a case, even stationary objects generate apparent optical flow in the image. In addition, because distinct motion scenarios among the environment, objects, and the camera may generate very similar optical flow fields in the images

[62], interpretation of object motion can be quite ambiguous given only the optical flow information. In our application, the camera motion parameters are constantly available from the Inertial Navigation System (INS) onboard the helicopter. Such information is useful to resolve the ambiguity because the image velocities due to camera motion can be factored out, leaving only the part that is contributed by independently moving objects. A method called the *constraint region filtering* is thus developed to detect moving objects by using the known camera motion information. It is a modification of Nelson's algorithm [38]. After detection of moving objects, a method based on extracting object models directly from the image is used to track moving objects. An advantage of this method is that no a priori knowledge of the object models is required. Furthermore, each object model is updated from one frame to the next in order to pick up the 2D shape change resulting from relative motion between the camera and the object. Updating the object model makes the tracking process more reliable. Using the tracking results, Kalman filtering is finally performed to estimate the object motion parameters.

2.1 Motion Detection

The ability to visually detect moving objects is important in a wide variety of circumstances, such as traffic control, military, remote surveillance of industrial areas, biomedical studies, target tracking, etc. In the case that the camera is known to be stationary and the lighting is well controlled, motion detection is simple — just by temporal differencing the image frames, i.e., obtaining the intensity differences between one image and the next [20, 25, 26]. The remaining tasks are thus to deal with noise, to distinguish between occlusion and disocclusion, and to infer object motion velocity if necessary [8, 15, 19, 44]. However, the problem becomes significantly more difficult if the camera is also moving since now the task is to detect objects moving with respect to the environment and not the camera. In such a case, differencing is of little value as all

visible surfaces are likely to be moving with respect to the camera in a manner that will generate noticeable changes throughout the image. To overcome this problem, we make use of the information about the camera motion provided by the INS. The optical flow in the image is first computed using the algorithm developed by Lucas and Kanade [32]. The camera motion information is then used in the constraint region filtering to extract optical flow which may be due to independently moving objects. As the algorithm for optical flow computation is generally very susceptible to noise, some flow are actually due to noise. A segmentation method based on testing the spatial and temporal consistency of the optical flow fields is then developed to segment the flow caused by moving objects instead of noise. Such an approach for motion detection is described as follows.

2.1.1 Producing the Optical Flow

A fundamental problem in motion analysis on image sequences is the measurement of image velocity (the terms *optical flow*, *image velocity*, and *motion field* will be used interchangeably in this report). The goal is to compute an approximation to the 2D image velocity — a projection of the 3D velocities of surface points onto the imaging surface — from spatiotemporal patterns of image intensity. Barron et al. [4] have compared and implemented several optical flow computation algorithms, including methods reported by Anandan [2], Fleet & Jepson [17], Heeger [21], Horn & Schunk [22], Lucas & Kanade [32], Nagel [37], Singh [48], Uras et al. [65], and Waxman et al. [66]. They conclude that the method developed by Lucas and Kanade performs most reliably and consistently over all the test images.

To conduct the experiments, an independently moving truck is simulated in the helicopter image *line* sequence [49]. The truck is a region extracted from one of the frames, and it is patched at the location in each image frame according to a projected 3D constant velocity

v = (26.67, 8.33, 0) feet/sec, starting from position $(X, Y, Z) = (640, -550, 0)$. Figure 1 depicts the 2D positional relationships among the trucks and the helicopter during the sequence. The first and the final frames of the image sequence are shown in Figure 2. The algorithm by Lucas and Kanade is then applied on Frame 5. The foreground pixels in Figure 3 shows the locations where optical flow information is available as the algorithm usually does not produce complete flow field outputs. Only 55% of total pixels have optical flow reports in Figure 3. Figure 4 shows the computed optical flow fields.

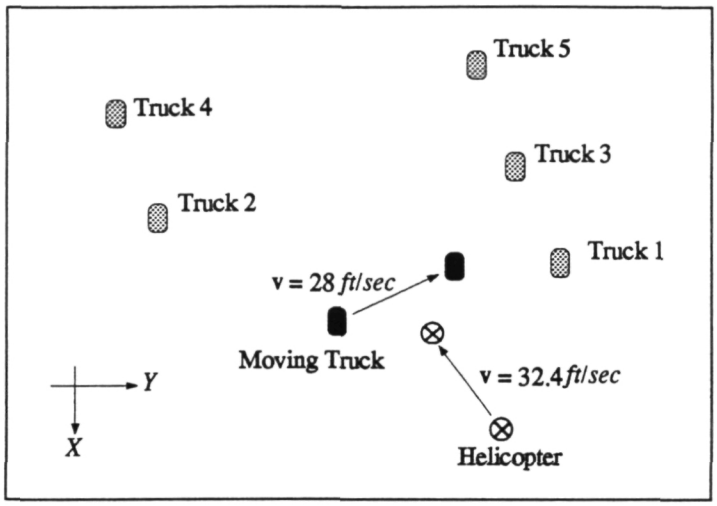


Figure 1: The 2D geometric relationships among the trucks and the helicopter.

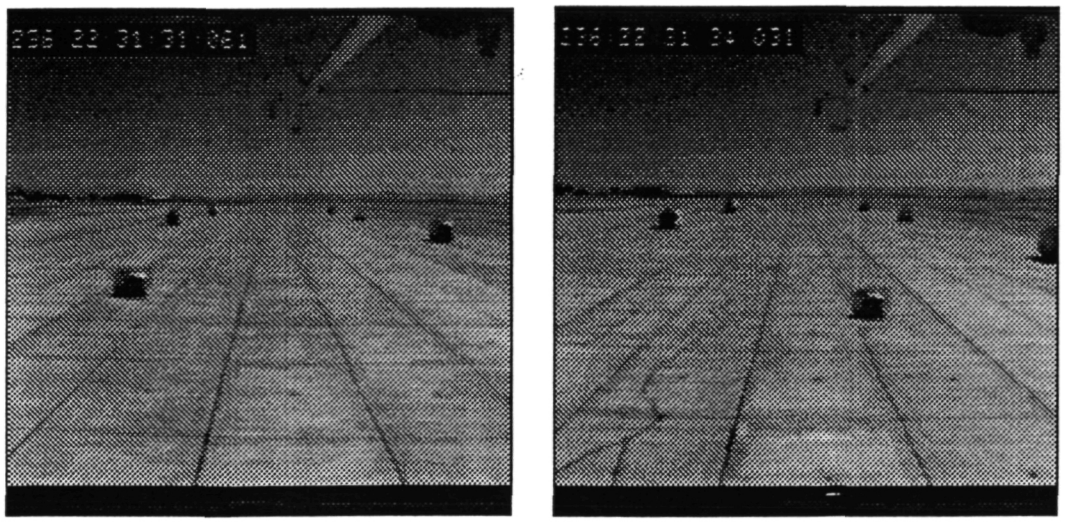


Figure 2: The first (left) and the last (right) frames of the image sequence which contain a moving truck.

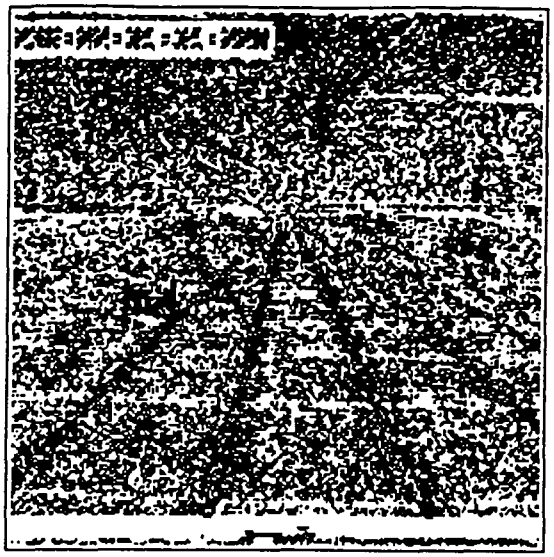


Figure 3: The pixel locations (black) where optical flow reports are available

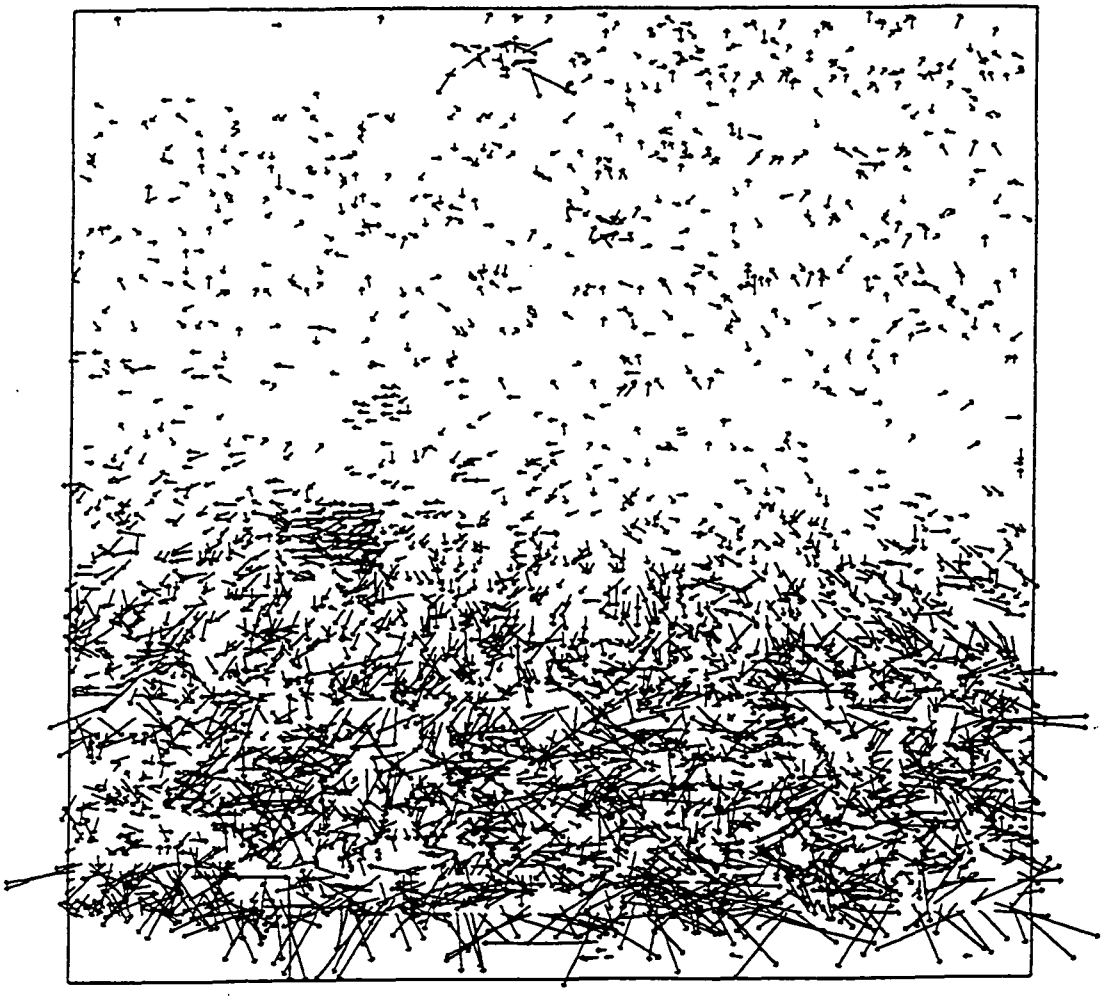


Figure 4: The computed optical flow in Frame 5.

2.1.2 The Constraint Ray Filtering

The ability to rapidly detect moving objects seems to be almost universal in animals with eyes. A few studies have concentrated specifically on the detection of moving objects by a moving observer. Jain [28] describes how a polar transform can make non-stationary objects easily discriminated by an observer undergoing known translation. Thompson and Pong [62] describe similar principles that can be used in a wider variety of circumstances, when various aspects of the observer motion are known. Bhanu et al. [5] propose a method of detecting moving targets based on the identification of a fuzzy focus of expansion and a qualitative analysis of the motion of scene points. Nelson [38] proposes a method called the *constraint ray filtering* for motion detection. It is based on the fact, noted in the context used by Thompson and Pong [62], that in a rigid environment, the projected 3D velocity at any point in the image is constrained to lie on a 1D locus in velocity space whose parameters depend only on the camera motion. Thus in principle, if the motion field and camera motion are known, an independently moving object can be detected because its projected velocity is unlikely to fall on this locus. This can be seen by the following derivations. Consider a camera observing a stationary point p in the scene. Following Smith's [49] formulation for modeling the imaging system, the image point (u, v) in unit of pixels corresponding to the point p is given by the perspective projection equations:

$$\begin{cases} u = u_0 + Af(y/x) \\ v = v_0 + f(z/x) \end{cases}$$

where $p = (x, y, z)$ is the object's position relative to the camera, (u_0, v_0) is the image center, A is the aspect ratio, and f is the focal length. As the camera moves, the image of p will also move. If p is assumed fixed in the Earth frame, the rate of change of p in the camera's axes system can be determined using the following equation [31]:

$$\dot{p} \equiv \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = -T - R \times p = \begin{bmatrix} -V_x \\ -V_y \\ -V_z \end{bmatrix} + \begin{bmatrix} -z\omega_y + y\omega_z \\ -x\omega_z + z\omega_x \\ -y\omega_x + x\omega_y \end{bmatrix}$$

where $T = (V_x, V_y, V_z)^T$ and $R = (\omega_x, \omega_y, \omega_z)^T$ are the camera's translational and rotational velocities, respectively. Differentiating the perspective projection equations with respect to time, we have the image velocity $V = (\dot{u}, \dot{v})$, where

$$\begin{cases} \dot{u} = Af(\dfrac{\dot{y}x - y\dot{x}}{x^2}) \\ \dot{v} = f(\dfrac{\dot{z}x - z\dot{x}}{x^2}) \end{cases}$$

Substituting for $\dot{p}$ according to the above equation yields the well-known optical flow equations which relate camera motion, object motion in the image, and the object's range x:

$$\begin{aligned} \dot{u} &= Af\left\{\left(-V_y - x\omega_z + z\omega_x\right)x - y\left(-V_x - z\omega_y + y\omega_z\right)\right\}/x^2 \\ &= \left\{-AfV_y + (u-u_0)V_x\right\}/x + A(v-v_0)\omega_x + \frac{(u-u_0)(v-v_0)}{f}\omega_y - Af\left\{1 + \frac{(u-u_0)^2}{A^2f^2}\right\}\omega_z \end{aligned}$$

$$\begin{aligned} \dot{v} &= f\left\{\left(-V_z - y\omega_x + x\omega_y\right)x - z\left(-V_x - z\omega_y + y\omega_z\right)\right\}/x^2 \\ &= \left\{-fV_z + (v-v_0)V_x\right\}/x - \frac{(u-u_0)}{A}\omega_x + f\left\{1 + \frac{(v-v_0)^2}{f^2}\right\}\omega_y - \frac{(u-u_0)(v-v_0)}{Af}\omega_z \end{aligned}$$

The optical flow can be decomposed into two components:

$$V = V_t + V_r$$

where $V_t = (\dot{u}_t, \dot{v}_t)$ and $V_r = (\dot{u}_r, \dot{v}_r)$ are the components due to camera translation and rotation, respectively, hence

$$\begin{cases} \dot{u} = \dot{u}_t + \dot{u}_r \\ \dot{v} = \dot{v}_t + \dot{v}_r \end{cases}$$

where

$$\begin{cases} \dot{u}_t = \{-AfV_y + (u-u_0)V_x\}/x \\ \dot{v}_t = \{-fV_z + (v-v_0)V_x\}/x \end{cases}$$

$$\begin{cases} \dot{u}_r = A(v-v_0)\omega_x + \dfrac{(u-u_0)(v-v_0)}{f}\omega_y - Af\left\{1+\dfrac{(u-u_0)^2}{A^2f^2}\right\}\omega_z \\ \dot{v}_r = -\dfrac{(u-u_0)}{A}\omega_x + f\left\{1+\dfrac{(v-v_0)^2}{f^2}\right\}\omega_y - \dfrac{(u-u_0)(v-v_0)}{Af}\omega_z \end{cases}$$

In other words, letting $V_t = (V_y/x, V_z/x)$, the optical flow can be represented by

$$V = V_r + V_{yz}/x \tag{1}$$

where $V_{yz} = (V_y, V_z)$ is the projected translational velocity. Note that only the optical flow due to translation V_t is a function of the object's range x. Hence, given the range of the stationary point p and the camera motion parameters $(V_x, V_y, V_z, \omega_x, \omega_y, \omega_z)$, the image velocity of p is uniquely determined. On the other hand, if p is not stationary, the image velocity will be unlikely to satisfy the above equations. This provides a constraint in motion detection called the *constraint ray filtering* [38] — for a stationary scene point, its projected image velocity is a function of the range which is characterized by Eq. (1). This is illustrated in Figure 5, where the image velocity has to be some point on the half line (ray) shown by the thick solid ray. If the computed optical flow does not fall on the constraint ray, it must be due to an independently moving object. Motion detection thus becomes a task to filter out the optical flow which does not satisfy the constraint.

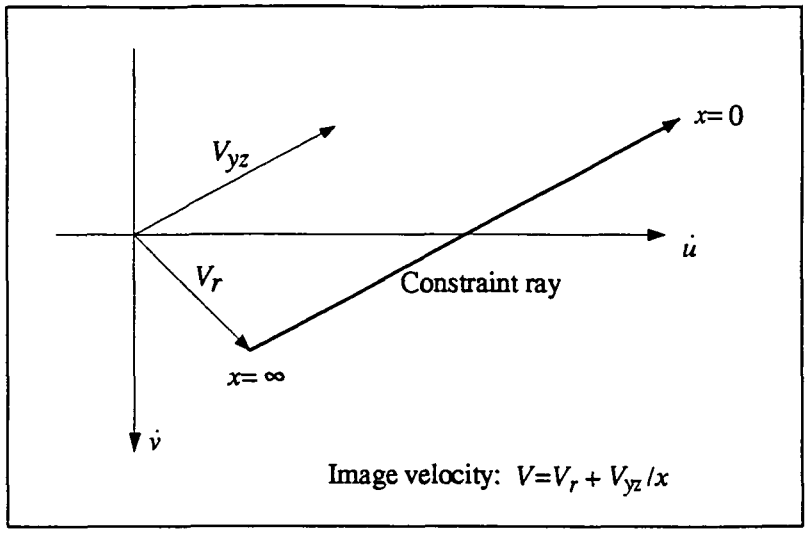


Figure 5: The constraint ray generated by Eq. (1) for x ranging from zero to infinity.

2.1.3 Constraint Region Filtering

There are three problems in applying the constraint ray filtering. First, it is obvious from Eq. (1) that the constraint ray goes to infinity if there is no a priori knowledge about the range of x, resulting in an unlimited constraint. Second, the constraint ray is too strict to compensate for the inaccuracies in the camera motion parameters. And finally, optical flow computation is generally very noisy — some flow dissatisfying the constraint may actually be due to noise, instead of moving objects. Some practical considerations are useful in coping with these problems, as described below.

Thresholding the Constraint Ray

It is easy to see that, in Eq. (1), the translation component vanishes when x approaches infinity, and the rotation component dominates the constraint. On the other hand, if x approaches zero, the optical flow can be very large, resulting in a loose constraint. Knowing the range of x can help in making the constraint more strict. In practice, a point in the scene has to have some distance from the camera. If this distance is known to be sufficiently large, we end up with a stronger constraint, i.e., the constraint ray in Figure 5 becomes a line segment. This consideration provides

more constraint in filtering out the optical flow due to moving objects. In the helicopter image sequence, if the height of the camera center from the ground level is z, the pitch angle is θ, the vertical field of view of the camera is 2α, and the highest object on the ground has height h ($h < z$), then the range threshold, x_r, which is the nearest distance of an object to appear in the image, can be computed by the following equation (see Figure 6):

$$x_r = (z-h)\tan(\frac{\pi}{2}-\theta-\alpha).$$

The larger the x_r, the shorter the constraint segment. As a typical example, if the helicopter is 20 feet above the ground, and the pitch angle and the field of view of the camera are about 4 and 40 degrees, respectively, then the minimum distance for an 8-foot high truck to appear in the image is about $x_r = 45$ feet. This value can reduce the length of the constraint ray considerably as the magnitude of the image velocity (pixels/sec) is usually in the order of tens.

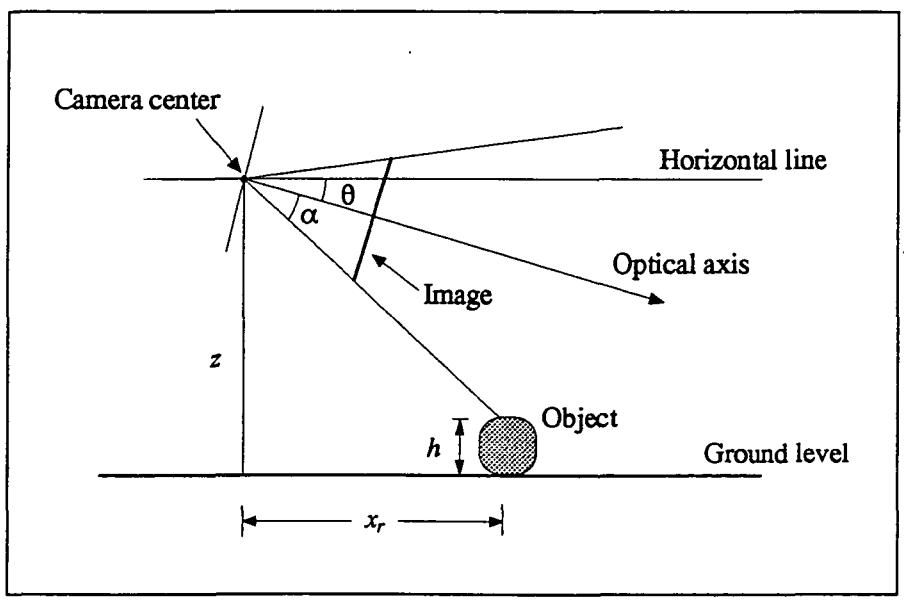


Figure 6: The minimum range x_r for an object to appear in the image.

On the other hand, if the helicopter is navigating in an unknown terrain, it is impossible to know h in advance. Moreover, there is a need to navigate *around* objects instead of *above* them,

which is useful in high-threat battle areas, where helicopters have to fly close to the surface of the earth to utilize the surrounding terrain, vegetation, or man-made objects in order to minimize the risk of being detected by the enemy. This type of flight is called the *nap-of-the-earth* (NOE) flight. In such cases, there is still a minimum distance (at least the distance from the camera to the front end of the rotor) in order for the helicopter to navigate safely without bumping into the object. This minimum distance also provides a threshold in the length of the constraint segment.

Expanding the Constraint Segment

Due to the inaccuracies in the camera motion parameters, the constraint segment is too strict to be practical. Taking into considerations the inaccuracies of the camera motion parameters, the constraint segment expands into a region called the *constraint region*, as illustrated in Figure 7. The resulting filtering method now becomes the *constraint region filtering*. After applying this filtering on the computed optical flow in the image, any flow not in the constraint region is considered due to independently moving objects. Figure 8 and Figure 9 depict, respectively, the optical flow satisfying and dissatisfying the constraint after filtering. The sky area is not processed as it is of no interest. Also, any optical flow with too small or too large a magnitude is discarded as they tend to be noisy. The thresholds are set to 0.1 and 8 pixels/frame, respectively, in the experiments since the greatest image velocity measured in the *line* sequence is about 6 pixels/frame.

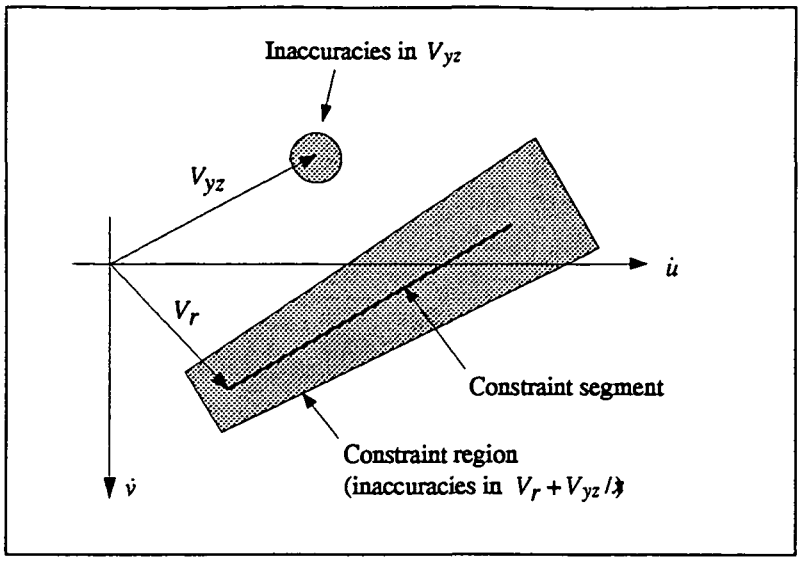


Figure 7: The constraint region resulted from thresholding and expanding the constraint ray.

2.2 Moving Object Segmentation

The purpose of moving object segmentation is to label regions in the image which correspond to moving objects. As is well known, image segmentation is an extremely difficult problem. A considerable knowledge about the structures and properties of the scene has to be available in order to obtain good results. Clustering is one of the important problems in segmentation, where different pixels (or group of pixels) are merged to form one cluster (or one region) if they satisfy certain criteria. In the experiments, only a minor degree of clustering is performed because it is desirable that a single object can be decomposed into several smaller objects with the same properties, rather than combining two different objects into a large one, although they may have the same properties. In other words, it is allowable to have two or more segmented regions which actually correspond to the same object in the scene. This loosely defined segmentation is appropriate since we would like the segmentation to be only a low-level processing — detection is the most important

objective, not identifying a whole moving object. Therefore, the idea here is to group different regions only when there is strong evidence that they should belong to the same object.

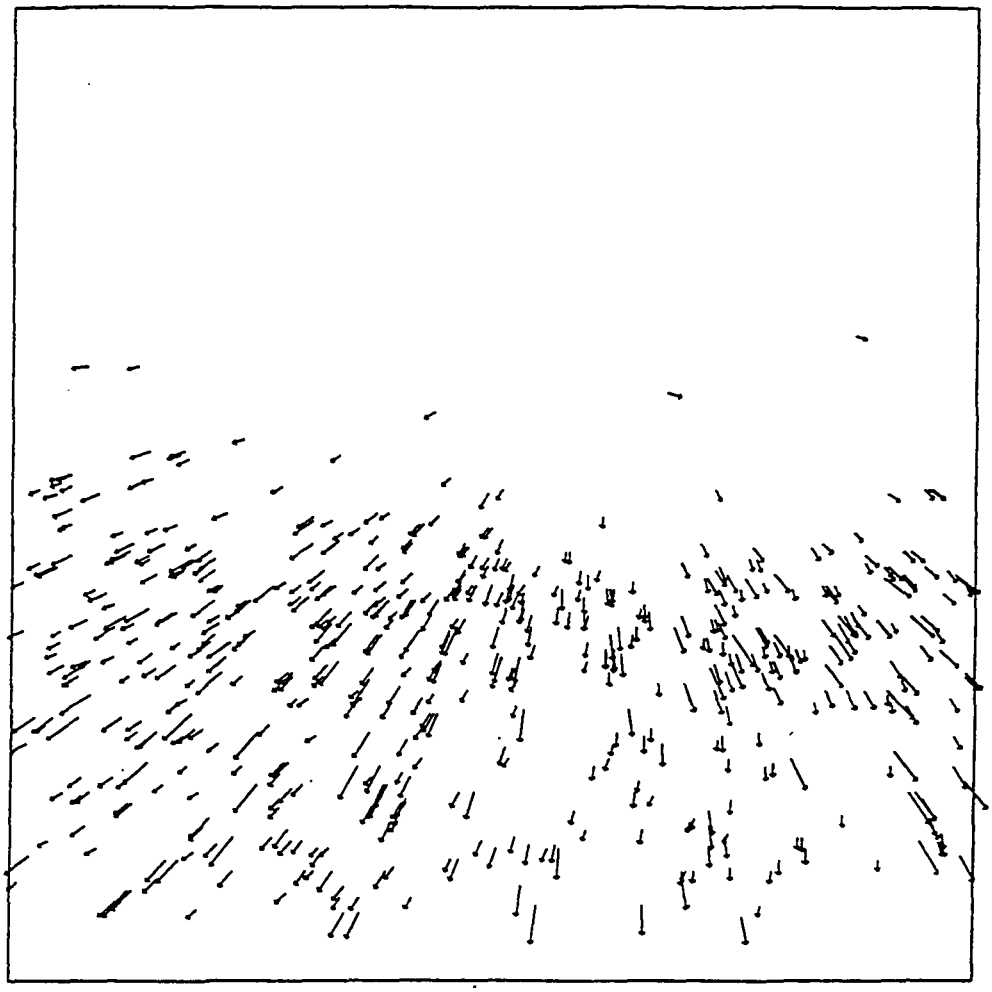


Figure 8: The optical flow satisfying the constraint.

After constraint region filtering, optical flow which may be due to moving objects are extracted. This result is the input of the segmentation process. It is very obvious from Figure 9 that most of the computed image velocities are actually due to noise (resulting from either the image noise or the algorithm). Segmentation hence serves another purpose: to label regions which have *reliable* optical flow reports. As noisy image velocities occur randomly, they have no common characteristics, such as similar magnitudes or orientations, in a small neighborhood in the same

image, nor have they common properties over time. This observation is the basic idea of the developed segmentation method. For a pixel in the image to be classified belonging to a moving object and not to the noise, it has to have reliable image velocity, i.e., it should have local support from its neighbors and it should maintain its properties over time. The local support means there should be a number of other pixels in the neighborhood which have the same image velocity. This is called the *spatial consistency*. In addition, the spatial consistency should maintain its properties in successive images. This is called the *temporal consistency*. Based on these two criteria, the first step of the segmentation is to select reliable image velocities.

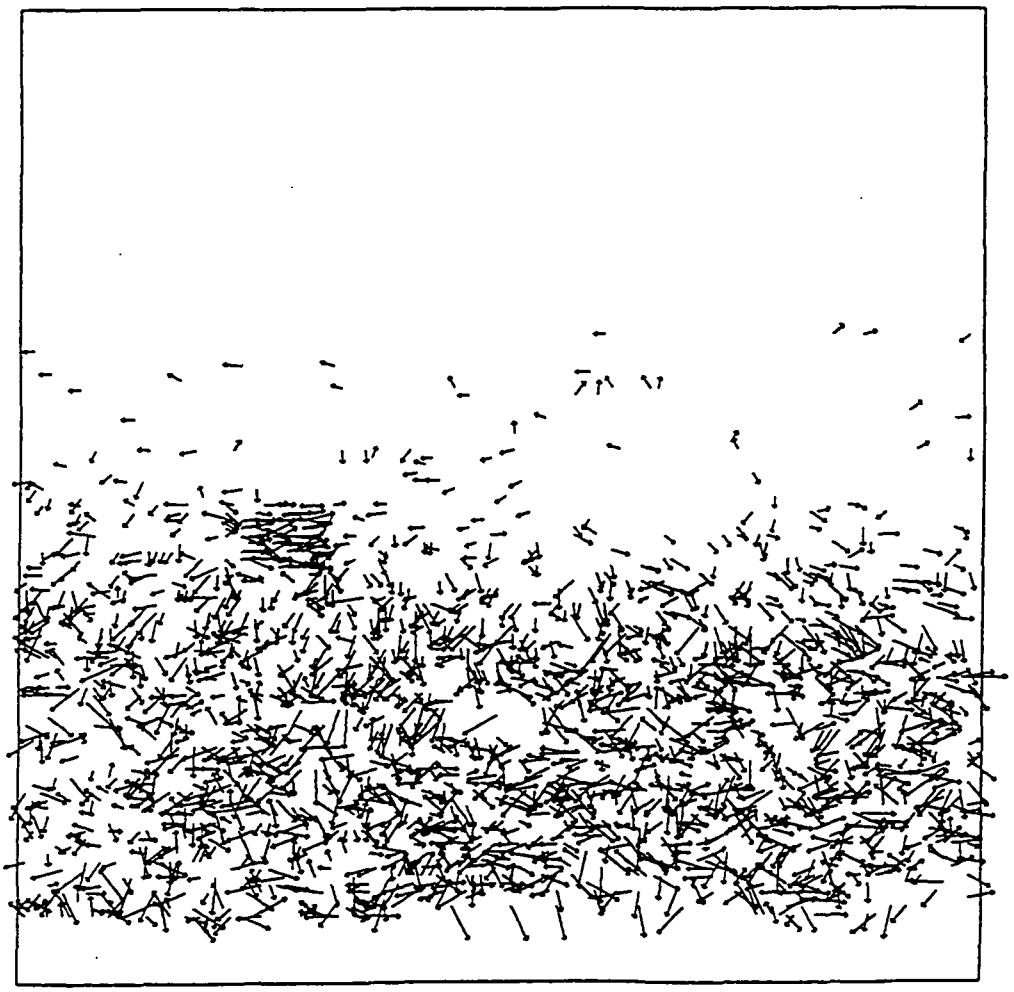


Figure 9: The optical flow dissatisfying the constraint.

2.2.1 Consistency of the Image Velocities

The segmentation of moving objects is based on grouping the reliable image velocities that exhibits similar properties both in a small region in the same image and over image frames. Two image velocities at positions $\mathbf{X}_1$ and $\mathbf{X}_2$ in image j are considered the same if they have the same magnitude and orientation, i.e.,

$$\Phi(\mathbf{x}_1^j, \mathbf{x}_2^j) = \begin{cases} \text{true, if } \left|m(\mathbf{x}_1^j) - m(\mathbf{x}_2^j)\right| < S_m \text{ and } \left|\theta(\mathbf{x}_1^j) - \theta(\mathbf{x}_2^j)\right| < S_\theta, \\ \text{false, otherwise,} \end{cases} \tag{2}$$

where $m(\mathbf{X})$ and $\theta(\mathbf{X})$ are the magnitude and the orientation of the image velocity, respectively, and S_m and S_θ are the thresholds. The test of spatial consistency is conducted by a voting system. The vote is defined as

$$S(\mathbf{x}^j, \mathbf{x}_k^j) = \begin{cases} 1, \text{ if } \Phi(\mathbf{x}^j, \mathbf{x}_k^j) \text{ is true,} \\ 0, \text{ otherwise,} \end{cases} \tag{3}$$

where $\mathbf{x}^j$ and $\mathbf{x}_k^j$ are two locations in image j. Eq. (3) says that the source pixel $\mathbf{x}^j$ will receive one vote if the image velocity at position $\mathbf{x}_k^j$ has the same magnitude and orientation. The spatial consistency at location $\mathbf{X}^j$ can thus be computed by considering a neighborhood defined by $R(\mathbf{X}^j)$ (a 9×9 window in the experiments) and compute the ratio of the number of votes to the total number of pixels in the region. That is,

$$C_s(\mathbf{x}^j) = \frac{1}{N_R} \sum_{k \in R(\mathbf{x}^j)} S(\mathbf{x}^j, \mathbf{x}_k^j)$$

where N_R is the total number of pixels in the neighborhood. If the ratio exceeds some threshold, say 0.5, which corresponds to half of the total pixels in the region, the pixel in question is consistent with its defined neighborhood. In other words, most of the image velocities in the defined region

are the same as the one in question. Taking temporal information into consideration, the spatial consistency test is applied over a number of frames. This gives a total measure of the image velocity consistency at pixel position $\mathbf{x}^k$ in image k:

$$\begin{aligned} C(\mathbf{x}^k) &= \frac{1}{2N_F+1} \sum_{i=k-N_F}^{k+N_F} C_s(f(i,\mathbf{x}^k)) \\ &= \frac{1}{(2N_F+1)N_R} \sum_{i=k-N_F}^{k+N_F} \sum_{j\in R(f(i,\mathbf{x}^k))} S(f(i,\mathbf{x}^k), \mathbf{x}_j^i) \end{aligned} \tag{4}$$

where $(2N_F+1)$ is the number of image frames considered, and $f(i,\mathbf{x}^k) = \mathbf{x}^k + (i-k)\dot{\mathbf{x}}^k$ predicts the position in image i where pixel $\mathbf{x}^k$ is going to move, according to the image velocity information $\dot{\mathbf{x}}^k$. Thus the spatial consistency test is performed in different positions over image frames. An implicit assumption in Eq. (4) is that the image velocity is constant over the image frames considered. This assumption is sustained if the following two conditions are satisfied. First, the image acquisition rate is high enough so that the change of velocity is smooth within the considered frames. In addition, the number of image frames considered is not large. The first requirement is satisfied since the image sampling rate is 30 frames/sec for the helicopter image sequences. The second requirement is also satisfied if we choose only a small number, say 5, of frames. Figure 10 shows the result after image velocity consistency test.

2.2.2 Segmentation

The result after applying the consistency test is the image velocities which are reliable and consistent both spatially and temporally. These velocities are thus considered to be due to independently moving objects in the scene. The next step of segmentation is to merge individual pixels into regions which belong to an object in the scene. The segmentation process consists of three steps:

Step 1: Group the pixels which are 8-connected and satisfy constraint Φ in Eq. (2). This step produces a primitive segmentation of regions. The result is shown in Figure 11. The average image velocity of each region is then computed and a rectangular bounding box is assigned to enclose the region. The reasoning here is that if the pixels have the same image velocity and they are connected, they should belong to the same object. There is, however, an exception, where two scene objects have the same image velocity and their image projections touch or occlude each other. In such a case, there is no way of distinguishing between the two by considering only the image velocity information. Additional knowledge, such as the intensities or the shapes of the objects, is needed for further segmentation. As long as the two objects have the same image velocity and their 2D projections touch each other, there is no harm in regarding them as one.

Figure 10: The image velocities selected by the consistency test.

Step 2: Delete any bounding box which is completely enclosed by another and has the same average image velocity as that of the enclosing box. As a consequence of the noise from optical flow computation and the consistency test, regions supposedly belonging to a single object tend to be broken and fail to be grouped as one in Step 1. Now, Step 2 comes to the rescue. It is based on the rationale that if a small region is completely enclosed by a large region and both have the same image velocity, the small region must be part of the large one. The two objects in the image can safely be regarded as one even if they are actually different objects in the scene as long as they have the same velocity. As mentioned before, it is desirable to keep the object segmentation as low-level processing. Unconnected regions, although they may have the same image velocity and be close to each other, will not be grouped since there is no evidence of their belonging to one object. Therefore, no further grouping of pixels or regions will be done after this step.

Step 3: Classify regions according to their image velocities. That is, regions with same image velocity will be put together to form a set. Higher level processing can thus use this information to perform object recognition or tracking. For example, the geometric relationships among the members in a set may indicate that they actually correspond to a single object. The relationships among sets are also useful to analyze relative motion between objects. The results after steps 2 and 3 are shown in Figure 12, where small regions are all discarded.

2.3 Tracking Moving Objects

In many applications, accurate responses based on visual information are needed to control the operations of moving objects. These include, for example, robotics, cargo handling, vehicle guidance, tele-operated manipulators, and mining equipment. Because of the needed visual feedback, constant mental and physical alertness is required for the operator. An automatic vision system could free the operator from the need to control every operation in minute detail, eliminate

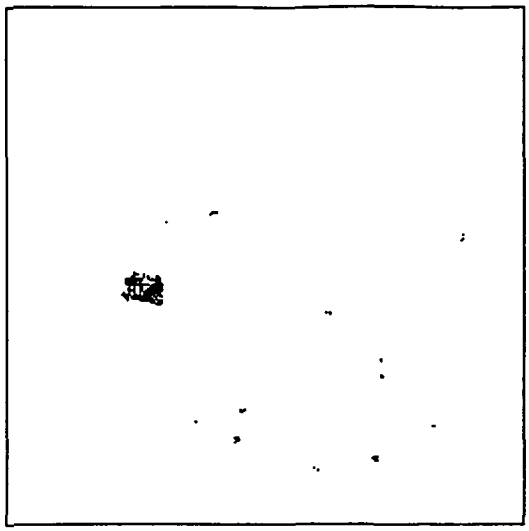


Figure 11: The result of segmentation step1 (the sky is not processed)

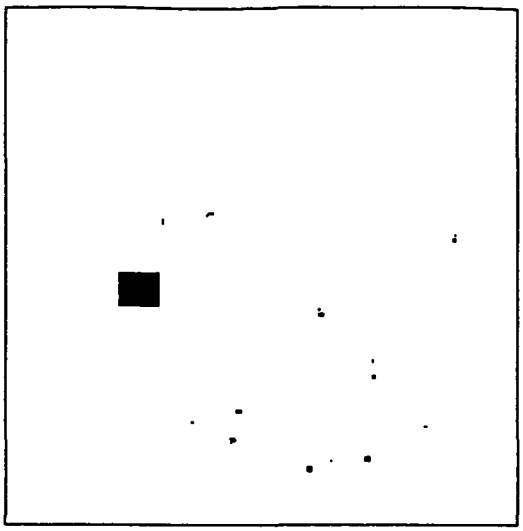


Figure 12: The bounding box of the segmented regions.

monotonous tasks, and enable one person to supervise more machines at the same time. This type of automatic visual control relies on accurate tracking of moving objects in the scene through analysis of sequences of images. An essential issue in tracking an object is to identify the feature points at different times that represent the same physical object. This process is called the correspondence problem and is an important research topic in both motion analysis and stereopsis. Most of the existing approaches are based on the matching of similar image features between consecutive frames. Examples of the criteria for the similarity measurement include the intensity statistics [22, 27, 62] average speed computed [12, 41], and spatial-temporal relations [40, 46]. The problem of tracking image features comes mostly from the temporal inconsistency of the feature extractor, which deals with thresholding and hence is not a function of time. Temporally inconsistent features make feature matching extremely difficult because an image feature in one frame can be matched to none, one, or several image features in the next frame. Furthermore, it is highly likely to establish incorrect correspondences if similar features are close to each other.

Therefore, the success of feature tracking depends significantly upon how to select reliable and discriminating features in the scene.

Crowley et al. [13] use straight lines as the image features for tracking. An elegant way of representing line segments is developed. The parameterization of a line segment includes several attributes each of which is associated with a variance measure to handle noise. Tracking of a line segment is to search correspondence which satisfies certain constraints such as the position, orientation, length, etc. Huttenlocher et al. [23, 24] proposed a model-based feature tracking approach which is able to handle non-rigid objects. The image of a moving 3D object is decomposed into a 2D motion and a 2D shape change. A bounding box is first manually selected in the image to specify the location of the object to be tracked. *Hausdorff distance* is then computed to measure the difference between the 2D pattern of the image object and its neighborhood. The position which minimizes the Hausdorff distance is identified as a match. Zheng and Chellappa [70] presented an automatic ego motion compensation based feature detection and correspondence algorithm. They first register two images to compensate for the motion due to the camera ego motion. The remaining motion in the image will then be completely due to moving objects. Gabor filtering is finally used to extract distinctive features for matching. Wu et al. [68] used an extended Kalman filtering technique to estimate the 3D motion of objects. They assumed that motion correspondences have been established and hence only motion estimation was discussed. Koller et al. [29] developed a complete model-based motion analysis system for tracking vehicles in road traffic scenes. A very efficient and universal model composed of edge segments is used to represent all kinds of vehicles. In their research, motion characteristics of a vehicle are studied in detail. Zhang and Faugeras [69] also use the extended Kalman filtering technique in estimating object motion parameters.

Tracking of moving objects is also popular in robotics and automation, especially for the eye-in-hand robotic configuration, where a camera is mounted on a robot arm and the task is to grasp or manipulate the tracked object. The main objective of the vision system is to keep the tracked object in a specified location in the image through interacting with the control system. Researchers in this area emphasize more on estimating and controlling the robot arm states [16, 18, 39, 47]. For these applications, objects in the scene are usually known a priori and the lighting condition is well controlled. Therefore, 3D model-based object recognition is suitable for tracking.

Most existing approaches for object tracking can be attributed to having some of the following characteristics: the camera is assumed stationary, the 3D geometry of the objects in the scene is known, or the types of motion of the objects are restricted, e.g., only in a 2D space. For our application in which the task is to track objects seen from a helicopter, several characteristics can be distinguished from other applications. First, objects in the scene are not known a priori; hence, it is impossible to use predefined 3D models for object recognition. Second, the camera is moving with reference to the environment. Third, objects in the scene may appear small in the image. Edge-based tracking is thus not a good choice since the line segments detected on small objects are unreliable. And finally, illumination can be an important effect since the outdoor situation is dealt with. For example, the shadow of an object changes from frame to frame due to object motion, making feature extraction and matching more complicated. A tracking method is introduced in this section which deals with the above problems. It is based on extracting a 2D model, called the *shape*, of an object *directly* from analyzing the sensed image. No a priori knowledge of the object model is thus required. Tracking of objects is then formulated as matching similar shapes from one frame to the next. By matching the shapes and not the features, the notorious correspondence problem can therefore be avoided. The shape of an object is also allowed to change in order to simulate the projection changes due to object motion. Once the tracking

results are obtained, an extended Kalman filter is then used to recursively estimate the motion parameters of the object.

2.3.1 Object Tracking

A common approach to object recognition and tracking in computer vision is to identify image features and establish feature correspondences. Some approaches track objects by using model-based recognition techniques [29, 18]. Others track features by minimizing the sum-of-differences (SSD) measures [34, 39, 40]. Once the correspondence is established, the motion parameters can be determined. Unfortunately, the process of finding feature correspondence has been recognized to be a very difficult and noisy process. The problem comes from various sources, such as the noise of the image, illumination conditions, the noise from the feature extractor itself, occlusion and disocclusion, etc. We argue that the difficulty in establishing feature correspondence results from the assumption that the image features are independent of each other, hence, the confusion that one-to-many or many-to-one correspondences may be established. Ignoring the inter-relationships among features is the major reason for this confusion. Therefore, tracking should be done at the object level instead of the feature level. Several advantages can be easily identified. First, the object level is the "right" place to work since the task is to track moving objects. Second, as an object usually consists of a number of features, they are all considered in the matching process. Hence, a better matching result is obtainable because more information is taken into account. Third, by defining an object, the inter-relationships among features are implied. Blindly matching among features can thus be avoided.

Definition of an Object

A problem with working at the object level is that it is difficult to identify an object in the image if no a priori knowledge of the object model is given. In such a case, object segmentation or human interference will be required to provide the initial objects. As described in Section 2, the output of the motion detection module is a set of bounding boxes which defines each moving object in the image. These boxes hence provide the initial objects for tracking. An object is defined to be a region in the edge-detected image. In other words, an object is extracted directly from the image and it is defined as a 2D rectangular box, called the *shape*, which consists of a group of edge-points. The size of the shape and the number of points in the shape have to be large enough in order for reliable tracking. As mentioned before, since objects in the scene can move independently and the illumination is an important factor when dealing with the outdoor situations, the 2D projections of the moving objects will vary more significantly from frame to frame than in the cases where either the camera or the objects are stationary. It turns out that the 2D projections of rigid objects are actually non-rigid in the images. Therefore, to simulate the non-rigidity, the 2D shape of an object is allowed to change moderately between image frames and it is updated after each matching step. It is worth noting that the major assumption underlying the proposed tracking method is that the 2D shape of an object will change slowly from one frame to the next. There is no assumption, however, that the 2D image motion between successive frames will be small.

2.3.2. Object Matching

A 2D shape is defined by two elements:

$$P = (b, \{p_i\}_{i=1..n}) \tag{5}$$

where b is the bounding box, $\{p_i\}_{i=1..n}$ is the set of the edge points, and n is the total number of points in the shape. In order to measure the similarity between two shapes P and Q, a similarity measure is defined as

$$h(P,Q) = \sum_{p \in P} \sum_{q \in Q} p = q$$

Hence, the larger the value h, the more similar the two shapes P and Q. Supposing the shape of an object in image I_t is denoted by S_t, the problem of matching shapes between images I_t and I_{t+1} can be formulated as searching for the best transformation g^* which maximizes the similarity measure:

$$g^* = \max_{g \in G} h(g(S_t), I_{t+1}) \tag{6}$$

where G is a group of allowable transformations. In other words, the similarity between two shapes is the maximum value of the similarity measure between them under all possible transformations of one shape with respect to the other. And the task of matching is to find a transformation in G that brings one shape near another in order for the similarity measure to be maximum. The result g^* is thus the output of the matching. For completeness, the group G should include all 2D transformations, such as translation, rotation, and scaling. However, in the experiments it is defined to be only translation in the image in order to reduce the computation complexity. In addition, since it is assumed that the shape will not change dramatically, the tracker is able to pick up some small degree of rotation and scaling of the shape by updating the shape after each matching step (described later).

In order to obtain reliable tracking results, the bounding box b and the number of points n in Eq. (5) have to be large enough and the similarity measure has to exceed some threshold. A small bounding box means the size of the object is small and it will not be tracked reliably. A small number of points indicates that the 2D shape is very simple and hence there may be multiple

solutions to Eq. (6). As an example of the degenerated case where the shape consists of only a straight line, it can be matched to any neighboring line in the next frame and these two lines can slide along each other while having the same value of the similarity measure. In the experiments, the thresholds for the bounding box and the number of points should be greater than 30×30 and 100 pixels, respectively. The similarity measure is used to indicate the goodness of matching and if it is less than some threshold, there is no match for the shape.

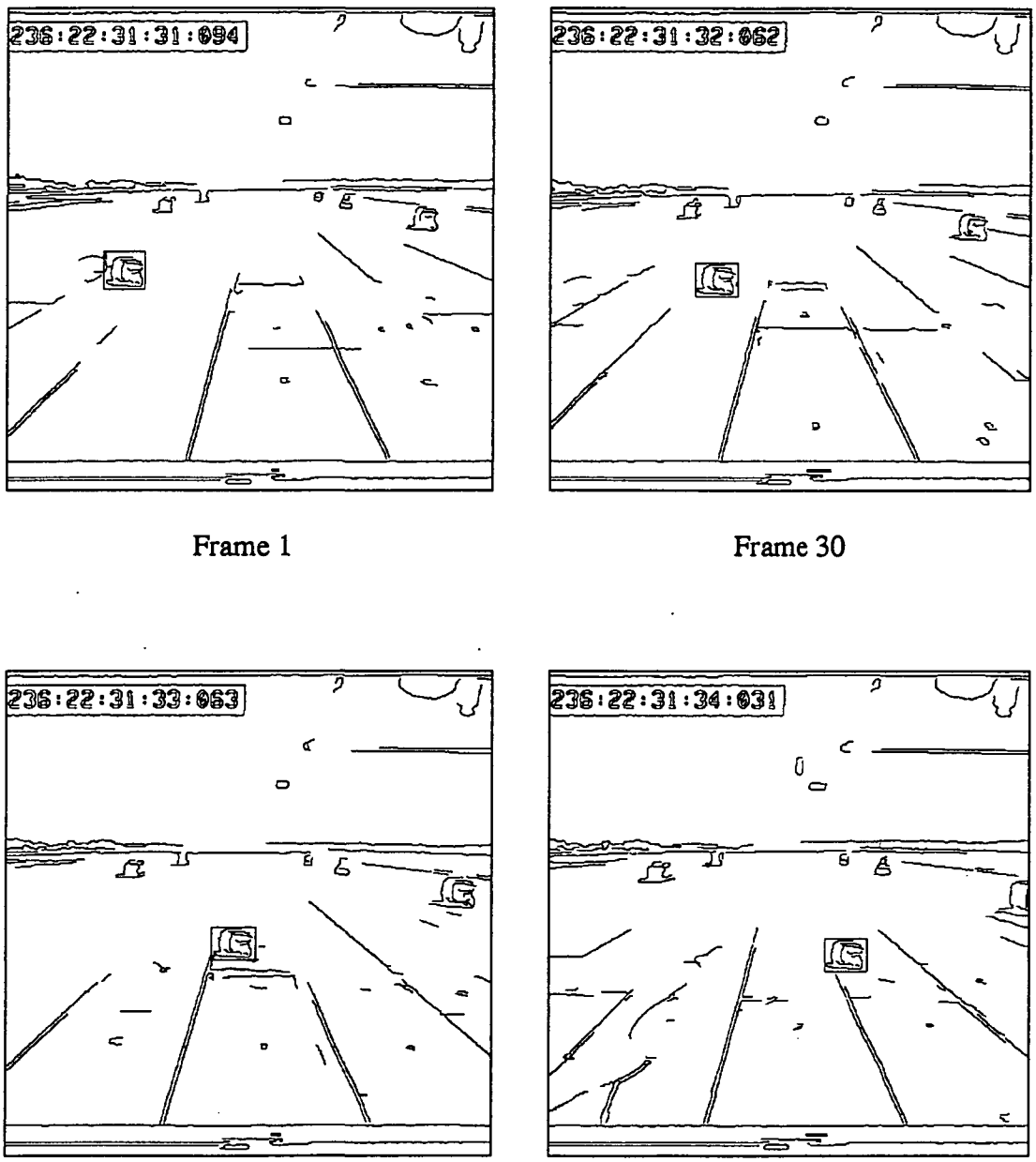
2.3.3. Updating the Shape

Since the 2D shape of an object is extracted directly from the image, it is only a projection of the object at a certain time instance and hence cannot be used as the sole model of the object for tracking in successive image frames. This problem is overcome by updating the shape after each matching step in order to keep good track of the object. The updating method is proposed by Huttenlocher et al. (1993). Having used Eq. (6) to identify the best location g^* of the shape S_t in the subsequent image frame I_{t+1}, the update of the shape can be defined as follows:

$$S_{t+1} = \left\{ q \in I_{t+1} \middle| \min_{p \in S_t} \| g^*(p) - q \| \leq \delta \right\}$$

for some distance δ. In other words, S_{t+1} is all those points of the image that are within distance δ of some point of $g^*(S_t)$. The side-effect of shape updates is that a certain degree of non-rigid motion of the object is allowed. The tracker is thus capable of tracking non-rigid objects. The choice of the distance δ controls the degree to which the method is able to track objects that change shape. For example, if $\delta = 0$ only those pixels of I_{t+1} that are directly superimposed on $g^*(S_t)$ will be included in S_{t+1}, and thus it will cause the tracker to lose the object after several frames even if the object shape does not actually change, due to noise and uncertainty in the

locations of pixels. The larger the value of δ, the more that the tracker is able to follow non-rigid motion, and to "pick up" new parts of an object that have come into view since the previous frame. Figure 13 shows the tracking of the moving object in a number of frames.



Frame 1 Frame 30

Frame 60 Frame 90

Figure 13: Tracking of the moving truck.

2.4 Motion Parameter Estimation

Given the camera's motion and the feature's locations in successive images, estimation of the motion parameters of the tracked object can be formulated as a state estimation problem using a Kalman filter. The Kalman filter is well-suited to this application because it combines redundant measurements to recursively improve its estimate over time. In addition, the state covariance matrix provided by the Kalman filter gives an indication of the estimate accuracy. To estimate the position and velocity of the moving object, the motion kinetics has to be modeled first. We assume that the motion of the object in the 3D space is only translational with constant velocity (V_X, V_Y, V_Z). Supposing that at time instance t, the location of the object is at (X_t, Y_t, Z_t) in the earth coordinate system, its location at time $t+\tau$ will then be at $(X_{t+\tau}, Y_{t+\tau}, Z_{t+\tau}) = (X_t+\tau V_x, Y_t+\tau V_y, Z_t+\tau V_z)$, where τ is the time step. Letting $\mathbf{X}_{st} = (X_{st}, Y_{st}, Z_{st})$ be the same object point in the camera coordinate system, we have

$$\begin{bmatrix} X_{st} \\ Y_{st} \\ Z_{st} \end{bmatrix} = R_t \begin{bmatrix} X_t - X_{ct} \\ Y_t - Y_{ct} \\ Z_t - Z_{ct} \end{bmatrix}$$

$$= \begin{bmatrix} r_{11}(X_{t-\tau} + \tau V_x - X_{ct}) + r_{12}(Y_{t-\tau} + \tau V_y - Y_{ct}) + r_{13}(Z_{t-\tau} + \tau V_z - Z_{ct}) \\ r_{21}(X_{t-\tau} + \tau V_x - X_{ct}) + r_{22}(Y_{t-\tau} + \tau V_y - Y_{ct}) + r_{23}(Z_{t-\tau} + \tau V_z - Z_{ct}) \\ r_{31}(X_{t-\tau} + \tau V_x - X_{ct}) + r_{32}(Y_{t-\tau} + \tau V_y - Y_{ct}) + r_{33}(Z_{t-\tau} + \tau V_z - Z_{ct}) \end{bmatrix}$$

$$\equiv h(\mathbf{X}_t) \tag{7}$$

where R_t is the rotation matrix from earth coordinate to camera coordinate system, $[X_{Ct}, Y_{Ct}, Z_{Ct}]^T$ is the camera center at time t, and vector $\mathbf{X}_t = (X_t, Y_t, Z_t, V_X, V_Y, V_Z)$ describes the position and velocity of the object. If the object point being tracked is at $[y_t, z_t]^T$ in the image, the following relationships can be established according to the perspective projection:

$$\begin{bmatrix} y_t \\ z_t \end{bmatrix} = \begin{bmatrix} fY_{st}/X_{st} \\ fZ_{st}/X_{st} \end{bmatrix} \tag{8}$$

where f is the focal length. Dropping the subscript t and defining the state vector $\mathbf{X} \equiv [X, Y, Z, V_x, V_y, V_z]^{\mathrm{T}}$ and the measurement vector $\mathbf{Z} \equiv [y, z]^{\mathrm{T}}$, the state equation and the measurement equation can be written as follows

$$\begin{cases} \dot{\mathbf{X}} = \left[V_x, V_y, V_z, 0, 0, 0\right]^{\mathrm{T}} = F\mathbf{X} \\ Z = h(X) = \left[fY_s/X_s, fZ_s/X_s\right]^{\mathrm{T}} \end{cases}$$

where

$$F = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The state equation is a time varying linear system that depends on the camera's position and orientation. The measurement equation is a nonlinear function of the states and it is related to the state equation via Eq. (7).

Solving for the state equation and representing the state and the measurement by their discrete time equivalents, the discrete time system equations can be expressed as

$$\begin{cases} \mathbf{X}_{k+1} = \Phi_k \mathbf{X}_k + w_k \\ \mathbf{Z}_k = h(\mathbf{X}_k) + v_k \end{cases} \tag{9}$$

where Φ_k is the state transition matrix:

$$\Phi_k = \begin{bmatrix} 1 & 0 & 0 & \tau & 0 & 0 \\ 0 & 1 & 0 & 0 & \tau & 0 \\ 0 & 0 & 1 & 0 & 0 & \tau \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and w_k and v_k are the process and measurement noise, respectively. Zero mean Gaussian white noise is assumed such that $w_k \sim N(0, Q_k)$ and $v_k \sim N(0, R_k)$. It is obvious that the measurement equation in Eq. (9) is a nonlinear function of the state vector. An extended Kalman filter is thus necessary. The measurement equation is linearized about the current estimate of **X** giving

$$\mathbf{Z}_k = H_k(\hat{\mathbf{X}}_k^-)\mathbf{X}_k + v_k$$

where

$$H_k(\hat{\mathbf{X}}_k^-) = \left.\frac{\partial h(\mathbf{X})}{\partial \mathbf{X}}\right|_{X=\hat{\mathbf{X}}_k^-}$$

$$= \frac{f}{X_s^2}\begin{bmatrix} r_{21}X_s - Y_s r_{11} & r_{22}X_s - Y_s r_{12} & r_{23}X_s - Y_s r_{13} \\ r_{21}X_s - Z_s r_{11} & r_{32}X_s - Z_s r_{12} & r_{33}X_s - Z_s r_{13} \end{bmatrix}$$

$$\left.\begin{matrix} \tau(r_{21}X_s - Y_s r_{11}) & \tau(r_{22}X_s - Y_s r_{12}) & \tau(r_{23}X_s - Y_s r_{13}) \\ \tau(r_{31}X_s - Z_s r_{11}) & \tau(r_{32}X_s - Z_s r_{12}) & \tau(r_{33}X_s - Z_s r_{13}) \end{matrix}\right]\Bigg|_{\mathbf{X}=\hat{\mathbf{X}}_k^-}$$

and $\hat{\mathbf{X}}_k^-$ is the current estimate of the state vector.

The Kalman filter consists of two parts: the measurement update which improves the state estimate given a new measurement, and the time update which propagates the state forward in time according to the system dynamics. In entering each iteration of the filter, the Kalman gain K_k is first computed as

$$K_k = P_k^- H_k^{\mathrm{T}}\left[H_k(\hat{\mathbf{X}}_k^-)P_k^- H_k^{\mathrm{T}}(\hat{\mathbf{X}}_k^-) + R_k\right]^{-1} \tag{10}$$

The measurement update is then performed according to the following equations

$$\hat{\mathbf{X}}_k^+ = \hat{\mathbf{X}}_k^- + K_k\left[\mathbf{Z}_k - h(\hat{\mathbf{X}}_k^-)\right]$$
$$P_k^+ = \left[I - K_k H_k(\hat{\mathbf{X}}_k^-)\right]P_k^-$$

The time update equations are

$$\hat{X}_{k+1}^{-} = \Phi_k \hat{X}_k^{+}$$

$$P_{k+1}^{-} = \Phi_k P_k^{+} \Phi_k^{T} + Q_k$$

Starting from Eq. (10), the Kalman filter will run recursively as new measurements are available. The required initial estimate for **X** in Eq. (10) can be computed by triangulation on the first few frames using Eqs. (7) and (8). Since there are six unknowns in the state vector, at least three frames are necessary to solve the simultaneous equations. However, for better initial estimates, more frames can be considered, which results in an over-determined system.

Experimental Results

The developed algorithms are tested using the tracking outputs obtained from the object tracker. The initial conditions for covariance matrices are set as follows:

$$P_0 = \begin{bmatrix} 100^2 & & & & & \\ & 30^2 & & & 0 & \\ & & 30^2 & & & \\ & & & 30^2 & & \\ & 0 & & & 30^2 & \\ & & & & & 30^2 \end{bmatrix}_{6\times 6} ft^2,$$

$$R_k = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}, \text{ and } Q_k = [0]_{6\times 6}$$

Figure 14 shows the position and velocity estimates as a function of number of frames processed. Compared to the truth values, the estimates are reasonably good. Figure 15 and Figure 16 show the error covariance measures for each parameter. As can be seen, all estimates converge before frame 40. That is, good estimates are obtainable within 2 seconds since the image acquisition rate is 30 frames/sec. The Kalman filter works interactively with the object tracker in that the window for searching can be reduced if reliable estimates of the object motion are available. This is done by

predicting the object position in the next image frame using the current motion information. The amount of computation is thus significantly reduced since the object matching takes most of the processing time in the current implementation.

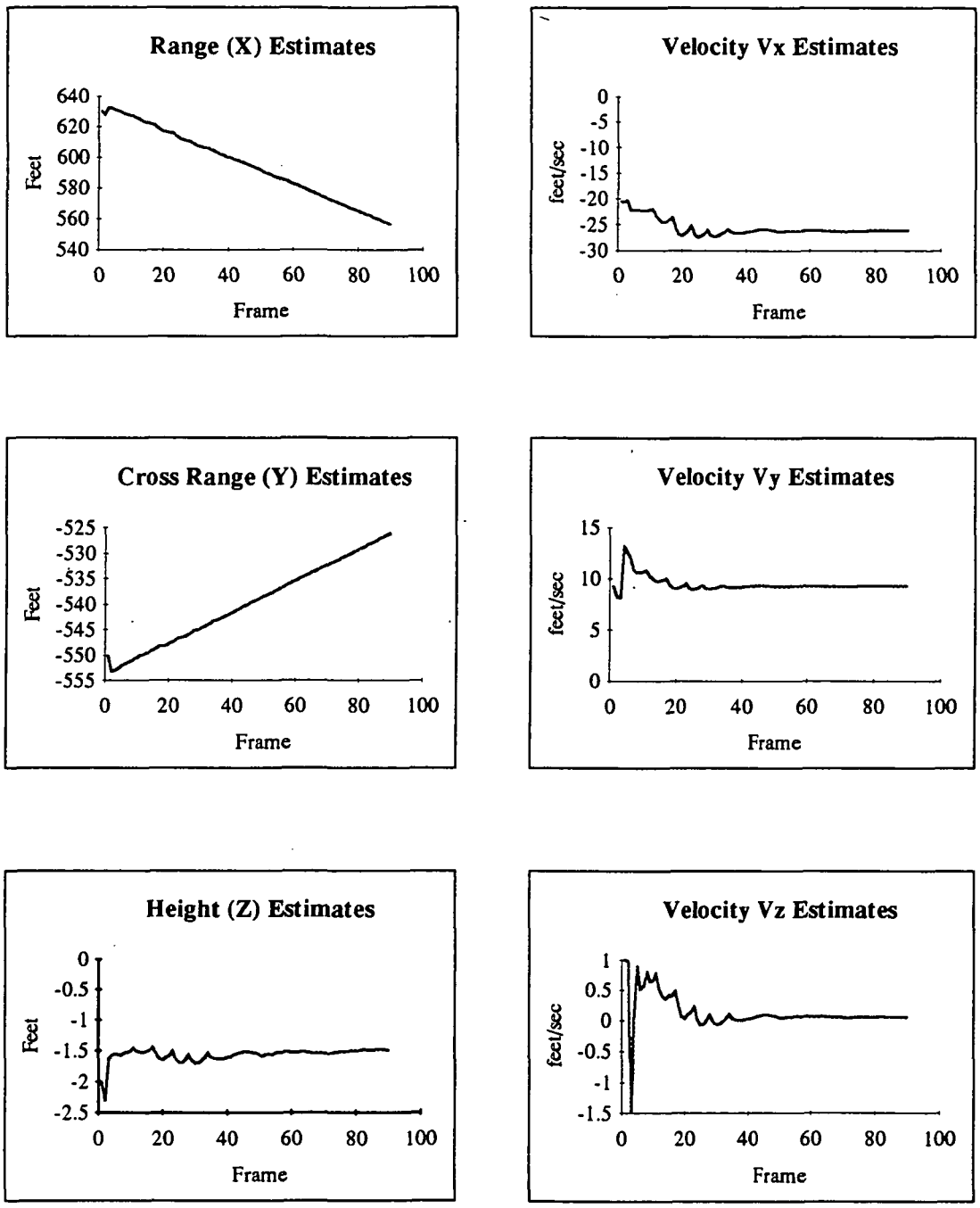


Figure 14: Motion parameter estimates of the moving truck. The truth values are X: 640 ~ 561, Y: −550 ~ −525, Z: 0, V_x: −26.67, V_y: 8.33, V_z: 0.

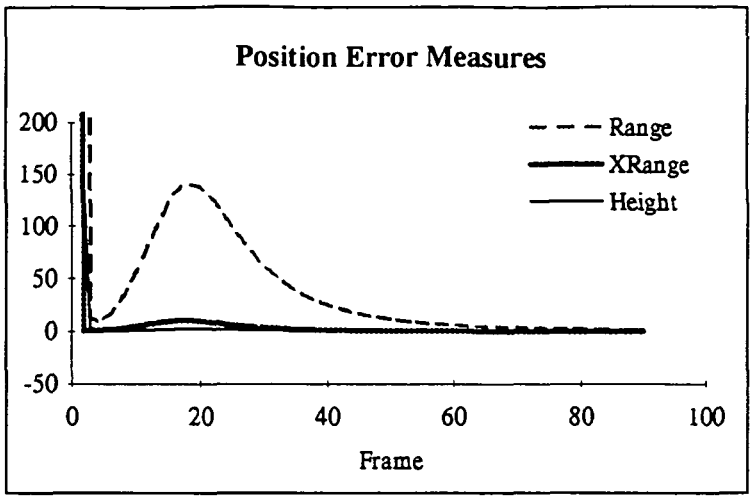


Figure 15: Error measures of position estimates.

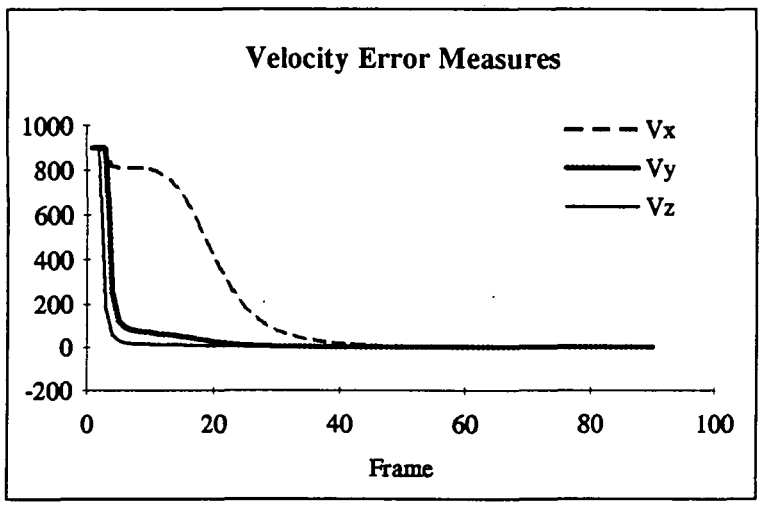


Figure 16: Error measures of velocity estimates.

3. Proposed Enhancements to the Algorithm

Even though the above algorithm was successful in detecting and tracking the moving object in the simulated test image sequence, the algorithm has many limitations. In this section of the report we list the limitations of the algorithm described in the previous section followed by possible solutions to overcome the shortcomings as a part of our future research work. In our proposed work we also address various other issues regarding the design of a robust system.

3.1 Limitations of the present algorithm

1. A one time search for moving objects is carried out at the very beginning and the algorithm does not check for any new incoming objects.

2. A single detected region could be potentially due to multiple objects occluding each other, but the algorithm does not allow breaking of a detected object into multiple objects.

3. During object segmentation, only regions of sufficiently large size and violating the flow constraint are detected as those due to moving objects. However, the size of the image region corresponding to an object is a function of the range to the object and the size of the object itself and hence the image regions corresponding to an object might appear initially small and become larger as the camera gets closer. However the size constraint for moving object segmentation does not dynamically change depending on the range to the object.

4. The search region in the next image frame is a rectangular region around the current position of the object in the image plane. The algorithm does not make use of the predicted position given by the Kalman filter. Similarly, it does not take advantage of the Kalman filter output to define the search region as a function of the estimated range and error in the estimated range given by the covariance matrix.

5. In a moving camera, moving object situation the object path cannot be determined uniquely unless a constraint such as that the object position is known at some time instant is applied. The present algorithm does not consider this uniqueness problem. Hence the solution could be potentially unstable.

6. The above Kalman filter is based on the assumption that the velocity of the moving object is practically constant and all the changes in the velocity are modeled as noise. However, for objects whose velocities change rapidly or continuously, the above model is not useful.

7. In cases where the camera is moving towards an object the image region corresponding to the object increases in size in successive frames. In addition, the object shape might change significantly from frame to frame. However, in the simulated test sequence the patch was identical in every frame of the sequence.

3.2 Proposed approach

To overcome the limitations noted in the previous section and to address other robustness issues discussed in detail in the following sections, we propose to develop the vision-based system shown in Figure 17. Each of these blocks are discussed in detail in the following sections.

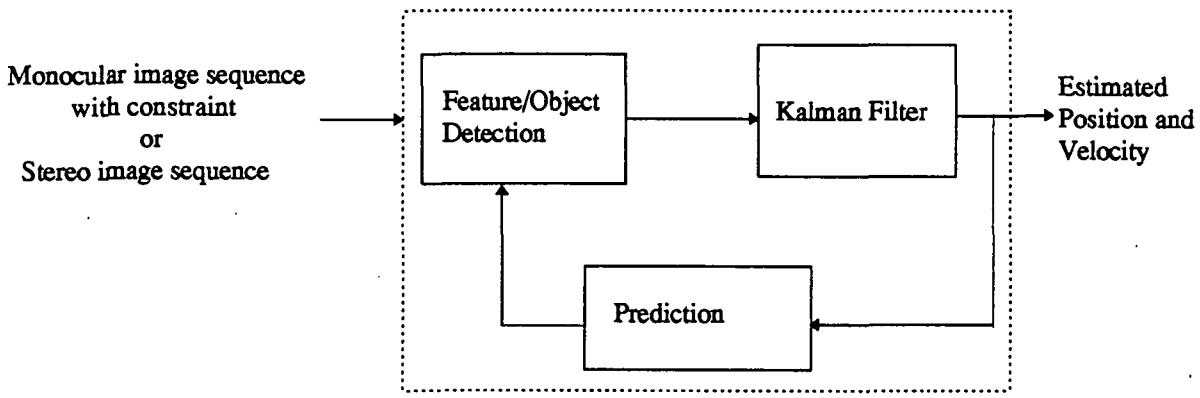


Figure 17: Block schematic of the proposed approach

The system will detect features due to moving objects or object regions in each image and will track them over the entire image sequence. The position and velocity of the object(s) in the world coordinate system corresponding to each image feature/object will be estimated using a Kalman filter based recursive estimation procedure. The predicted position of the object/feature and the error in the estimated position and velocity of the object will be used to define the search region for the objects/features in the next image. However the initial state vector cannot be computed uniquely. Assuming constant velocity for the moving objects, the problem with

estimating the initial position of the object can be solved either by use of some constraints or by use of stereo image sequence as explained below.

Use of constraints

In this method we assume that we have the knowledge about the terrain where the objects are moving i.e. z = *f(x, y)*. Hence the range to the object is known and object's velocity in the *z* direction is also constrained by the equation

$$V_z = f_x(x,y)V_x + f_y(x,y)V_y \tag{7}$$

where f_x and f_y are the partial derivatives of *f(x, y)* and V_x, V_y and V_z are the *x*, *y* and *z* components of the velocity vector. Since *z* and V_z are constrained, the initial state vector can be uniquely computed using two frames. In a Kalman filter based state estimation procedure the *x* and *y* components of the object position and velocity become the state vector and the image coordinates of the corresponding feature become the observation vector.

Use of stereo

Since a single camera is not capable of giving the range of the moving object in the general case, two or more cameras could be used. Assuming we have two cameras we can estimate the object position using two frames. In a Kalman filter based state estimation procedure, the state variables are the *x*, *y* and *z* components of the object position and velocity and the observation vector is the image positions of the object in the two cameras.

3.2.1 Feature/Object detection

Object detection was done at the feature level in the passive ranging algorithm designed by Sridhar et al. at the NASA Ames Research Center [50, 53, 54]. Range estimation was done for

every feature and these features were not grouped until the end of the sequence. Tracking every feature over the entire sequence and estimating the position and velocity for each of these features is time consuming. Since a single world object can lead to multiple image features, features originating from a single object can be grouped to form an image object. On the other hand, the motion estimation algorithm described in [57, 58] detects regions corresponding to moving objects at the very beginning. Such regions could potentially belong to multiple objects, but no provision is made to separate them at subsequent frames. In addition, the moving object detection is done only once at the beginning and new incoming objects are not detected.

We propose to integrate the two methods where the initial detection will be carried out at the feature level and these features will be grouped to form objects after they are tracked over several frames based on spatial and temporal integrity constraints. The feature level detection will be carried out on every frame. Features will be defined as regions of size *kxk* pixels with high variance. A detected feature not covered by previously detected object/feature will be considered as a new feature. However these features could be due to moving or stationary objects. To retain only the features due to moving objects, Nelson's constraint will be used. Spatially neighboring features will be considered for possible integration as single objects by considering the spatial and temporal integrity constraints.

Features that are within the *kxk* neighborhood will be assumed to satisfy the spatial integrity constraint if they have consistent velocity and position information. Features that are within the *kxk* neighborhood satisfying the spatial integrity constraint will also be assumed to satisfy the temporal integrity constraint if they could be tracked over a number of frames and are found to have consistent position and velocity information. A bounding box enclosing group of features satisfying both the spatial and temporal integrity constraints will be used as the object model.

Features grouped into an object will not be tracked individually, instead the object representing these features will be tracked over the remaining frames of the sequence.

However, this method would detect object features as well as extraneous features like those mainly due to the tire marks. Tire marks do have textural properties. These properties could be used to separate out such features.

3.2.2 Tracking moving objects/features

Tracking will be carried out separately for each of the detected features and also for the object. Initially only features need to be tracked. However after a number of frames have elapsed some of the features are likely to have been grouped to form objects by satisfying the spatial and temporal integrity constraint from where onwards only the objects need to be tracked. The search region in the next frame to find the best match for an already detected feature can be defined as an elliptical region around the image position predicted by the Kalman filter. The size of the search region could be a function of the estimated range and the error in the estimation. An autocorrelation function could be then used to find the actual position of the feature in the search region.

Object matching and tracking poses a difficult problem since the 3D shape of the object is not known and also the 2D shape of the projected 3D object in the image plane keeps changing significantly as the camera and the objects move independently. Hence it turns out that the 2D projections are actually non-rigid in the images. The initial model for the object will be the output of the motion detection algorithm and it will consist of a rectangular box containing edge pixels. Since the 2D shape of the object will be extracted from the image, it will only be a projection of the object at a certain time instance and hence we will not be able to use it as the sole model of the object for tracking in successive image frames. This problem can be overcome by updating the

shape after each matching step in order to keep a good track of the projected object shape. In this work we propose to use the method described in [24] where the shape change is modeled explicitly by decomposing the image of a solid object moving in space into a two-dimensional motion and a two dimensional shape change. This method is briefly described below:

1. Let M_0 be the initial model for a 2D shape for an object P. M_0 is a rectangular box consisting of the m_0 edge pixels. The rectangular box is the region detected as violating the optical flow constraint in the moving object detection stage. The 2D shape model is updated after every matching to account for the change in 2D shape of the object due to camera and object motion. Let M_t be the 2D model shape for the object at time t which is a rectangular box consisting of m_t edge pixels obtained by updating the model M_{t-1} at time *t-1*.

2. If g is the equivalent 2D image plane transformation corresponding to the estimated object position and velocity in the world coordinate system and the reported camera position and velocity, then for every edge pixel p in M_t search for an edge pixel q within a distance of d pixels around $g(p)$ in the next image I_{t+1} in the sequence. For a given d, a portion of I_{t+1} is said to match the model M_t if at least K of m pixels of M_t could be matched according to the above definition. Then the best match is the one with minimum d. This matching can be written as

$$\min\left\{ d \left| {}_{K^{th}} \sum_{p \in M_t, q \in I_{t+1}} \left(\|g(p)-q\| \le d \right) \right. \right\} \text{ where } d = 0, 1, 2 \ldots . d_{max} \qquad (8)$$

Ideally if there is no change in the object shape and if g is the correct transformation then the value d_{max} should be zero. However since the object position and velocity estimates and the camera position and motion information are not accurate and also the object shape is changing from

frame to frame, d_{max} is a non-zero value and is a function of error in the estimates and the INS information.

3. Once the object is located in the image I_{t+1}, the object model need to be updated to M_{t+1} by determining which part of I_{t+1} are part of the new model. This is done by using the distance from each point of I_{t+1} to the nearest point of $g(M_t)$ as a criterion for selecting the subset of image points that belong to M_{t+1}. That is we define

$$M_{t+1} = \left\{ q \in I_{t+1} \middle| \min_{p \in M_t} \| g(p) - q \| \leq \delta \right\} \tag{9}$$

for some distance δ. In other words, M_{t+1} is all those points of the image that are within distance δ of some point of $g(M_t)$. The choice of δ controls the degree to which the method is able to track objects that change shape.

Stereo image sequences can be used to solve the uniqueness problem. However, when stereo images are used, additional problems like the correspondence and object/feature tracking in stereo image sequences need to be solved. Since a single camera cannot provide reliable range estimation for objects located near the focus of expansion, Smith et. al. have used a hybrid motion/stereo algorithm for estimating range to stationary objects. In their initial implementation even though the initial range estimates were significantly better in the case of hybrid/motion stereo, the algorithm sometimes produced less accurate results due to problems with feature matching. The problem with feature matching could be due to image noise, differences in the camera themselves and also due to some parallax errors. In this implementation, when stereo match could not be obtained the features were killed and no range estimation was carried out for such features. In a later implementation, to enhance the motion/stereo result the tracking algorithm was modified to allow the range estimates to be propagated based on monocular sequence only, whenever stereo

match could not be obtained. Although we will not be using stereo in this research work, we will explore its potential using the results provided by NASA.

3.2.3 Kalman filter based estimation and prediction

Given the camera's motion parameter and the features' locations in successive images, estimation of the motion parameters of the tracked object/feature is formulated as a state estimation problem using a Kalman filter. We propose to modify the Kalman filter implemented in our earlier work to handle the constraint based solution approach for monocular image sequence. Estimation will be done for every feature and one estimation for every object. No estimation will be continued for individual features that are grouped into objects and the estimation for the entire object will be initialized and updated in the future iterations. However, as it was noted in the earlier section, the initial state vector cannot be computed uniquely from the monocular sequence. We have described above how this problem could be solved by using some constraints or by using a stereo image sequence.

Under constant velocity assumption, it is possible to determine the object path uniquely either by using the knowledge of the terrain in which objects are moving or by using stereo image sequences. In real situations the object velocity might not be constant over a given period of time. In addition, noise in the image coordinates and errors in the camera position and velocity information can lead to considerable error in the estimated position of the object. Hence we also propose to evaluate the performance of the Kalman filter based estimation and tracking considering the noise in the image coordinates, the error in the INS report and deviations from the constant velocity assumption for the moving objects.

3.3 Preliminary Results

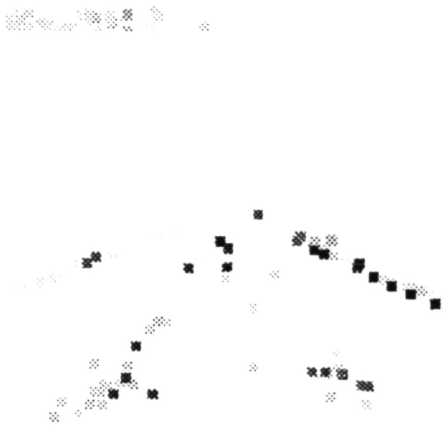
In our earlier work described in section 2, Nelson's constraint was applied on the computed optical flow to detect regions due to moving object. In order to overcome the limitations described in the previous section we have proposed an integrated approach where initial detection will be carried out at the feature level and the features are later grouped to form objects. To distinguish the features due to moving objects from those due to static objects, Nelson's constraint will be applied on the detected features. Many of the features detected at the feature detection stage of the algorithm are due to extraneous objects such as tire marks, runway marks etc. and very few are due to the static or moving obstacles such as trucks. Avoiding such extraneous features could save a lot of computation time. Since tire marks have textural property there is potential for using the texture measure as a means to identifying these features. In the following sections we describe the procedure followed in evaluating the Nelson's constraint for feature-based detection and also use of textures in avoiding the features due to tire marks. Results obtained from NASA image sequences *runway_crossing_new* and *converging_truck_new* are presented.

3.3.1 Applying Nelson's constraint on Feature Based Detection Algorithm:

An attempt was made to apply the Nelson's constraint to the feature detection algorithm of Sridhar et al. The software 'Optflow' from NASA [54, 55], was used to detect features of size 7x7 in the sequence of images. However, instead of tracking the features using this software, they were independently tracked by us. The software gives for each feature a feature number. Features having same feature number in adjacent frames were tracked, as long as the positions in the two consecutive frames had a distance below a threshold. Optical flow for these features was determined by estimating the derivative of the position of the feature. This was done using a derivative of Gaussian mask to provide smoothing. Linear and angular velocities given by the INS

were smoothed using Gaussian mask and parameters of Nelson's constraint were determined in a similar manner to that in section 2. Deviation of the optical flow of the feature from the Nelson's constraint was obtained and thresholded.

(a)



(b)

(c)

(d)

Fig. 18 : (a) 5th image in the sequence *runway_crossing_new* (b) Detected features with gray level corresponding to the deviation from Nelson's constraint (c) Tracked features (d) Features identified as violating the Nelson's constraint.

The above procedure was applied to the sequence of 9 images of a truck crossing the runway (*runway_crossing_new*). The central image is shown in Fig. 18 (a). Detected features are shown in Fig. 18 (b) with their gray level corresponding to the deviation from the Nelson's constraint. These are shown in Fig. 18 (c) as tracked features. The features not satisfying the Nelson's constraint with an appropriate threshold are shown in Fig. 18 (d). It is observed that the feature corresponding to the moving object is detected. However, some features corresponding to tire marks are also detected as false alarms.

3.3.2 Use of Texture for Segmentation

Initial experiment using the NASA's Optflow algorithm showed that many of the detected features in the image sequences tested were mainly due to the tire marks or runway marks and very few features were due to the moving/static truck. Avoiding features due to tire marks can greatly improve the performance of the algorithm. Assuming that the runway is piecewise planar and that the objects of interest are of certain minimum height from the runway plane Sull and Sridhar [71] uses the optical flow method to detect obstacles on the runway.

Since the tire marks on the runway resemble some kind of directional and repetitive flow pattern which is a representative property of textures we make use of texture energy measure to distinguish regions due to object from that due to the tire marks. There has been numerous previous work on the textured image segmentation problem. The texture segmentation approaches are characterized by two main steps: Computing the texture properties, and the segmentation using these properties. Since defining new texture features or developing a special segmentation algorithm is not our main concern, we decided to choose a set of known texture features which can provide us good discriminating power for segmentation and which are easy to compute and a simple segmentation algorithm.

Commonly used texture measures are: Fourier transform domain texture energy, co-occurence matrix, mean and covariance, coarseness, second-order gray level statistic, Gauss-

Markov random field model, and Gibbs random field model. Segmentation algorithms used by various researchers include region growing, clustering and thresholding, and estimation theoretic approaches. We found that Laws' texture energy feature set [72] meets our criteria. The approach requires only a few convolution with small integer coefficient masks, followed by a few moving window absolute average operations. In the following section we briefly describe the Laws' texture energy features and results obtained on the NASA image sequences *runway_crossing_new* and *converging_truck_new* using these features.

Laws' Textured Image Segmentation System:

Laws' approach [72] consists of three steps:

First the image is convolved with a set of filters having a small region of support in the spatial domain. These filters are called the microtexture masks and the filtered outputs are called microtexture features. These masks are combination of the following one dimensional masks:

L5 = [1 4 6 4 1]

E5 = [-1 -2 0 2 1]

S5 = [-1 0 2 0 -1]

W5 = [-1 2 0 -2 1]

R5 = [1 -4 6 -4 1]

Laws' microtexture masks are designed to act as matched filters for certain types of quasiperiodic variations commonly found in textured images. In most cases, the sum of the elements of the mask is zero, which results in the output image having a mean of zero. The convolution masks are intended to be sensitive to visual structure such as edges, ripples and spots.

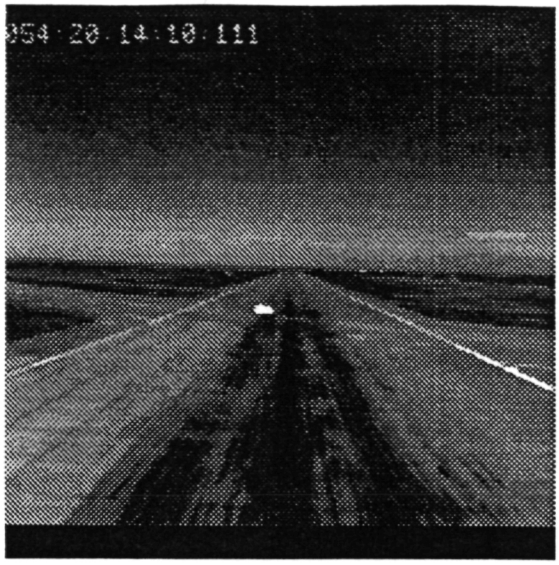
Second, each filtered image is converted to a texture energy image. Local texture energy is measured by the sum of absolute values in a window or local region (e.g. 15x15, 31x31) of the filtered image. It is similar to the local standard deviation if the filtered image is zero-mean. An

optional step is combination of the texture energy measures to form a smaller number of more useful texture measures. Three operations may be used: *normalization* used to adjust for the luminance and contrast of the original image; *rotational averaging* used to account for rotated versions of original texture field; *extraction of principal components* used to reduce the number of features passed to the classifier.

The final operation is classification. The classifier computes the linear discriminant functions from the measured texture energy values for each pixel. The texture class for which the discriminant is greatest is determined to be the source class for that pixel neighborhood.

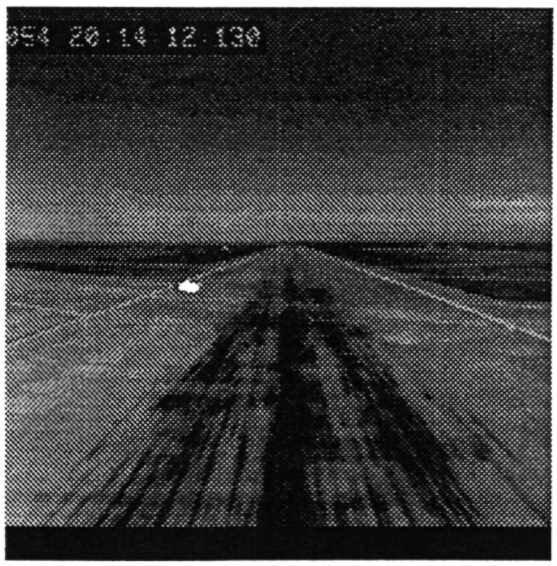
Since the primary objective here is to distinguish regions due to objects from that due to textured background and not multitexture image segmentation, we implemented only the first and second parts of the above algorithm followed by a simple thresholding. A 15x15 size region was used for estimating the texture energy from the filtered images. We have experimented with various combinations of the masks and found that E5L5 performed the best for the two NASA image sequences namely *runway_crossing_new* and *converging_truck_new*. Fig. 19 shows the detected region superimposed on the original image in frames 0, 20, 40 and 80 of the sequence *runway_crossing_new* and Fig. 20 shows the detected region superimposed on the original image in frames 20, 50, 70 and 90 in the sequence *converging_truck_new*. The algorithm did not detect the object in frame 20 in case of *converging_truck_new*. The energy measure obtained from the above algorithm was normalized in the range of 0 to 255 and a threshold of 170 was used in both the cases. Decrease in the threshold was found to detect runway markings and the regions outside the runway and hardly any region on the runway other than the object. Mask S5L5 was found to detect regions inside the runway mainly concentrated around the tire marks. This algorithm runs very fast as convolution operation is separable. The initial results indicate that there is potential for using the texture property for detecting the objects on the runway.

Frame 0



Frame 20

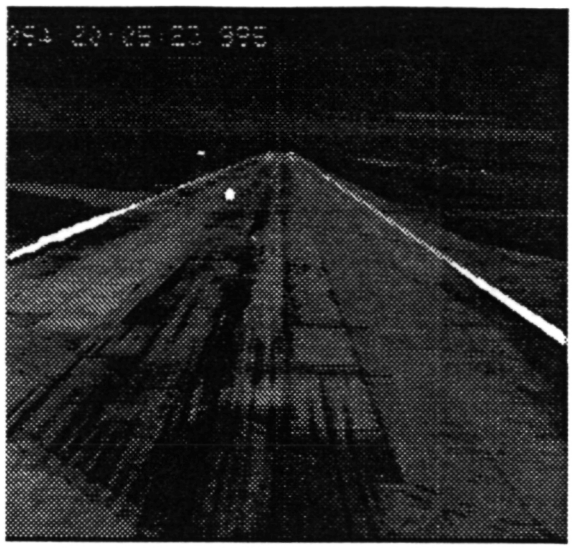
Frame 40



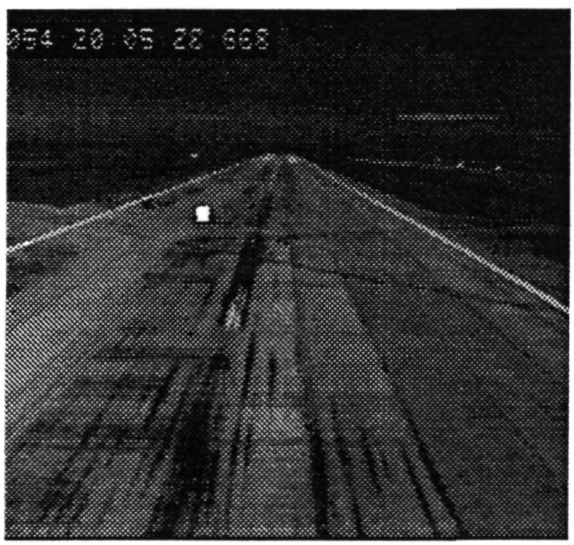
Frame 80

Fig. 19 Detected objects (marked white) superimposed on the original image

Frame 20



Frame 50



Frame 70

Frame 90

Fig. 20 Detected objects (marked white) superimposed on the original image

4. Conclusions

In this report algorithms for detecting moving objects and estimating their position and velocity by analyzing a monocular sequence of images obtained from an on-board camera is described. The algorithm detects the image regions corresponding to the moving objects by applying constraint filtering on the optical flow computed from the initial few frames. The objects are tracked over the entire sequence and the position and velocity of the object in the world coordinate system is estimated using the Kalman filter based recursive estimation procedure. The current implementation has many limitations which are to be solved to complete the design of an usable system. Results from the initial experiments show that the Nelson's constraint can be used to detect features due to moving objects. However many false alarms were produced due to the features resulting from the tire marks. In another experiment, we have shown that texture energy measure can be used to avoid extraneous features resulting from the tire marks.

As part of our on-going research project supported by the NASA Ames Research Center, we have proposed to develop a robust algorithm for detection of moving objects in a sequence of images captured from an on-board video camera and estimation of their position and velocity in the world coordinate system. In particular, we will be exploring solutions to the problem of uniquely estimating the initial position and velocity for the detected moving objects and evaluating the performance of the Kalman filter based tracking in non-ideal conditions such as noise in the image coordinates, non-constant object velocity and errors in the camera position information. Tracking of moving objects in a monocular sequence can have nonunique solutions. We will be exploring two possible alternatives to obtain a solution to this problem: 1) Assume that the objects are moving on known terrain e.g. a flat runway, 2) Use of stereo image sequences. We also propose to explore additional methods for moving object detection and tracking, integration of features into objects, constraint filtering for separation of moving objects from stationary objects and integration

of our algorithm with the passive ranging algorithm developed at the NASA Ames Research Center.

References

[1] Adiv, G., "Determining three-Dimensional Motion and Structure from Optical Flow Generated by Several Moving Objects", *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol. 7, No. 4, pp. 384-401, 1985.

[2] Anandan, P., "A Computational Framework and an Algorithm for the Measurement of Visual Motion," *Int. J. Computer Vision*, vol. 2, pp. 238–310, 1989.

[3] Baker, H. H., and R. C. Bolles, "Generalizing Epipolar-Plane Image Analysis on the Spatio-temporal Surface," *Int. J. Computer Vision*, Vol. 3, pp. 33–49, 1989.

[4] Barron, J. L., D. J. Fleet, S.S. Beauchemin, and T. A. Burkitt, "Performance of Optical Flow Techniques," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 236–242, 1992.

[5] Bhanu, B., P. Symosek, J. Ming, W. Berger, H. Nasr, and J. Kim, "Qualitative Motion Detection and Tracking," *DARPA Image Understanding Workshop*, pp. 370–398, 1989.

[6] Balck, M. J., "Combining Intensity and Motion for Incremental Segmentation and Tracking over Long Image Sequences", *Proc. 2nd European Conference on Computer Vision*, pp. 485-493, 1992.

[7] Bolles, R. C, H. H. Baker, and D. H. Marimont, "Epipolar-Plane Image Analysis: An Approach to Determining Structure from Motion," *Int. J. Computer Vision*, Vol. 1, pp. 7–55, 1987.

[8] Bouthemy, P., and P. Lalande, "Motion Detection in an Image Sequence Using Gibbs Distributions," *Int. Conf. Acoustics, Speech, and Signal Processing*, pp. 1651–1654, 1989.

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

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

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

[12] Chow, W., and J. K. Aggarwal, "Computer Analysis of Planar Curvilinear Moving Images," *IEEE Trans. Computers*, vol. 26, pp. 179–185, 1977.

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

[14] Debrunner, C., and N. Ahuja, "Motion and Structure Factorization and Segmentation of Long Multiple Motion Image Sequences", *Proc. 2nd European Conf. on Computer Vision*, pp. 217-221, 1992.

[15] Duncan, J. H., and T. C. Chou, "On the Detection of Motion and the Computation of Optical Flow," *IEEE Trans. Pattern Analysis and Machine Intelligence*," vol. 14, no. 3, pp. 346–352, 1992.

[16] Feddema, J. T., and C. S. G. Lee, "Adaptive Image Feature Prediction and Control for Visual Tracking with a Hand-Eye Coordinated Camera," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 20, no. 5, pp. 1172–1183, 1990.

[17] Fleet, D. J., and A. D. Jepson, "Computation of Component Image Velocity from Local Phase Information," *Int. J. Computer Vision*, vol. 5, pp. 77–104, 1990.

[18] Gennery, D. B., "Visual Tracking of Known Three-Dimensional Objects," *Int. J. Computer Vision*, vol. 7, no. 3, pp. 243–270, 1992.

[19] Haim, S., "Detecting Motion in Out-of-Register Pictures," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 696–699, 1988.

[20] Haynes, A. S., and R. Jain, "Time-Varying Edge Detection," *Computer Vision, Graphics, and Image Processing*, vol. 21, pp. 345–367, 1983.

[21] Heeger, D. J., "Optical Flow Using Spatiotemporal Filters," *Int. J. Computer Vision*, vol. 1, pp. 279–302, 1988.

[22] Horn, B. K. P. and B. G. Schunk, "Determining Optical Flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

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

[24] Huttenlocher, D. P., J. J. Noh, and W. J. Rucklidge, "Tracking Non-Rigid Objects in Complex Scenes," *Int. Conf. Computer Vision*, pp. 93–101, 1993.

[25] Jain, R. D. Militzer, and H.-H. Nagel, "Separating Non-Stationary from Stationary Scene Components in a Sequence of Real World TV Images," *Int. Joint Conf. Artificial Intelligence*, pp. 425–428, 1977.

[26] Jain, A. K., and H. H. Nagel, "On the Analysis of Accumulative Difference Pictures from Image Sequences of Real World Scenes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 1, pp. 206–214, 1979.

[27] Jain, R., "Dynamic Scene Analysis Using Pixel-Based Processes," *IEEE Computer*, vol. 14, pp. 12–18., 1981.

[28] Jain, R. C., "Segmentation of Frame Sequences Obtained by a Moving Observer," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, no. 5, pp. 624–629., 1984.

[29] Koller, D. K. Danilidis and H. H. Nagel, "Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes," *Int. J. Computer Vision*, vol. 10, no. 3, pp. 257–281, 1993.

[30] Liu, Y., and T. S. Huang, "A Linear Algorithm for Motion Estimation using Straight Line Correspondences", *Computer Vision, Graphics and Image Processing*, vol. 44, pp 35-37, 1988.

[31] Longuet-Higgins, H. C., and K. Prazdny, "The Interpretation of a Moving Retinal Image," *Proc. Royal Society of London B*, vol. 208, pp. 385–397, 1980.

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

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

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

[35] Mitiche, A., O. Faugera and J. K. Agarwal, "Counting Straight Lines", *Computer Vision, Graphics and Image Processing*", vol. 47, pp. 353-360, 1989.

[36] Mygret A., and M. Thonnat "Segmentation of Optical Flow and 3D Data for the Interpretation of Mobile Objects", *Proc. 3rd Int. conf. on Computer Vision* pp. 238-245, 1990.

[37] Nagel, H. H., "On the Estimation of Optical Flow: Relations Between Different Approaches and Some New Results," *Artificial Intelligence*, vol. 33, pp. 299–324, 1987.

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

[39] Papanikolopoulos, N. P., P. K. Khosla and T. Kanade, "Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combination of Control and Vision," *IEEE Trans. Robotics and Automation*, vol. 9, no. 1, pp. 14–35, 1993.

[40] Rashid, R. F., "Toward a System for the Interpretation of Moving Light Display," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-2, pp. 574–581, 1980.

[41] Roach, J. W., and J. K. Aggarwal, "Determining the Movement of Objects from a Sequence of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-2, pp. 554–562, 1980.

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

[43] Rognone A., M. Campani and A. Verri, "Identifying Multiple Motions from Optical Flow", *Proc. 2nd European Conf. on Computer Vision*, pp 258-266, 1992.

[44] Sabri, A. M., "Motion Detection and Estimation of Multiple Moving Objects in an Image Sequence Using the Cosine Area Transform (CAT)," *IEEE Proc. Part I: Communications, Speech, and Vision*, pp. 351–356, 1991.

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

[46] Sethi, I. K. and R. Jain, "Finding Trajectories of Feature Points in a Monocular Image Sequence," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 1, pp. 56–73, 1987.

[47] Silvén, O. and T. Repo, "Experiments with Monocular Visual Tracking and Environment Modeling," *Int. Conf. Computer Vision*, pp. 84–92, 1992.

[48] Singh, A., "An Estimation-Theoretic Framework for Image-Flow Computation," *Int. Conf. Computer Vision*, pp. 168–177, 1990.

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

[50] Smith P. N., Sridhar B. and Hussien B., "Vision-based Range Estimation using Helicopter Flight Data", *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 202-208, 1992.

[51] Spoerri A. and S. Ullman "The Early Detection of Motion Boundaries", *Proc. 1st Int. Conf. on Computer Vision*, pp. 209-218, 1987.

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

[53] Sridhar B. and Pathak A. V., "Analysis of Image-Based Navigation System for Rotorcraft Low-Altitude Flight", *IEEE Transaction on Systems, Man and Cybernetics*, vol. 22, no. 2, March/April 1992.

[54] Sridhar B., Suorsa R. and Hussien B., "Passive Range Estimation for Rotorcraft Low-Altitude Flight", *Machine Vision and Applications*, vol. 6, pp. 10-24, 1993.

[55] Suorsa, R. E., and B. Sridhar, "A Parallel Implementation of a Multisensor Feature-Based Range-Estimation Method", *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 379-385, 1983.

[56] Tang, Y. L., "An Airborne Vision System for Runway Recognition and Obstacle Detection", *Ph. D. dissertation*, August 1994.

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

[58] Tang, Y. L., and R. Kasturi, "Accurate Estimation of Object Location in an Image Sequence using helicopter flight data", *Journal of Robotics and Computer Integrated Manufacturing*, vol. 11, No. 4, 1995.

[59] Tang, Y. L., and R. Kasturi, "Runway Recognition in an Image Sequence", *Proc. IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, vol. 2421, 1994.

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

[61] Tang, Y. L., S. Devadiga, R. Kasturi. and R. Harris Sr., "Model-based Approach for Detection of Objects in Low Resolution Passive-Millimeter Wave Images", *Proc. IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, vol. 2182, pp. 320-330, 1994.

[62] Thompson, W. B. and T. C. Pong, "Detecting Moving Objects," *Int. J. Computer Vision*, vol. 4, pp. 39–57.

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

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

[65] Uras, S., F. Girosi, A. Verri, and V. Torre, "A Computational Approach to Motion Perception," *Biological Cybernetics*, vol. 60, pp. 79–97, 1988.

[66] Waxman, A. M., J. Wu and F. Bergholm, "Convected Activation Profiles and Receptive Fields for Real Time Measurement of Short Visual Motion," *IEEE Conf. Computer Vision and Pattern Recognition*, pp. 717–723, 1988.

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

[68] Wu, J. J., R. E. Rink, T. M. Caelli, and V. G. Gourishankar, "Recovery of the 3-D Location and Motion of a Rigid Object Through Camera Image (An Extended Kalman Filter Approach)," *Int. J. Computer Vision*, vol. 3, pp. 373–394, 1988.

[69] Zhang, Z. and O. D. Faugeras, "Three-Dimensional Motion Computation and Object Segmentation in a Long Sequence of Stereo Frames," *Int. J. Computer Vision*, vol. 7, no. 3, pp. 211–241, 1992.

[70] Zheng, Q. and R. Chellappa, "A Computational Vision Approach to Image Registration," *IEEE Trans. Image Processing*, vol. 2, no. 3, pp. 335–339, 1993.

[71] Sull, S. and B. Sridhar, "Model-based Obstacle Detection from Image Sequences", to appear in *Int. Conf. on Image Processing*, 1995.

[72] Laws, K. I., "Rapid Texture Identification", *Proc. SPIE* Vol. 238, pp. 376-380, 1980.