Real-Time Visual Tracking Using Image Processing and Filtering Methods

A Thesis
Presented to
The Academic Faculty

by

Jin-cheol Ha

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Aerospace Engineering
Georgia Institute of Technology
April 1, 2008

Real-Time Visual Tracking Using Image Processing and Filtering Methods

Approved by:

Date Approved: March 28, 2008

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Glossary

$\mathbf{R}$	the set of real numbers.
$\mathbf{R}^n$	the vector space of n-tuples of real numbers.
$\mathbf{x}_k$	the state vector at time step k.
$\mathbf{y}_k$	the measurement vector at time step k.
$\|\|\mathbf{x}\|\|$	the vector norm.
$\mathbf{z} = [x, y]^T$	a position vector in $\mathbf{R}^2$.
I	an image.
Δ	the Laplace operator.
∇	the gradient operator.
$\mathcal{N}$	the inward unit normal.
$H(\cdot)$	the heavy side function.
$\Psi\ (\mathbf{z}, t)$	the continuous higher-dimensional functional.
$\Psi^s\ (\mathbf{z}, t)$	the discrete higher-dimensional functional.
C	a curve.
$\widetilde{C}$	a predicted curve.
c_i, c_o	the average value of features inside (outside) the curve C.
E_a	the energy function related to a.
$p\ (\ a \mid b\ )$	the conditional probability of a given b.
ω_k^i	the normalized weight for the i-th sample at time step k.
π_m	the weight of the m-th component in the multiple target tracking.

SUMMARY

The main goal of this thesis is to develop real-time computer vision algorithms in order to detect and to track targets in uncertain complex environments purely based on a visual sensor. Two major subjects addressed by this work are:

1. The development of fast and robust image segmentation algorithms that are able to search and automatically detect targets in a given image.

2. The development of sound filtering algorithms to reduce the effects of noise in signals from the image processing.

The main constraint of this research is that the algorithms should work in real-time with limited computing power on an onboard computer in an aircraft. In particular, we focus on contour tracking which tracks the outline of the target represented by contours in the image plane. This thesis is concerned with three specific categories, namely image segmentation, shape modeling, and signal filtering.

We have designed image segmentation algorithms based on geometric active contours implemented via level set methods. Geometric active contours are deformable contours that automatically track the outlines of objects in images. In this approach, the contour in the image plane is represented as the zero-level set of a higher dimensional function. (One example of the higher dimensional function is a three-dimensional surface for a two-dimensional contour.) This approach handles the topological changes (e.g., merging, splitting) of the contour naturally. Although geometric active contours prevail in many fields of computer vision, they suffer from the high computational costs associated with level set methods. Therefore, simplified versions of level set methods such as fast marching methods are often used in problems of real-time visual tracking. This thesis presents the development of a fast and robust segmentation algorithm based on up-to-date extensions of level set methods and geometric active contours, namely a fast implementation of Chan-Vese's (active contour) model (FICVM).

The shape prior is a useful cue in the recognition of the true target. For the contour tracker, the outline of the target can be easily disrupted by noise. In geometric active contours, to cope with deviations from the true outline of the target, a higher dimensional function is constructed based on the shape prior, and the contour tracks the outline of an object by considering the difference between the higher dimensional functions obtained from the shape prior and from a measurement in a given image. The higher dimensional function is often a distance map which requires high computational costs for construction. This thesis focuses on the extraction of shape information from only the zero-level set of the higher dimensional function. This strategy compensates for inaccuracies in the calculation of the shape difference that occur when a simplified higher dimensional function is used. This is named as contour-based shape modeling.

Filtering is an essential element in tracking problems because of the presence of noise in system models and measurements. The well-known Kalman filter provides an exact solution only for problems which have linear models and Gaussian distributions (linear/Gaussian problems). For nonlinear/non-Gaussian problems, particle filters have received much attention in recent years. Particle filtering is useful in the approximation of complicated posterior probability distribution functions. However, the computational burden of particle filtering prevents it from performing at full capacity in real-time applications. This thesis concentrates on improving the processing time of particle filtering for real-time applications.

In principle, we follow the particle filter in the geometric active contour framework. This thesis proposes an advanced blob tracking scheme in which a blob contains shape prior information of the target. This scheme simplifies the sampling process and quickly suggests the samples which have a high probability of being the target. Only for these samples is the contour tracking algorithm applied to obtain a more detailed state estimate. Curve evolution in the contour tracking is realized by the FICVM. The dissimilarity measure is calculated by the contour based shape modeling method and the shape prior is updated when it satisfies certain conditions. The new particle filter is applied to the problems of low contrast and severe daylight conditions, to cluttered environments, and to the appearing/disappearing target tracking. We have also demonstrated the utility of the filtering algorithm for multiple target tracking in the presence of occlusions. This thesis presents several test results from simulations and flight tests. In these tests, the proposed algorithms demonstrated promising results in varied situations of tracking.

CHAPTER I

INTRODUCTION

1.1 Active Vision

Vision is the primary sensing mechanism for humans [1]. It is the main source of information of our surroundings. Considering the amount of data obtained from vision, the data-processing ability of the human visual system is remarkable. In fact, the human brain interprets only a small fraction of data and saves them [2]. The ability of the human vision system in sensing and processing data enables us to interact intelligently with our environment.

Since the advent of the digital computer, several research efforts have attempted to give machines the same ability as the human vision system. This research is referred to "machine vision." This term is used to describe any mechanism that utilizes vision as the main sensor for any machine that works in real-time [3]. Most of the progress in machine vision has resulted in applications in industries in which machines perform repetitive tasks with a limited range of well-defined objects. Prominent applications of machine vision are seen in the large-scale industrial manufacturing, inspection, and the guidance of robot arms [1, 2, 4].

As described in two seminal papers [5, 6], machine vision may be classified into two categories: passive and active vision. In passive vision, an observer senses a scene in space and time, determined by known visual parameters and environmental conditions. In active vision, an observer is able to manipulate its visual parameters in a controlled manner in order to extract useful data about the scene in space and time that would allow it to best perform a set of tasks [6]. Active vision systems are designed to take actions according to the data obtained from the vision sensor.

These actions determine what kind of information is going to be extracted from the image. Prominent applications of active vision systems include robot control [8, 9, 12], unmanned vehicle navigation [13, 23], and visual surveillance [10, 15]. In this thesis, an unmanned aerial vehicle (UAV) is considered as an active vision control system that navigates and determines control actions based on information obtained by visual surveillance and tracking of targets.

1.2 Vision and UAVs

Many in-flight tasks such as intelligence, surveillance, reconnaissance, and even combat missions that have traditionally performed by manned aircraft, are now being performed by unmanned aerial vehicles (UAVs). A UAV is defined as a powered aerial vehicle that does not carry a human operator, uses aerodynamic forces to provide vehicle lift, can fly autonomously or be piloted remotely [16]. UAVs have demonstrated effective performances in tedious, dirty, or potentially hazardous tasks that would otherwise be impossible or exceptionally dangerous for human pilots.

To further extend the fundamental advantages offered by UAVs, current research has been directed towards enabling UAVs to operate in unfavorable conditions, such as at low altitudes and in congested and obstacle-rich environments. Rich information (e.g., color, texture, shape) of the environment surrounding the UAV can be provided by a vision system. Utilizing vision sensors that derive the measurements of such relative states has gained a lot of attention in recent years due to the many advantages offered by these vision sensors, namely, low-cost, light weight, and available commercially off-the-shelf. The vision system provides the navigation system with the measurements of relative states of targets or obstacles, which are indispensable elements in several UAV missions such as target tracking, obstacle avoidance, and so on.

Navigating an uncertain environment and executing tasks during flight using only a vision sensor is a challenging task for an unmanned aerial vehicle (UAV) due to variations in illumination, and the possible presence of clutter and obstacles. In such unfavorable conditions, the vision system on the UAV must interpret the data from the sensor in order to derive meaningful information since the navigation and control systems of the UAV will make decisions based on this information.

UAVs have employed vision in various applications. Vision sensors have been used for the path-planning of a UAV in low-altitude navigation [17] and the estimation of relative positions for landing in conjunction with other sensors [18]. See-and-avoid system for a UAV is designed by using several commercial vision libraries [87]. In [19, 20], a flight control system was developed for the navigation of a glider using only vision. The glider was guided to a fixed target using a camera as the sole sensor. An unmanned helicopter succeeded in autonomous visual searching and a formation flight with another UAV with a single camera as the vision sensor [22, 23].

1.3 Computer Vision

The vision system processes the image data from the vision sensor in a computer via certain computer vision algorithms. Two basic elements of such algorithms are image processing and filtering. Image processing algorithms extract useful data from the given image, and filtering algorithms reduce the effects of noise on them by considering results from multiple frames and process modeling. Image processing methods applied in active vision can be found in many papers. Many common methods are edge detection, thresholding, template matching, optical flow, deformable contours, and stereo vision [7, 9, 10, 11, 12, 13, 14]. As the demands of active vision systems become more and more complicated, one requires faster and more robust image processing algorithms.

Even though useful data may be obtained from images, the signals have to be first filtered to reduce the effect of noise. The well-known filter is the Kalman filter which has been used in numerous applications because of its simplicity and ease of implementation in a digital computer [50, 51]. The Kalman filter provides an exact solution only for problems which have linear models and Gaussian distributions (linear/Gaussian problems). However, many dynamical systems encountered in practice are nonlinear and/or non-Gaussian (nonlinear/non-Gaussian) problems. As such, the extended Kalman filter utilizes the linearization of a given nonlinear model. However, if the initial guess is poor or if disturbances are very large, the filter may diverge. For nonlinear/non-Gaussian problems, particle filtering has received much attention in recent years. Particle filters are sequential Monte Carlo methods that can approximate the solutions of any nonlinear/non-Gaussian dynamical models.

1.4 Scope

One of the fundamental tasks of a vision system is visual target tracking. Visual tracking is the recursive processing of measurements of a target obtained from a set of vision sensors in order to predict and obtain estimates of the target states [25]. This thesis is concerned with visual tracking problems for the navigation and control of UAVs. The purpose of our research is to develop real-time computer vision algorithms in order to track targets in uncertain complex environments purely based on a visual sensor. Two major subjects of this work are:

1. The development of fast and robust image processing algorithms that are able to search and automatically detect aerial targets in a given image.

2. The development of sound filtering algorithms to reduce the effects of noise in signals from the image processing.

The main constraint of this research is that the algorithms should work in real-time with limited computing power on the onboard computer in a UAV. This thesis concentrates on fast and accurate state estimation by using the measurements from a two-dimensional image. In particular, we focus on contour tracking which tracks the outline of the target which is represented as a contour in an image plane.

During the course of tracking, the outline of the target undergoes various uncertain situations in the image plane (e.g., occlusions, diffusions, and clutters) that make it hard to infer which one belongs to the true target. Furthermore, there are many nonlinear/non-Gaussian problems in which the existing Kalman filter is hard to apply such as tracking multiple targets and tracking targets in the presence of clutters. This thesis incorporates the contour tracker with some knowledge of the target's shape along with particle filters to aid in tracking with the aforementioned complicated problems. In particular, we discuss various cases of visual tracking including tracking in a cluttered environment, tracking multiple targets, and tracking appearing/disappering targets.

There are no limitations on the flight situation of the targets (maneuvers, altitude changes). This thesis considers only gray-scale intensity images since color information may not be reliable in our applications. The color information in images is unavailable for two reasons: the target is often shaded from sunlight, and the target is too small and thin to be recognized by its color. The tracker in the follower airplane may recognize the color information of the target when the target moves close to it. However, this is not desirable situation since the risk of collision increases with the decreased range.

1.5 Contour Tracker

This thesis is concerned primarily with the specific aspects of the contour tracker, namely the fast implementation of geometric active contours, contour-based shape modeling, and particle filtering in the geometric active contour framework.

Shape information inferred from the outline of the target may play an important role in recognizing the correct target. A tracker which detects and tracks the outline of the target using a deformable contour is defined as a contour tracker. The contour should be represented by a certain data type to be used in computer vision. Often, the contour is represented by sets of explicit points or splines in computer vision [26, 7, 94, 91, 93, 92]. The most prominent contour tracker proposed in [24] shows robust results for target tracking in a cluttered environment. The contour in this approach is represented by B-splines and the state space consists of global motion parameters (affine motion parameters). However, this approach can not handle the topological changes (e.g., merging and splitting) of the contour.

On the other hand, geometric active contours using an implicit representation of the contour has gained much attention in recent years and remarkable progress has been made. Geometric active contours are deformable contours that automatically track the outlines of objects in images [27, 28, 29]. In this approach, the contour in the image plane is represented as the zero-level set of a higher dimensional function. (One example of the higher dimensional function is a three-dimensional surface for a two-dimensional contour.) This approach naturally handles the topological changes of the contour.

Although geometric active contours prevail in many fields of computer vision, they suffer from the high computational costs associated with level set methods. Therefore, simplified versions of level set methods, such as fast marching methods, are often used in problems that require a fast update rate of image processing [31, 33]. However, fast marching methods are sensitive to noise since they rely on high-gradients of intensity or edges that are sensitive to noise. Region-based level set methods that utilize global information from a given image may increase the robustness to noise [40, 42, 36]. In particular, the Chan-Vese's region-based active contour model [36] is widely used, and several extensions of the algorithm are proposed to speed up the segmentation process [71, 37, 73]. Song and Chan [37] proposed a unique optimization method for the Chan-Vese model which drastically increases the speed of the algorithm (see 3.2 for details).

In the mean time, Shi and Karl [44] proposed a fast level set implementation in which the contour moves at the pixel resolution (see 3.1 for details). This thesis combines a unique optimization method with a fast level set implementation, and proposes a fast and robust segmentation algorithm. This is called as the fast implementation of the Chan-Vese model (FICVM).

Prior knowledge about the target's shape (shape prior) is a useful cue in the recognition of the true target since the outline of the target can be easily disrupted by noise (e.g., diffusion, occlusion, and clutter). In some existing geometric active contours, to cope with deviations from the true outline of the target, the higher dimensional function is constructed based on the shape prior, and the contour tracks the outline of an object by considering the difference between the higher dimensional functions obtained from the shape prior and from a measurement in a given image [75, 76, 77, 99, 98]. However, these methods are hard to apply to our real-time applications because the higher dimensional function often requires high computational costs for construction (e.g., distance map from the contour). Instead of constructing a complicated higher dimensional function, this thesis focuses on the extraction of shape information from only the zero-level set of the higher dimensional function, and proposes a novel method which can quickly detect multiple objects in a given image. Furthermore, this method can compensate some inaccuracy caused by using a simple higher dimensional function in the calculation of the shape difference (the FICVM uses a simplified discrete higher dimensional function). This is named as contour-based shape modeling.

The target's shape may change over time, and the presence of noise alters the outline of the target during tracking. Filters may reduce the effect of noise and establishes temporal coherence of the shape. In [7, 45] Kalman filtering is incorporated with explicit contour representation. However, the Kalman filters can not handle nonlinear/non-Gaussian problems (e.g., multiple observed points, occlusions) and topological changes of the contour. Particle filters can can overcome these limitations of the Kalman filter. In particular, the authors in [24] estimated the global motion of the target's shape and demonstrated robust results for tracking in clutters by using a particle filter. (Shape information may be divided into global motion and local deformation [88, 89].) However, global motion requires parametrization of the entire motion of the shape so that it may not account for local deformation of the target. In the mean time, the estimation of local deformation is not a trivial task since local deformation can be obtained by (theoretically) infinitely many choices [89, 21]. In [81], the authors proposed a particle filter in the geometric active contour framework which can estimate local deformation as well as global motion of the shape. Since it is based on geometric active contours, the topological change of the target can be handled. This thesis barrows their framework and improves the computational speeds of the algorithm by applying the FICVM and contour-based shape information. For further improving processing time, blob tracking is combined with contour tracking. The blob representation fits well with particle filtering since a blob is easy to model and can quickly localize a target [101, 64, 102]. This thesis defines a blob based on the shape prior, and simplifies the prediction step by using it. This thesis discusses solutions for a broad range of real-time tracking problems faced in our applications.

1.6 Contributions

The main subjects of this thesis can be classified into three categories which are summarized below and are described in greater detail in subsequent chapters. The contributions for each subject are addressed as follows:

A fast implementation of geometric active contours (FICVM): A fast implementation of the Chan-Vese model improves the speed of the segmentation process while maintaining robustness to local minima. These improvements are obtained by the combination of a unique optimization method [37] and a fast level set implementation [44]. These methods both uses simple discrete higher dimensional functions instead of using computationally expensive continuous distance map of level set methods. These simple structures are beneficial for using multiple initial contours without significantly increasing computational time, and for reducing the computational time of particle filters. Unlike fast marching methods, this method is well-suited for the temporal coherence of a target's motion, including shape changes, by using the contour information in the previous frames. Refer to Chapter 3 for details of the new fast segmentation method.

A contour-based shape modeling: this thesis proposes a novel shape modeling that considers only the contours in the image plane (the zero-level sets of the higher dimensional function). A novel method for calculating the difference between two contours is proposed, and it can quickly identify multiple objects of interest in a given image. Furthermore, this method can compensate some inaccuracy caused by using FICVM in the calculation of the shape difference between two higher dimensional functions. Refer to Chapter 4 for details of the novel shape modeling scheme.

Particle filtering in the geometric active contour framework: The main contribution of this thesis in particle filtering is the improvement of the computational time while enhancing the ability of tracking in varied conditions. In principle, we follow the particle filtering in the geometric active contour framework [81]. In addition, this thesis proposes an advanced blob tracking scheme in which a blob contains the shape prior of the target. This scheme simplifies the sampling process and quickly suggests the samples which have a high probability of being the target. Only for these samples is the contour tracking algorithm applied in order to obtain a more detailed state estimate. The FICVM and the contour-based shape modeling method are incorporated with the particle filter. Shape information of the target is updated when it satisfies certain conditions.

We have tested the algorithm in various problems faced in our visual tracking problems and have demonstrated the utility of particle filters. The new particle filter is applied to the problems of low contrast and severe sunlight conditions, to cluttered environments, and to the appearing/disappearing target tracking. We have also demonstrated the utility of the filtering algorithm for multiple target tracking in the presence of occlusions. Several promising test results from simulations and flight tests are presented. Refer to Chapter 5 for details of the particle filters.

1.7 Outline of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 provides a brief overview of the geometric active contours, and Chapter 3 presents the development of the FICVM. Chapter 4 presents a novel approach for contour-based shape modeling. Chapter 5 presents a brief overview of particle filtering. Chapter 6 proposes a particle filter by combining blob tracking and contour tracking. The FICVM and the contour-based shape modeling are applied to the particle filtering in the geometric active contour framework. We discuss the particle filtering scheme for various examples in our applications. Chapter 7 provides detection and tracking strategies based on image processing and particle filtering. Finally, Chapter 8 presents conclusions and possible future research.

CHAPTER II

GEOMETRIC ACTIVE CONTOURS

This chapter briefly reviews geometric active contours. Section 2.1 overviews the background of geometric active contours. Section 2.2 describes geometric active contour model. Section 2.3 describes region-based geometric active contours. Finally, Section 2.4 reviews fast marching methods.

2.1 Background of Geometric Active Contours

Snakes, or active contours, are deformable contours that automatically track various features of interest in an image. The original active contour model is based on an energy minimization approach proposed by Kass *et al.* [26]. In [26], active contours evolve in a way that minimize a given energy functional combining internal and external energy. Internal energy is designed to represent energy from the smoothness of the contour, and external energy is associated with image features such as intensity-based edges in the image. In the original model, the contour, as the form of a closed curve in the image plane, is parameterized with a set of discrete nodes, or the Lagrangian formulation. This formulation contains problems in the evolution of the curve. First, when nodes come close, the calculation of a curvature term yields large errors. Second, topological changes such as splitting or merging are very difficult. In addition, initial nodes should be placed close to the interesting features in the image [28].

Caselles *et al.* [27] and Malladi *et al.* [28] proposed geometric active contours based on level set methods. In the level set approach, a curve in an image plane is represented as the zero level set of a higher-dimensional surface, or the Eulerian formulation. By employing level set methods, geometric active contours solve such problems addressed in the classical active contours and are able to handle topological changes automatically. In geometric active contours, curve evolution is based on a geometric partial differential equation (PDE). Kichenassamy *et al.* [29, 30] proposed geodesic active contours, which are based on the generalizations of the curve shortening flow to conformally Euclidean metrics.

Level set methods were invented by Osher and Sethian and are based on implicit representations of the evolving contour; see [35] and the references therein. In image processing, level set methods are used in conjunction with information from the given image. According to the type of information, geometric active contours may be classified into several categories including edge-based approaches [27, 28] and region-based approaches [40, 42]. While edge-based approaches use local information (e.g, intensity, intensity-based edge), region-based approaches use global information (e.g., average intensity values of segments) in an image. Region-based approaches yield more robust results to local minima than edge-based approaches. Furthermore, initial contours can be placed anywhere in the image in region-based approaches. In particular, the Chan-Vese region-based active contour model [36], based on level set methods, the Mumford-Shah functional for segmentation, and curve evolution theory, is widely used.

Although geometric active contours prevail in many fields of computer vision, they suffer from high computational costs of level set methods. In two-dimensional image processing problems, level set methods require that a three-dimensional functional be constructed, so these methods are computationally expensive. As a result, some applications that require a fast update rate of image processing are reluctant to use them. One solution for this problem is to use a narrow band method [34, 35] in which a curve only evolves in a given band width, and the curve is re-initialized when the curve reaches the boundary of the narrow band. Another alternative numerical technique is the fast marching method. The fast marching method is an extreme case of the narrow band method with the one-pixel bandwidth, and a curve move in only one direction in these methods [31, 33]. Therefore, fast marching methods are used for applications that require fast image segmentation [80, 96, 95].

Recently, Shi and Karl [44] proposed a fast level set implementation that leads to curve evolution at a pixel resolution. Furthermore, several extensions of the Chan-Vese model reported tremendous computational speeds fast enough to cover a broad range of real-time applications [71, 37, 73]. In general, these extensions increase the speed of the model by not solving partial differential equations which are bounded to stability constraints [37]. They apply smoothing processes periodically to ensure the smoothness of the evolving contour and ignore the length-minimizing term in the Chan-Vese model [71, 73, 44].

The following sections briefly review the theories of geometric active contours based on the curve evolution theory and level set methods, region-based approaches, and fast marching methods.

2.2 Geometric Active Contours and Curve Evolution

Geometric active contours are introduced in [27, 28]. In these models, the active contour is represented as the zero-level set of a higher-dimensional functional, and the evolution of the contour is based on geometric PDEs. In [29], the authors interpreted the models in a conformal curve shortening point of view. This section briefly reviews geometric active contour models based on the curve evolution theory and level set methods.

Let $C = C(p, t)$ be a smooth family of closed curves, i. e., $C(0, t) = C(1, t)$, where t parameterizes the family and p the given curve, say $0 \leq p \leq 1$. Consider the length functional

$$L(t) = \int_0^1 \|\frac{\partial C}{\partial p}\| \, dp \; . \tag{1}$$

Using the calculus of variation, the corresponding Euler-Lagrange equation is

$$\frac{\partial C}{\partial t} = \kappa \mathcal{N} \; , \tag{2}$$

where C is a planar curve in $\mathbf{R}^2$, κ is the curvature, and $\mathcal{N}$ is the inward unit normal [29]. Equation (2) is well-known geometric heat equation. Thus, the planar curve C evolves according to equation (2).

In level set methods, a planar curve is represented by the zero-level set of a higher-dimensional function Ψ [28]. Let initial surfaces $\Psi(\mathbf{z}, t = 0)$, where $\mathbf{z} \in \mathbf{R}^2$, be defined as

$$\Psi(\mathbf{z}, t = 0) = \pm d \; , \tag{3}$$

where d is the minimum distance from $\mathbf{z}$ to the plane curve. The zero level set is $\Psi(\mathbf{z}, t) = 0$. If equation (3) is differentiated with respect to t, then, by chain rule

$$\Psi_t + (\Psi_{\mathbf{z}_1}, \Psi_{\mathbf{z}_2}, ..., \Psi_{\mathbf{z}_N}) \cdot (\mathbf{z}_{1_t}, \mathbf{z}_{2_t}, ..., \mathbf{z}_{N_t}) = 0 \; . \tag{4}$$

Then, Ψ_t becomes

$$\begin{aligned} \Psi_t &= < \nabla\Psi, \; \frac{\partial C}{\partial t} > \\ &= < \nabla\Psi, \; \kappa\mathcal{N} > \\ &= \kappa < \nabla\Psi, \; \frac{\nabla\Psi}{\|\nabla\Psi\|} > \\ &= \kappa\|\nabla\Psi\| \; . \end{aligned} \tag{5}$$

Since $\kappa = \mathtt{div}(\frac{\nabla\Psi}{\|\nabla\Psi\|})$ [27], we have the curve evolution equation in the level set formulation

$$\frac{\partial\Psi}{\partial t} = \|\nabla\Psi\| \; \mathtt{div}(\frac{\nabla\Psi}{\|\nabla\Psi\|}) \; . \tag{6}$$

To incorporate with information from an image, consider a function $\phi(x, y)$ which depends on the given image, and is used as a "stopping term." Typically, the term $\phi(x, y)$ is chosen to be small near an intensity-based edge, and acts to stop evolution when the contour gets close to an edge. One may define

$$\phi(x, y) = \frac{1}{1 + \|\nabla G_\sigma * I\|^2} \, , \tag{7}$$

where $I(x, y)$ is a given gray-scale input image, and G_σ is a Gaussian smoothing filter [27, 28]. Then the final equation is [29],

$$\frac{\partial \Psi}{\partial t} = \phi(x, y)\|\nabla \Psi\|(\, \mathtt{div}(\frac{\nabla \Psi}{\|\nabla \Psi\|}) + \nu) + \nabla \phi \cdot \nabla \Psi \, , \tag{8}$$

where a constant inflation term, ν is added as in [27, 28]. The function $\Psi(x, y, t)$ in equation (8) evolves according to the associated level set flow for planar curve evolution in the normal direction with speed as a function of curvature.

2.3 Region-Based Geometric Active Contours

The idea of region-based geometric active contours is to minimize an energy functional based on features (e.g., means, variances) representing each segment of an image. In these models, curve evolution equations are designed to separate values of these features as far as possible and also to satisfy geometric constraints such as the smoothness of the curve [42]. Values of these features are obtained from the entire region of an image, while edge-based geometric active contours utilize only local information around a curve. Therefore, region-based geometric active contours yield much more robust results to noise and to any locations of initial curves than those from edge-based geometric active contours.

Chan and Vese proposed a model based on curve evolution, Mumford-Shah functional for segmentation, and level set methods [36]. Let Ω be an open subset of $\mathbf{R}^2$,

$u_0 : \overline{\Omega} \to \mathbf{R}$. Then, energy in the Chan-Vese model is defined as

$$\begin{aligned} E(c_i, c_o, \Psi) &= \mu_1 \int_\Omega \delta(\Psi(x,y)) \|\nabla \Psi(x,y)\| dxdy \\ &+ \mu_2 \int_\Omega H(\Psi(x,y)) dxdy \\ &+ \lambda_1 \int_\Omega |u_0(x,y) - c_i|^2 H(\Psi(x,y)) dxdy \\ &+ \lambda_2 \int_\Omega |u_0(x,y) - c_o|^2 (1 - H(\Psi(x,y))) dxdy \ , \end{aligned} \tag{9}$$

where $\mu_1 \geq 0,\ \mu_2 \geq 0,\ \lambda_1, \lambda_2 > 0$, and $H(\cdot)$ is a heavy side function. The first two terms in the right-hand side of equation (9) represent energy associated with the smoothness of the evolving curve, and the third and fourth term represent energy associated with the features in the image, where c_i is the average value of features inside the curve C, and c_o is that outside C. c_i and c_o are defined as

$$c_i(\Psi) = \frac{\int_\Omega u_0(x,y) H(\Psi(x,y)) dx\, dy}{\int_\Omega H(\Psi(x,y)) dx\, dy} \tag{10}$$

$$c_o(\Psi) = \frac{\int_\Omega u_0(x,y)(1 - H(\Psi(x,y))) dx\, dy}{\int_\Omega (1 - H(\Psi(x,y))) dx\, dy} \ . \tag{11}$$

The corresponding Euler-Lagrange equation is

$$\frac{\partial \Psi}{\partial t} = \delta(\Psi) \Big[\mu \ \mathtt{div}(\frac{\nabla \Psi}{\|\nabla \Psi\|}) - \nu - \lambda_1 (u_0 - c_i)^2 + \lambda_2 (u_0 - c_o)^2 \Big] \ . \tag{12}$$

2.4 Fast Marching Methods

In many applications, the fast marching method is used to reduce computational time of level set methods. The fast marching method is a special case of level set methods in which contours in an image plane move in only one direction where it means if the contour passes one pixel then that pixel can not be revisited. The method is the combination of a narrow-band approach with a single pixel bandwidth and a heap sorting algorithm. Thus, a unique value of "crossing time" is associated with each pixel. The resulting crossing time function is a time-invariant level set function that satisfies the following equation:

$$\phi(x, y)\|\nabla T(x, y)\| = 1 \ , \tag{13}$$

which is a form of the well-known Eikonal equation. The goal is to numerically solve equation (13) for $T(x, y)$ over a subset of an image data located around the expected target location. The numerical solution of equation (13) will result in active contours propagating across the image and "wrapping" themselves around objects. A two-dimensional numerical approximation of equation (13) is given as follows,

$$\begin{aligned}[&max(D_{ij}^{-x}T(x, y), 0)^2 + min(D_{ij}^{+x}T(x, y), 0)^2 \\ &+max(D_{ij}^{-x}T(x, y), 0)^2 + min(D_{ij}^{+x}T(x, y), 0)^2]^2 = \frac{1}{\phi(x, y)^2} \ .\end{aligned} \tag{14}$$

In fast marching methods, all pixels in an image belong to one of three sets. The first set, called the "Known" set, contains all pixels over which the curves have already passed. The second set, called the "Trial" set, contains all pixels in the curves. The third set, called the "Far" set, contains all the pixels which have not yet been processed. The details of fast marching methods are available in [32, 33, 35], and the algorithm is summarized as follows:

1. At the start of each frame, choose the location of the initial curves. Put all the points in the initial curves into "Trial."

2. Begin loop: Find the pixel which has the smallest T value in the "Trial" set using the heap-sorting algorithm. Label this pixel "A."

3. Add "A" to the "Known" set and remove it from the "Trial" set.

4. Add all pixels neighboring "A" that are not in the "Known" set to the "Trial" set. If any of these pixels is in the "Far" set, remove them.

5. Recompute the values of T(x,y) for all neighbors by solving equation (14). Return to step 2.

CHAPTER III

A FAST IMPLEMENTATION OF THE CHAN-VESE MODEL

A natural choice for image segmentation in real-time applications was using the fast marching method that works much faster than the standard level set technique. An unmanned helicopter succeeded in autonomous visual searching and a formation flight with another UAV using a single vision sensor [22, 23]. However, the segmentation method based on the fast marching method had some significant limitations. It was sensitive to local minima and could not effectively use the results from the previous frame. Refer to Section 7.2.1.4 for details of these limitations.

This chapter presents a fast implementation of geometric active contour model that is effective in real-time applications. This novel implementation improves the robustness and the speed of geometric active contours. Robustness to local minima is improved by employing the Chan-Vese's active contour model which is a region-based geometric active contour model. The computational time of this method is drastically improved by combining two factors: First, this method employs the pixel level curve evolution proposed by Shi and Karl [44]; and second, it employs a fast level-set-based optimization method. Furthermore, the fast implementation of the Chan-Vese model adds a novel regularizing term that ensures the smoothness of the evolving curve. With this method, real-time applications become more tractable in our visual tracking system. Section 3.1 reviews the fast level set implementation, Section 3.2 the fast level-set-based optimization, and Section 3.3 the formulation of the fast implementation. Finally, section 3.4 presents some results using the new method.

3.1 Fast Level Set Implementation

In level set methods, the evolving curve in an image plane is represented as the zero-level set of a higher-dimensional functional, $\Psi(\mathbf{z},t) : \mathbf{R}^2 \times \mathbf{R}^+ \rightarrow \mathbf{R}$, where $\mathbf{z} \in \mathbf{R}^2$. In general, Ψ is constructed by calculating the minimum distance from each point to the zero-level set, $\Psi(\mathbf{z},t) = 0$. The curve evolution is realized by the calculation of geometric PDEs on the function Ψ, which are bounded to stability constraints [28]. If curve evolution does not depend on PDEs, a simplified version of Ψ, namely, Ψ^s can be used [37, 44]. In these cases, Ψ^s is classified into the two groups, positive Ψ^s and negative Ψ^s, and the curve evolution is realized by switching elements between two groups according to a given conditions. Examples of these conditions are a sign of either the variation of the energy term [37] or the speed function derived from region competition methods [44]. This switching process speeds up curve evolution by not solving geometric PDEs.

Let us briefly review the fast level set implementation algorithm that leads to the curve evolution at a pixel resolution [44]. The Ψ^s which represents Ψ constructed by the fast level set implementation, is defined as given in [44],

$$\Psi^s(\mathbf{z}) = \begin{cases} +3, & \text{if } \mathbf{z} \text{ is an exterior point;} \\ +1, & \mathbf{z} \ \in \ L_{out}, \\ -1, & \mathbf{z} \ \in \ L_{in}, \\ -3, & \text{if } \mathbf{z} \text{ is an interior point,} \end{cases}$$

where $\mathbf{z}$, $\mathbf{z}_N \ \in \mathbf{R}^2$. L_{in} *and* L_{out} are defined as follows:

$$\begin{aligned} L_{out} &= \{ \ \mathbf{z}|\Psi^s(\mathbf{z}) > 0 \ and \ \exists \mathbf{z}_N \in N_4(\mathbf{z}) \ s.t. \ \Psi^s(\mathbf{z}_N) < 0 \ \}, \\ L_{in} &= \{ \ \mathbf{z}|\Psi^s(\mathbf{z}) < 0 \ and \ \exists \mathbf{z}_N \in N_4(\mathbf{z}) \ s.t. \ \Psi^s(\mathbf{z}_N) > 0 \ \} \ , \end{aligned}$$

where $N_4(\mathbf{z})$ represents the 4 neighbor points around $\mathbf{z}$. Points that are not in L_{in} *and* L_{out} are defined as either exterior points, if $\Psi^s(\mathbf{z}) > 0$, or interior points, if $\Psi^s(\mathbf{z}) < 0$. Curve evolution is realized by switching elements between the L_{in} and the L_{out} and its neighbors. Switching elements depends on the sign of $F(\mathbf{z})$, which is derived from the region competition algorithm [41] and the connectedness of the sign at each point. This switching process leads to a curve evolution at a pixel resolution. The procedure of switching is composed of two functions: $switch\_in(\mathbf{z})$ and $switch\_out(\mathbf{z})$. These functions are defined as follows:

$switch\_in(\mathbf{z})$:

1. step 1: Delete $\mathbf{z}$ from L_{out} and add it to L_{in}. Set $\Psi^s(\mathbf{z}) = -1$.
2. step 2: $\forall \mathbf{z}_N \in N_4(\mathbf{z})$ satisfying $\Psi^s(\mathbf{z}_N) = 3$, add $\mathbf{z}_N$ to L_{out}, and set $\Psi^s(\mathbf{z}_N) = 1$.

$switch\_out(\mathbf{z})$:

1. step 1: Delete $\mathbf{z}$ from L_{in} and add it to L_{out}. Set $\Psi^s(\mathbf{z}) = 1$.
2. step 2: $\forall \mathbf{z}_N \in N_4(\mathbf{z})$ satisfying $\Psi^s(\mathbf{z}_N) = -3$, add $\mathbf{z}_N$ to L_{in}, and set $\Psi^s(\mathbf{z}_N) = -1$.

See [44] for details of the algorithm.

3.2 Fast Level Set Based Optimization Method

The Chan-Vese's active contour model is robust to local minima as presented in Section 2.3. An elegant optimization method for the Chan-Vese model is introduced in [37]. Similar to the method in the previous section, this method improves the computational speed of the Chan-Vese model [36] by not solving geometric PDEs. Curve evolution is realized by switching the sign of $\Psi^s(\mathbf{z})$. A test of switching the sign of Ψ^s is performed in order to examine the variation of the energy functional with respect to the test. If energy is decreased by the switching, the change of sign is accepted, otherwise it is neglected.

A brief description of the method is given as follows. Define $\Psi^i = \{\mathbf{z}|\Psi^s(\mathbf{z}) < 0\}$, and $\Psi^o = \{\mathbf{z}|\Psi^s(\mathbf{z}) \geq 0\}$. Let η_i be the total number of pixels for Ψ^i and η_o for Ψ^o. Then, we can calculate E^i which is the Chan-Vese energy for the set Ψ^i; and E^o for Ψ^o. (The length minimizing coefficient μ is set to be zero in equation (9) for the sake of simplicity of the calculations of E^i and E^o [37]). If we switch one point from $\mathbf{z} \in \Psi^o$ to $\mathbf{z} \in \Psi^i$, the variation in energy ∇E^{oi} caused from the switch can be calculated; ∇E^{io} for the opposite case. ∇E^{io} and ∇E^{oi} are given as in [37],

$$\nabla E^{io} = (u_0 - c_o)^2 \frac{\eta_o}{\eta_o - 1} - (u_0 - c_i)^2 \frac{\eta_i}{\eta_i + 1}, \tag{15}$$

$$\nabla E^{oi} = (u_0 - c_i)^2 \frac{\eta_i}{\eta_i + 1} - (u_0 - c_o)^2 \frac{\eta_o}{\eta_o - 1} \ . \tag{16}$$

If $\nabla E^{io} < 0$, or $\nabla E^{oi} < 0$, then the switching is accepted, otherwise the switching is neglected. This method scans the entire image multiple times until the target object is completely separated from background. Refer to [37] for further details of this method.

3.3 Formulation of The Fast Implementation of The Chan-Vese's Model

This section combines the two fast algorithms described in Sections 3.1 and 3.2 and adds a regularizing term specifically designed for the pixel level curve evolution. The main disadvantage of the fast-level-set based optimization method is that the algorithm scans the entire image with a relaxation method. Such scans may not be necessary in cases when the target is already detected in the previous frame. Instead of scanning the entire image, if we consider the final contour in the previous frame as the initial contour in the present frame, the initial contour may converge fast in the present frame [100]. This will drastically reduce the required number of iterations. The fast level set implementation fits well this idea.

On the other hand, in the fast level set implementation method in [44], the speed function $F(\mathbf{z})$ is derived from the region competition method. However, all the probability models underlying the input images is required to know a priori to use the speed function [41]. On the other hand, the Chan-Vese active contour model doesn't require to know the probability models in advance, and initial contour can be placed anywhere in a given image. In addition, the Chan-Vese model is not sensitive to local minima. Hence, this thesis proposes a fast implementation of Chan-Vese active contour model based on fast level-set-based optimization employing fast level set implementation. This fast implementation greatly reduces the computational time of the Chan-Vese method so that it is very effective in real-time applications.

In several fast implementations of the Chan-Vese's model, researchers use a separate smoothing processes with curve evolution to ensure the smoothness of the evolving contour [71, 44, 73]. For example, Shi and Karl [44] apply a Gaussian smoothing around a curve after one iteration of curve evolution. However, instead of separating the curve evolution with smoothing processes, most of geometric active contours include regularizing terms in the energy functional and the corresponding curve evolution is directly influenced by the terms [28, 27, 29, 36].

This thesis proposes a regularizing term that can be directly implemented in equations (15) and (16). This new term ensures the smoothness of the curve during the evolution. This term is applied only in case when there is big difference between the lengths of each list. The corresponding changes of energy can be formulated as follows,

$$\begin{aligned} \nabla \widetilde{E^{io}} &= \nabla E^{io} + \rho_1 H(|1 - \frac{N_{out}}{N_{in}}| - \epsilon_N)(4 - N_8^+(\mathbf{z})), && (17) \\ \nabla \widetilde{E^{oi}} &= \nabla E^{oi} + \rho_1 H(|1 - \frac{N_{in}}{N_{out}}| - \epsilon_N)(N_8^+(\mathbf{z}) - 4)\ , && (18) \end{aligned}$$

where $\rho_1 > 0, \epsilon_N > 0$, N_{in} and N_{out} are the number of pixels in the list L_{in} and L_{out} respectively. $H(\cdot)$ is a heavy side function, and $N_8^+(\mathbf{z})$ is the number of pixels that have a plus sign for the eight neighboring pixels of $\mathbf{z}$. The idea of the term

$H(|1-\frac{N_{out}}{N_{in}}|-\epsilon_N)$ is to keep balance between the length of the two lists. Because the two lists border each other, if the difference between N_{in} and N_{out} is large, the curve adheres to a smaller object. (This term is calculated only at the start of each scan of the two lists.) In this case, the regularizing term is turned on. The idea of the regularizing term is to minimize $|4-N_8^+(\mathbf{z})|$, which balances between the number of negative and positive pixels in the neighborhood of $\mathbf{z}$ gives local smoothness.

Let us explore this method by using a simple example. Figure 1 contains a 12 by 12 pixel black and white image where the intensity value at a white pixel is 1 and at a black pixel 0. Figure 1.1. shows the original image. A point $\mathbf{z}=(x,y)$ where x is the horizontal coordinate, y is the vertical coordinate, and the origin is at the left-bottom point of the image. Figure 1.2. shows Ψ^s constructed on the image. Ψ^s possesses four values $\{-3,-1,+1,+3\}$ as in Section 3.1. The $\Psi^s(\mathbf{z})=-3$ is represented by 'O' and $\Psi^s(\mathbf{z})=+3$ by '&' for the better distinction with other numbers in the image. The initial zero level set is represented by the red rectangle. In this example the regularizing term is set to zero ($\rho_1=0$) for the sake of simplicity of the calculations of the variations of energy. However, Figure 2 shows the effects of the smoothing term in complicated images.

The details of the steps of the algorithm are as follows:

1. Figure 1.2:

   - From the initialization of Ψ^s, η_i=30, $\eta_o=114$, c_i=0.067, and c_o=0.833 from equations (10), (11).
   - For i = 1 to N_{in}, calculate $\nabla\widetilde{E^{io}}$
   - Consider $\mathbf{z}=(4,7)$ where $\nabla\widetilde{E^{io}}(\mathbf{z})=-0.873<0$. Therefore, the switch from Ψ^i to Ψ^o is accepted. The corresponding η_i=29, $\eta_o=115$, c_i=0.035, and c_o=0.835 are updated. Then, *switch_out*$(\mathbf{z})$ is executed.
   - Consider $\mathbf{z}=(4,8)$ where $\nabla\widetilde{E^{io}}(\mathbf{z})=-0.938<0$. Therefore, the switch

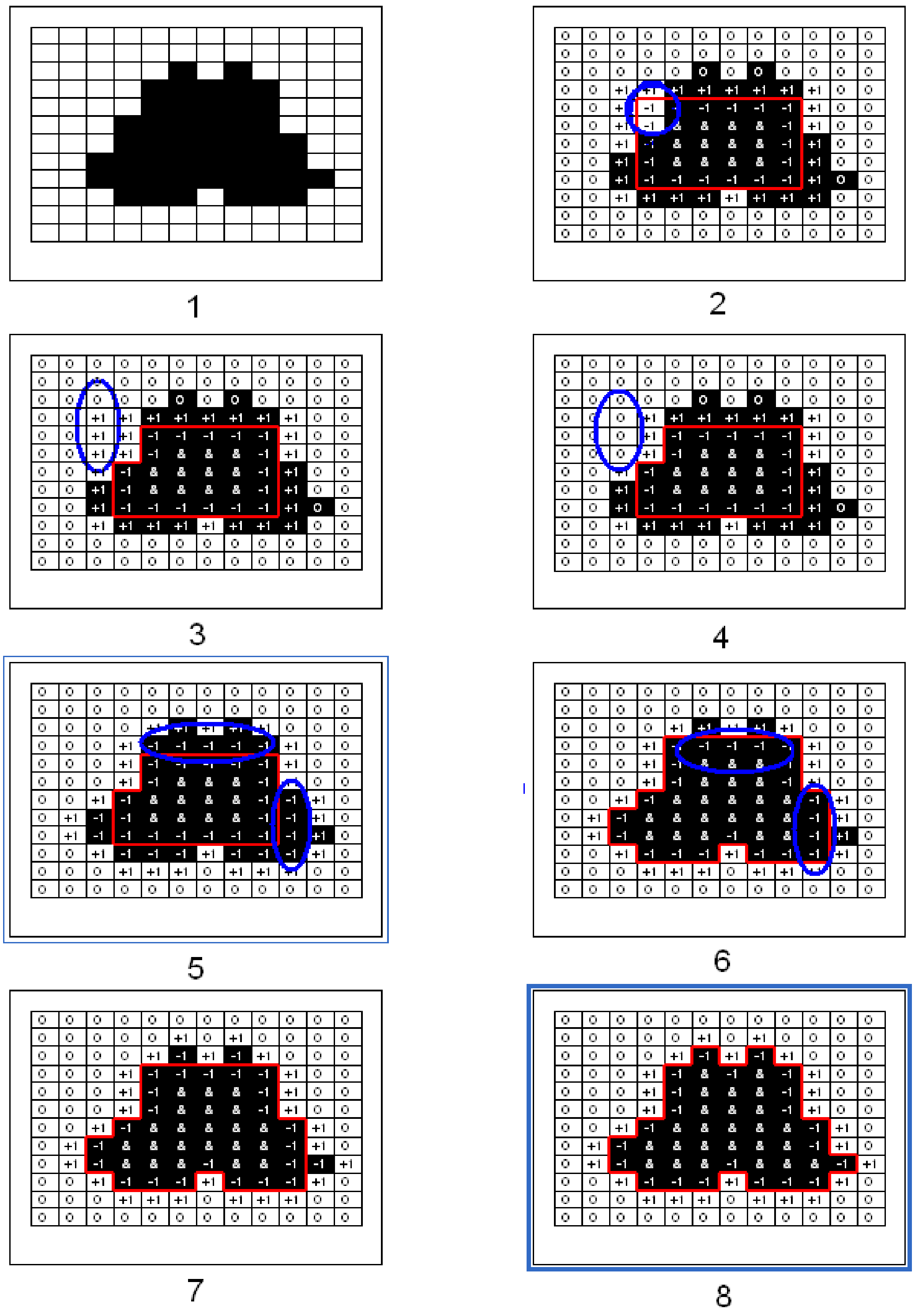


Figure 1: Procedure of the fast implementation of the Chan-Vese model

from Ψ^i to Ψ^o is accepted. The corresponding variables are updated. Then, *switch_out*$(\mathbf{z})$ is executed.

2. Figure 1.3: This image shows the Ψ after the switching test in L_{in}

3. Figure 1.4: For each point $\mathbf{z} \in L_{out}$, if $\forall \mathbf{z}_N \in N_4(\mathbf{z})$, if $\Psi^s(\mathbf{z}_N), > 0$ delete $\mathbf{z}$ from L_{out} and set $\Psi(\mathbf{z}) = 3$. Three points (3,7), (3,8), and (3,9) are changed to 'O' in Figure 1.3.

4. Figure 1.5:

   - For i = 1 to N_{out}, calculate $\nabla \widetilde{E^{oi}}$
   - Consider $\mathbf{z} = (3, 4)$ where $\nabla \widetilde{E^{oi}}(\mathbf{z}) = -0.705 < 0$. Therefore, the switch from Ψ^o to Ψ^i is accepted. The corresponding η_i=29, $\eta_o = 115$, c_i=0.843, and c_o=0.000 are updated. Then, *switch_in*$(\mathbf{z})$ is executed.
   - Consider $\mathbf{z} = (3, 5)$ where $\nabla \widetilde{E^{oi}}(\mathbf{z}) = -0.718 < 0$. Therefore, the switch from Ψ^o to Ψ^i is accepted. The corresponding variables are updated. Then, *switch_in*$(\mathbf{z})$ is executed.

5. Figure 1.6: For each point $\mathbf{z} \in L_{in}$, if $\forall \mathbf{z}_N \in N_4(\mathbf{z})$, if $\Psi(\mathbf{z}_N) < 0$, delete $\mathbf{z}$ from L_{in} and set $\Psi(\mathbf{z}) = -3$

6. Figures 1.7 and 1.8: Process from 1 to 6 is repeated. In Figure 1.8, c_i=0.000 and c_o=1.000. The target object is completely separated from background.

The entire procedure including equations (17) and (18) of the new method is summarized in Table 1.

3.4 Experiments

Table 2 shows a comparison of computational times between the fast implementation and the Chan-Vese model on several test images. The new model is at least 40

```
Initialization: Initialize $\Psi^s$, $c_i$ and $c_o$ from the location
    of the initial curve and divide the zero level set into
    two lists $L_{in}$ and $L_{out}$

Step 1:
    ▪ Calculate $N_{out}/N_{in}$
    ▪ For i = 1 to $N_{in}$, calculate $\nabla\widetilde{E^{io}}$ and if $\nabla\widetilde{E^{io}} < 0$,
      do $switch\_out(\mathbf{z})$ and update $c_i,\ c_o,\ \eta_i,\ \eta_o$
    ▪ For each point $\mathbf{z} \in L_{out}$, if $\forall \mathbf{z}_N \in N_4(\mathbf{z})$, if $\Psi^s(\mathbf{z}_N) > 0$,
      delete $\mathbf{z}$ from $L_{out}$ and set $\Psi^s(\mathbf{z}) = 3$

Step 2:
    ▪ Calculate $N_{in}/N_{out}$
    ▪ For i = 1 to $N_{out}$, calculate $\nabla\widetilde{E^{oi}}$ and if $\nabla\widetilde{E^{oi}} < 0$,
      do $switch\_in(\mathbf{z})$ and update $c_i,\ c_o,\ \eta_i,\ \eta_o$
    ▪ For each point $\mathbf{z} \in L_{in}$, if $\forall \mathbf{z}_N \in N_4(\mathbf{z})$, if $\Psi^s(\mathbf{z}_N) < 0$,
      delete $\mathbf{z}$ from $L_{in}$ and set $\Psi^s(\mathbf{z}) = -3$

Step 3: Check the stopping condition. If it is not satisfied,
        go to step 1. otherwise terminate the procedure.
```

Table 1: Basic algorithm of the new real-time geometric active contours.

Method	Region-based active contours	Real-time active contours
Europe Night	32.74 sec	0.1 sec
Spiral	108.85 sec	2.37 sec
GT-Edge	N/A	0.01 sec

Table 2: Comparison of the computational times

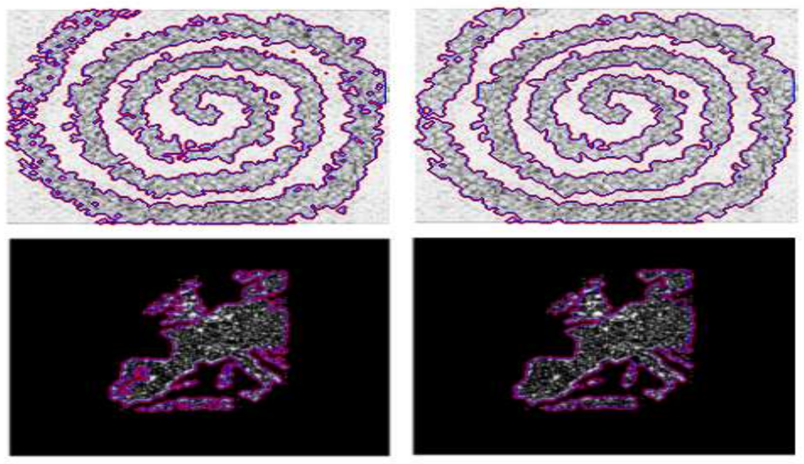


Figure 2: Left: Without the regularizing term, N_{in}=4042 , N_{out}=3932. Right: With the regularizing term, N_{in}=3125, N_{out}=3088.

times faster than the original Chan-Vese model. Figure 2 shows the effects of the regularizing term. In the top image of Figure 2, the spiral image, the total number of pixels of the final curve decreased about 25 percent. The final contour with the regularizing term is smoother than that of without the term. In the bottom of Figure 2, the "Europe-night light image," the final curve breached the boundary of the object without the regularizing term (the lower-left image of Figure 2). On the other hand, the leakage has disappeared with the regularizing term (lower-right image of Figure 2). Figure 3 shows curve evolution process on the "Europe night-lights" image using the new model. Figure 4 shows the results using the fast implementation of the Chan-Vese model applied to a sequence of aerial target tracking in cloudy images. The average computational time reached to 100 frames/seconds. Figure 5 shows the results of the new model applied to a sequence of GTEdge images. In this case, the final contour in one frame is used as the initial contour in the next frame. Figure 6 shows the curve evolution process with the new algorithm on the complicated spiral image. In all figures in this chapter, the initial curve is represented as blue or black and moving contours are represented as red.

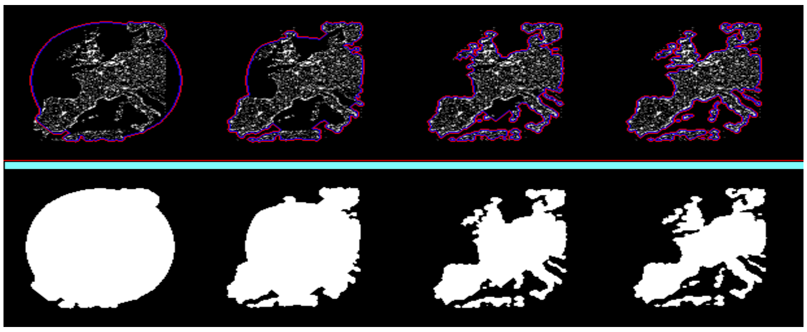


Figure 3: Curve evolution on Europe night-lights with the new method, size = 116 ×110.

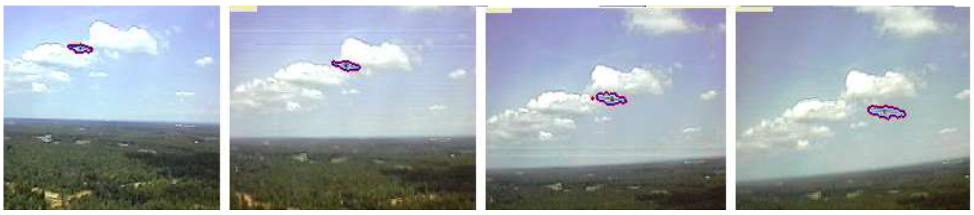


Figure 4: The fast implementation of Chan-Vese model applied to aerial target tracking in cloudy images, size = 240 ×352, cpu = 0.01seconds (average).

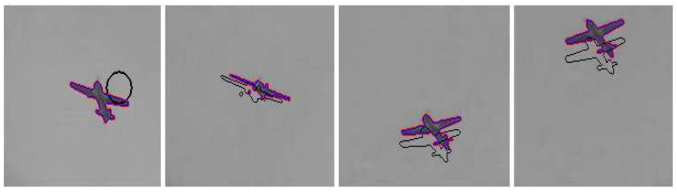


Figure 5: Results on GTEdge images with the new method, size = 240 ×352, cpu = 0.01sec(average).

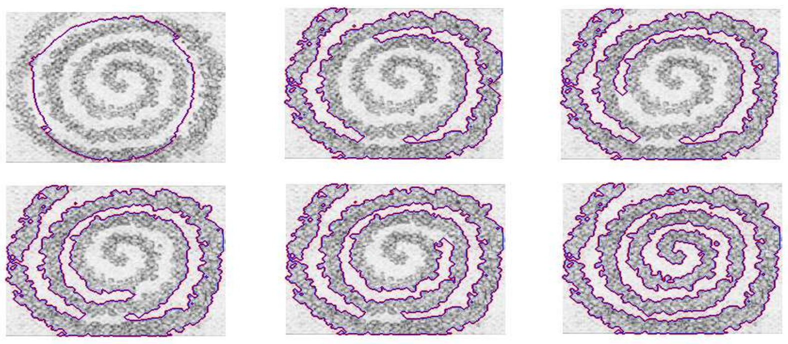


Figure 6: Curve evolution on a spiral image with the new method, size = 234×191.

CHAPTER IV

SHAPE MODELING

The outline of a target in the image plane can be easily disrupted by noise such as clutters, occlusions, diffusion, and so on. Therefore, prior knowledge about the shape of a target may play an important role for recognizing the correct target during tracking. Visual trackers may find the correct target by estimating differences between the known shape and the shape measurements obtained from a given image. A certain type of shape representation is required to quantify the differences.

In the past 20 years, geometric active contours have been extensively used in extracting the outline of an object [29, 27, 28, 42, 36]. In this approach, the outline of the target is implicitly represented by the zero-level set of higher dimensional function Ψ; Refer to Section 2.2 for details of Ψ. In some existing methods, prior shape inof0rmation is embedded into the higher dimensional function Ψ in order to be robust to noise [75, 97, 76, 98, 77, 99]. The difference between two shape is quantified by subtracting the corresponding two higher dimensional functionals. Geometric active contours based on shape prior have shown robust results to various noise such as occlusion, missing part and diffusion.

In [77], the authors proposed a dissimilarity measure, which calculates the differences between one shape from another, where shape information was embedded in Ψ. The dissimilarity model was

$$d^2(\Psi_1, \Psi_2) = \int_\Omega (\Psi_1 - \Psi_2)^2 \frac{h(\Psi_1) + h(\Psi_2)}{2}\, dxdy, \tag{19}$$

where $h(\Psi)$ is the normalized Heavyside function defined by $h(\Psi) = H(\Psi) / \int_\Omega H(\Psi) dxdy$. An energy equation for shape information was defined using this dissimilarity measure

d^2,

$$E^k_\Psi(\Psi_{sp}, \Psi_k) = d^2(\Psi_{sp}, \Psi_k), \tag{20}$$

where Ψ_{sp} contains prior shape information and Ψ_k is the k-the measurement obtained from a given image. This dissimilarity measure is useful for contour tracking in the presence of clutter and occlusions.

However, constructing Ψ is computationally expensive process in real-time tracking problems. In addition, the fast implementation of the Chan-Vese model (FICVM) in Chapter 3 has some disadvantages in the accuracy of the dissimilarity measure given in equation (19). These disadvantages arise from the simple structure of Ψ^s used in the FICVM. Refer to Section 3.1 for details of Ψ^s. In general, Ψ is constructed by calculating the minimum distance from each point in an image to the zero level set of Ψ. Therefore, Ψ is a smooth three-dimensional surface. In the FICVM, Ψ^s consists of only four values ($\{-3, -1, +1, +3\}$). Therefore, the Ψ^s in the FICVM cannot describe rich prior shape information as the Ψ in [77], and the dissimilarity measure in FICVM is not as accurate as the function in [77]. For example, consider a point $\mathbf{z}$ outside of the zero-level set. Then, $\Psi(\mathbf{z})$ is the minimum distance from the zero-level set to $\mathbf{z}$, however, $\Psi^s(\mathbf{z})$ is always $+3$ regardless of the distance from the zero-level set.

This thesis presents a novel shape representation based on the zero level set of Ψ^s which is computationally inexpensive but can extract useful information of the shape of the target. The new shape representation is used for quickly detecting multiple objects in a given image. Furthermore, the new shape energy function is combined with the energy equation (20) to compensate for the inaccuracy in the calculation of the dissimilarity measure in the FICVM.

The following sections introduce a contour-based shape modeling scheme. Section 4.1 presents the algorithm for extracting an individual closed contour from the segmented image. Section 4.2 presents the way of finding feature points in the contour. Section 4.3 discusses shape analysis using the histogram of line segments. Section 4.4 presents two examples of detecting multiple objects by using the new shape representation. Finally, a model for shape energy is provided in Section 4.5.

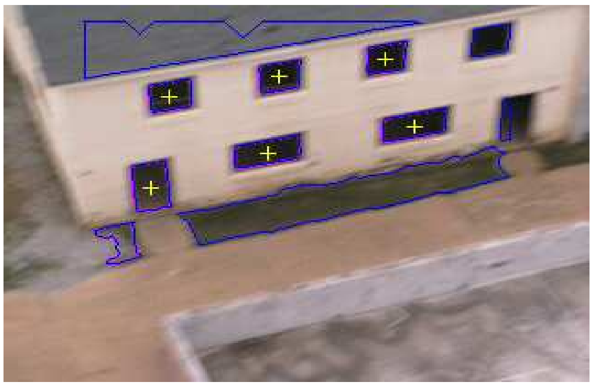
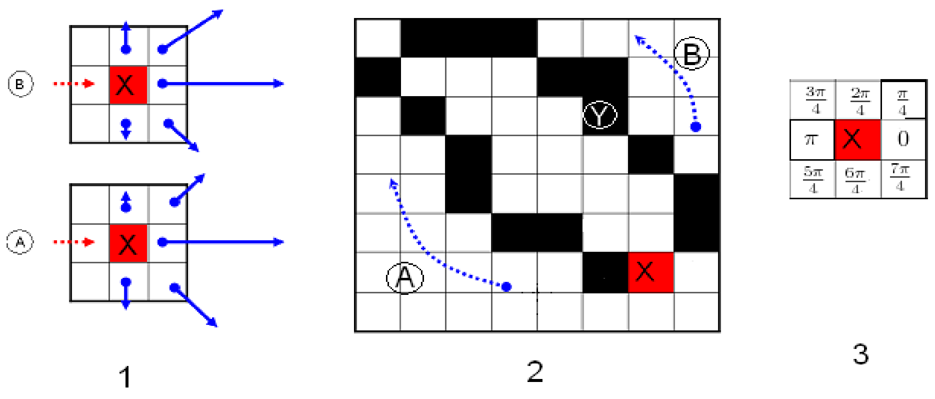


Figure 7: Detection of multiple windows in a building.

4.1 Extracting Individual Closed Contour

This section presents an algorithm for extracting individual closed contours from the zero level set of Ψ^s after segmentation. Active contours move toward the outlines of the objects in a given image as the curve evolution progresses. Because the topologies of active contours are free to change during the evolution [28], the exact number of individual closed contours is uncertain until the evolution is finished. For example, Figure 7 displays the results of segmentation using the FICVM. At the initial step of the segmentation, the active contours contain only one closed contour. However, eleven closed contours are produced at the end of the segmentation. Notice that the structure of Ψ^s is always preserved during the segmentation as is the zero-level set of Ψ^s, which is composed of multiple closed contours. In fact, one advantage of active contours over some primitive image processing methods (e.g., edge detection, corner detection) is that the final contour is closed. Therefore, the contour can be considered as one complete unit or a part of the target. Discontinuous edges or corners require



B
X
A
X
1
B
Y
A
X
2
$\frac{3\pi}{4}$ $\frac{2\pi}{4}$ $\frac{\pi}{4}$
π X 0
$\frac{5\pi}{4}$ $\frac{6\pi}{4}$ $\frac{7\pi}{4}$
3

Figure 8: Contour extracting algorithm

extensive post processing to recover the shape of the target. It is assumed that each closed contour corresponds to the entire or part of outline of an object so that shape data of each object can be determined by the shape of the closed contour. The tracker should identify each individual closed contour to obtain the shape data of the corresponding object.

This thesis focuses on shape information available from the zero-level set of Ψ^s which doesn't require large computational loads, and proposes a novel algorithm for extracting the individual closed contours quickly. The goal of the algorithm is to trace around and return to the starting point using the connected points of a contour. The algorithm may be asked to choose from multiple choices of the connected points during the process. In this case, the algorithm has some priorities and limitations in choosing the next connected points.

Let us explore this algorithm by using a simple example. Figure 8.2 contains an 8 by 8 pixel black and white image. The active contour is represented by the black points. The starting point is denoted by the red X-marked point at the bottom-right of Figure 8.2. The 8-neighborhood points of the starting point are numbered from 0 to π. Each label represents the orientation of the next connected point from the

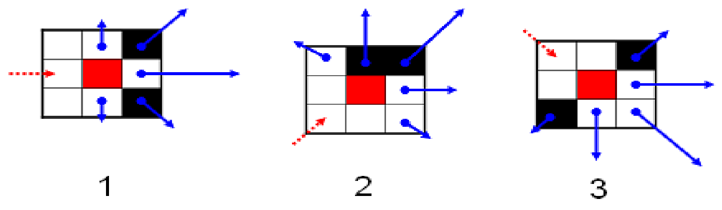


Figure 9: Priority in route B

starting point as in Figure 8.3. The starting point is the initial checking point. There are two route choices at the initial checking point, namely route A or B. Once the algorithm chooses one route, it can not alter the route selection during the process.

Each route has some priorities and limitations in its advancement to the next connected point as in Figure 8.1. Both routes have the same limitations that the algorithm can not choose the point in the opposite direction from the previously advanced direction. For example, Figure 9 shows three examples for choosing priority and for imposing limitations of in the algorithm of the advancement. In Figure 9.1 the current checking point comes from the 0 radians direction from the previous checking point (the previous direction is represented by the red line), hence the connected points in the directions of $\frac{3\pi}{4}, \pi$, and $\frac{5\pi}{4}$ are excluded from the advancement. The lengths of blue arrows in Figure 9 represent the possibilities of being selected as the next checking point. There are two connected points in the directions of $\frac{\pi}{4}$ radians and $\frac{7\pi}{4}$ radians which are represented as black squares in Figure 9.1. Because the length of the blue arrow in the point at the direction of $\frac{\pi}{4}$ radians is larger than that in the point at $\frac{7\pi}{4}$ radians, the algorithm selects the point at $\frac{\pi}{4}$ radians. In Figure 9.2, the connected points in the directions of π, $\frac{5\pi}{4}$ and $\frac{6\pi}{4}$ are excluded from the advancement, and the algorithm selects the point at the direction of $\frac{\pi}{4}$ radians from the two possible choices ($\frac{\pi}{4}$, $\frac{2\pi}{4}$). In Figure 9.3, the connected points in the directions of π, $\frac{3\pi}{4}$, and $\frac{2\pi}{4}$ are excluded from the advancement, and the algorithm selects the point at the

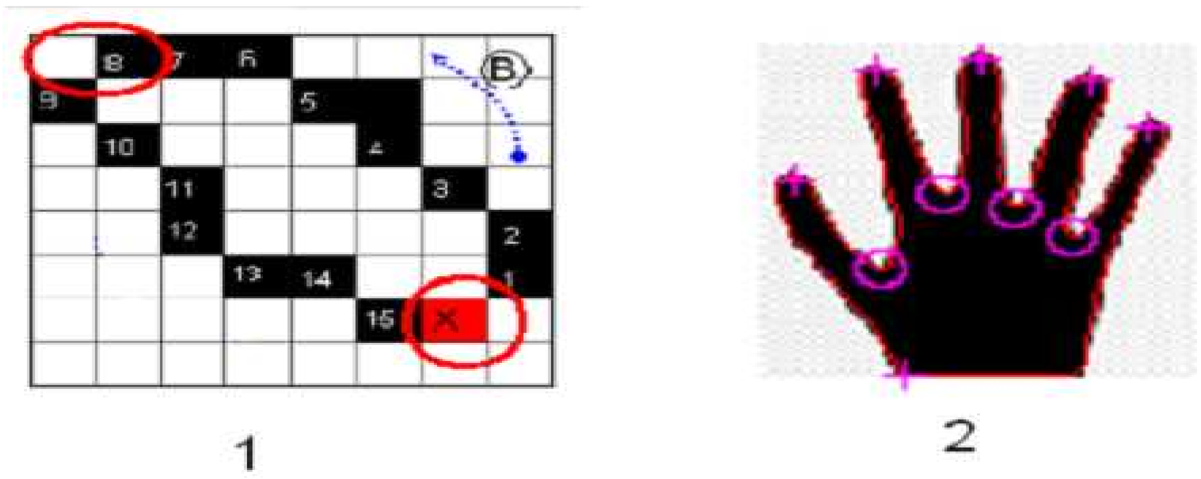


Figure 10: Sequence of contour following algorithm

Iteration	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Orientation	0	$\frac{\pi}{4}$	$\frac{2\pi}{4}$	$\frac{3\pi}{4}$	$\frac{3\pi}{4}$	$\frac{3\pi}{4}$	$\frac{3\pi}{4}$	π	π	$\frac{5\pi}{4}$	$\frac{7\pi}{4}$	$\frac{7\pi}{4}$	$\frac{6\pi}{4}$	$\frac{7\pi}{4}$	0	$\frac{7\pi}{4}$	0
Difference	0	$\frac{\pi}{4}$	$\frac{\pi}{4}$	$\frac{\pi}{4}$	0	0	0	$\frac{\pi}{4}$	0	$\frac{\pi}{4}$	$\frac{2\pi}{4}$	0	$\frac{-\pi}{4}$	$\frac{\pi}{4}$	$\frac{\pi}{4}$	$\frac{-\pi}{4}$	$\frac{\pi}{4}$

Table 3: Interpretation into angle information for Figure 10.1

direction of $\frac{\pi}{4}$ radians from the two possible choices ($\frac{\pi}{4}$, $\frac{5\pi}{4}$).

Each route has different priorities. If the algorithm follows route A, the priorities are in the clock-wise (CW) connected points, in the case of route B, they are the counter-clock-wise (CCW) ones. The checking point advances to each new point as the algorithm progresses.

Now let us follow route B in Figure 10. The initial orientation of the connected point at the starting point is 0 radians. The next connected point at the starting point is in the orientation of $\frac{\pi}{4}$. At the point Y, the algorithm has two choices in the orientations of $\frac{2\pi}{4}$ and $\frac{3\pi}{4}$, since the previous orientation was $\frac{3\pi}{4}$, the orientation $\frac{3\pi}{4}$ has more priority than $\frac{2\pi}{4}$ by the priorities in the advancement. With these basic principles, the contour following algorithm can return to the starting point as in Table 3. The entire procedure following route B is shown in Figure 10.1 and Table 3.

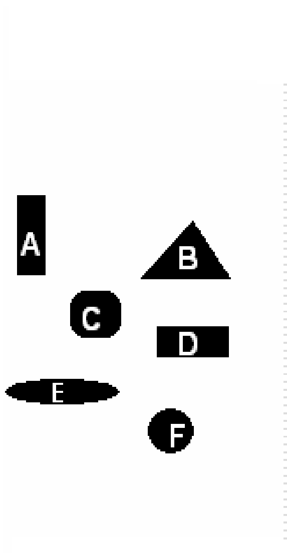
4.2 Finding Feature Points

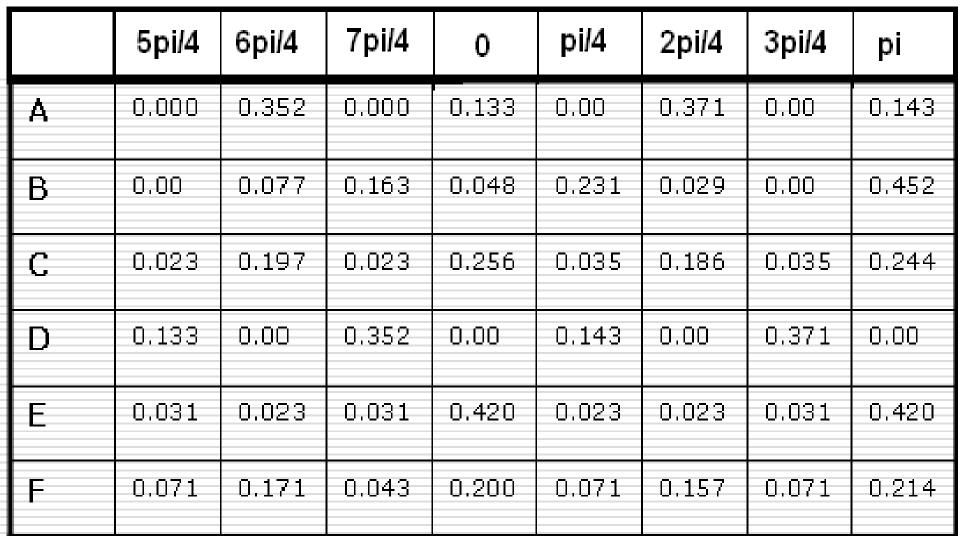
Contour-based shape information is subject to the correspondence problem which is less sensitive when Ψ is used [75]. Finding the correspondence between two shapes is an important step, but it is also a difficult one [75]. One way to solve the correspondence problem is to use feature points which have some noticeable geometric characteristics. This section presents a novel strategy in finding feature points based on the algorithm for extracting a closed contour in Section 4.1. The basic idea of the new strategy is to use the fact that π or $-\pi$ radian changes of the orientation of the connectivity is required to turn around to the opposite direction from the orientation of the connectivity at the starting point. To realize this idea, the change of the orientation of the connectivity with respect to the iteration number at each point is obtained and integrated in Table 3. The accumulated changed orientation of the connectivity from the starting point exceeds π radian at the ninth point of iteration in Table 3. Therefore, the eighth point is registered as a feature point as in Figure 10. Then, the accumulated change of the orientation of the connectivity is reset to zero. Another feature point is the starting point as seen from Table 3.

Figure 8.2 shows the feature point detection for a more complicated contour from the hand image. The same strategy is used in this case. The route B is selected for this problem so that the priorities in advancement are given to the neighboring points. A unique relationship exists between the route and the change of the orientation. If the advancement moves to the left hand side, the change of the orientation is positive, otherwise, the change is negative. Therefore, the π radians of accumulated orientation means that the curve is turning by using the CCW connected points in the advancement. This can be an important cue for the differentiation between feature points. Ten feature points are detected in Figure 8.2. The sign '+' denotes a feature point which turned around π radians from one feature point, and 'o' denotes the turning point in $-\pi$ radians from the previous feature point.

4.3 The Histogram of The Orientations

This section presents a simple algorithm for getting some shape information using the orientation histogram. The new algorithm reduces multidimensional data sets into lower dimensions by constructing the orientation of the connectivity. This new method does not discard any information, but it calculates the frequencies with respect to the eight-orientations of the connectivity like a histogram. Let's temporarily assume that the contour has a simple convex shape. The problem with respect to more complicated contours will be dealt with by incorporating it with feature points. Matching between two histograms can be quickly computed and can be useful to get a quick glimpse of the shape.





	5pi/4	6pi/4	7pi/4	0	pi/4	2pi/4	3pi/4	pi
A	0.000	0.352	0.000	0.133	0.00	0.371	0.00	0.143
B	0.00	0.077	0.163	0.048	0.231	0.029	0.00	0.452
C	0.023	0.197	0.023	0.256	0.035	0.186	0.035	0.244
D	0.133	0.00	0.352	0.00	0.143	0.00	0.371	0.00
E	0.031	0.023	0.031	0.420	0.023	0.023	0.031	0.420
F	0.071	0.171	0.043	0.200	0.071	0.157	0.071	0.214

Figure 11: General shape values with the normalized histogram of the connectivity.

Figure 11 shows the histograms for various shapes. The histograms are normalized for comparison with other shapes. Here, the v-h ratio denotes the ratio between the vertical and the horizontal elements in the histogram. Note that the vertical elements are $\frac{\pi}{2}$ and $\frac{-\pi}{2}$. The horizontal elements are 0 and π. We can relate the histograms to various shape information as follows:

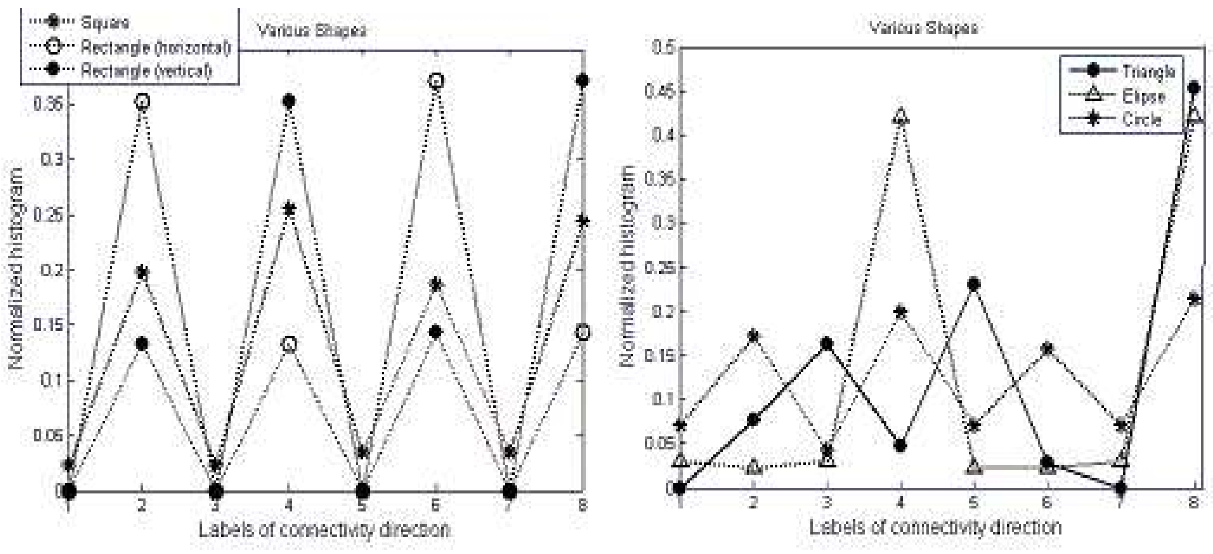


Figure 12: Normalized histogram values and connectivity direction.

- A) Rectangle: major components of the histogram are vertical elements, $\frac{\pi}{2}$ and $\frac{-\pi}{2}$. The v-h ratio is 2.620
- B) Triangle: major components of the histogram are three elements, $\frac{-\pi}{4}$, $\frac{\pi}{4}$, and π,
- C) Rectangle: major components are vertical and horizontal elements, $\frac{-\pi}{2}$, $\frac{\pi}{2}$, 0, and π. The v-h ratio is 0.766.
- D) Rectangle: major components of the histograms are horizontal elements, 0 and π. The v-h ratio is 0.382.
- E) Ellipse: major components of the histograms are horizontal elements, 0 and π. The v-h ratio is 0.055.
- F) Circle: major components are vertical and horizontal elements, $\frac{-\pi}{2}$, $\frac{\pi}{2}$, 0, and π. The v-h ratio is 0.792.

Figure 12 displays the graphical explanation of the histogram analysis.

However, the problem of using the histogram analysis is that a histogram can not be uniquely determined unless the shape has a simple contour. Consider the contour in Figure 10.2. The histogram for the entire contour in Figure 10.2 does not clearly relate to the shape of the hand. This problem can be ameliorated by breaking the closed contour into the feature points and the line segments between these feature points. Then, we may try to match the line segments. In general, the problem of matching line segments requires a point by point matching algorithm. However, finding correspondence between every point of two line segments is an exhausting process. Instead of seeking point by point correspondence, this thesis uses the histograms of the orientation of the connectivity for the line segments. The deformation over the line segment can be obtained by evaluating the differences between the two histograms corresponding the line segments.

4.4 Experiments

In this section, some results from experiments performed to test the proposed shape extracting algorithm are presented. The goal of these experiments was to quickly and automatically identify multiple known objects in a given image. We assumed that we had some knowledge about the geometry of the objects.

First, a given image (size = 240×240) is divided into multiple sub-images (size = 30×30). A seed point is placed in the highly probable area of each sub-image. Curve evolution by using FICVM is applied to each seed point. When the curve evolution is stopped, the shape information is extracted by using the proposed algorithm. We assumed that the curve evolution should converge after some iterations, otherwise the contour might be expanded to the entire background. In this application, if there were changes in more than five pixels after 12 iterations for each contour, the contour was deserted. Note that the sub-images are used only for finding the locations of the seed points.

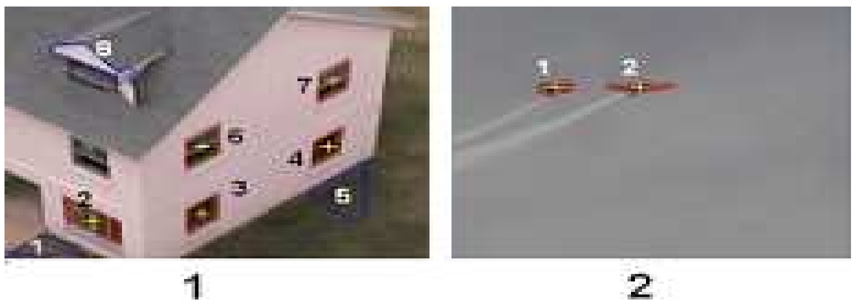


Figure 13: Detection with shape information. 13.1: multiple airplane detection, 13.2: multiple window detection.

Figure 13.1 shows a result image obtained by using by the proposed shape extracting algorithm from a sequence of visual surveillance over buildings. In this sequence, windows in images were assumed to have symmetric sides (e.g., symmetries between the bottom side and the top side of the window or between the left side and right side). Because of this symmetry assumption, the corresponding normalized histogram for window would have similar number between the bins with the π radian difference. Check the rectangles in Figure 11. The first eight rows of Table 4 show the histograms of the contours in Figure 13.1. The first and fifth contours have triangular shapes and the corresponding histograms doesn't have symmetries between the large components with the π radian difference. The first and fifth contour were not identified as windows. The histograms for the second, third, fourth, sixth, and seventh contours have nice symmetries between the bins with the π radian differences. Therefore, they were identified as windows. Curve evolution of the eighth contour was not converged so that it was excluded from the further examination of the geometry. The average processing time of the sequence was 30 frames per seconds with ROI = $240\times$ 240 for total 352 frames.

Figure 13.2 shows two airplane detected by the proposed shape extracting algorithm. We detect cruising airplanes which usually have an elliptic shape in images. The bottom two rows of Table 4 show the histograms of the contours in Figure 13.2.

Orientation	0	$\frac{\pi}{4}$	$\frac{2\pi}{4}$	$\frac{3\pi}{4}$	$\frac{4\pi}{4}$	$\frac{5\pi}{4}$	$\frac{6\pi}{4}$	$\frac{7\pi}{4}$
Contour 1	0.000	0.016	0.151	0.174	0.163	0.012	0.012	**0.477**
Contour 2	0.021	**0.188**	**0.178**	**0.125**	0.031	**0.156**	**0.177**	**0.125**
Contour 3	0.016	**0.266**	**0.203**	0.047	0.031	**0.203**	**0.203**	0.031
Contour 4	0.082	**0.192**	**0.205**	0.027	0.027	**0.260**	**0.205**	0.000
Contour 5	**0.387**	0.047	0.038	0.226	0.019	0.075	0.198	0.001
Contour 6	0.016	**0.274**	**0.194**	0.048	0.016	**0.210**	**0.226**	0.016
Contour 7	0.014	**0.292**	**0.181**	0.028	0.014	**0.278**	**0.153**	0.042
Orientation	0	$\frac{\pi}{4}$	$\frac{2\pi}{4}$	$\frac{3\pi}{4}$	$\frac{4\pi}{4}$	$\frac{5\pi}{4}$	$\frac{6\pi}{4}$	$\frac{7\pi}{4}$
Airplane 1	0.040	0.120	0.060	**0.320**	0.040	0.100	0.060	**0.260**
Airplane 2	0.025	0.013	0.051	**0.392**	0.038	0.013	0.051	**0.418**

Table 4: Histogram information for multiple contours in Figure 13

Based on the analysis in Section 4.3, the histograms for the first and the second contour could be classified to elliptic shapes.

4.5 Shape Energy

This section presents how to model a shape energy function based on feature points and the histograms of the line segments between two feature points. The new shape energy term compensates for the shortcomings in the calculation of the dissimilarity measure with small computational loads. In this method, global shape deformation parameters such as rotation, scaling, and the major axis of the target, can be calculated by using feature points. The local shape deformation can be obtained by matching two histograms corresponding the line segment between two feature points. The contour-based shape energy is,

$$E^k_{cs} = \sum_{i=1}^{M}\sum_{j=1}^{N}(\Upsilon_{sp}(i,j) - \Upsilon_k(i,j))^2, \tag{21}$$

where M is the number of feature points and N=1, ..., 8 is the number in the histogram. Υ is the orientation histogram of the line segment between two feature points.

Then the total shape energy for a measurement is,

$$E_s = \lambda_1 E_{\Psi^s} + \lambda_2 E_c, \tag{22}$$

where $\lambda_1 + \lambda_2 = 1$. The energy E_{Ψ^s} can explain local deformations of the shape but, has some limitations that originate from the simple structure of Ψ^s. The energy E_c can be quickly calculated and can describe more global deformations of the shape that may compensate for the limitations of E_{Ψ^s}.

CHAPTER V

PARTICLE FILTERING

Visual target tracking during flight is a challenging task because the tracker encounters various sources of noise which may cause a loss of tracking (e.g., variations in illumination and the possible presence of obstacles). Therefore, filtering is an essential element in visual tracking to make decisions based on these noise-corrupted signals. The well-known Kalman filter has closed solutions only for linear systems with Gaussian noise. For nonlinear/non-Gaussian problems, particle filtering has received much attention in recent years. Particle filters are simple to implement and flexible to many problems [83]. This chapter provides a brief overview of particle filtering.

Section 5.1 reviews recursive Bayesian estimation, Section 5.2 reviews sequential Monte Carlo methods. Various particle filters in computer vision are presented in Section 5.3.

5.1 Recursive Bayesian Estimation

Recursive Bayesian estimation fits well to tracking problems [83, 70]. A brief review of Bayes estimation is as follows. Assume that $\phi \in \Theta$ is an unknown parameter where Θ is a set of parameters. ξ is a random variable which has a certain probability distribution function, p_ϕ, which is a function of ϕ. Estimation is the evaluation of the unseen ϕ based on the observed ξ. A widely-used estimation method is the Bayesian estimator which is an optimal estimator with respect to a given prior probability density function and a nonnegative loss function. The Bayesian estimator is defined as (Theorem 1.1 in Chapter 4 of [52]):

$$S_\pi(\xi) = \underset{S(\xi)}{arg\ min} \int L(\phi, S(\xi))p(\phi|\xi)\ d\phi, \tag{23}$$

where $S(\xi)$ is an estimator; $L(\phi, S(\xi))$ is a nonnegative loss function, and $p(\phi|\xi)$ is the posterior probability distribution function (PDF). Note that the Bayesian estimator can be calculated in the case when $p(\phi|\xi)$ is known in equation (23) [53].

Consider a discrete dynamic system model assuming the Markov process,

$$\mathbf{x}_{k+1} = f_k(\mathbf{x}_k,\ \varpi_k), \tag{24}$$

$$\mathbf{y}_k = h_k(\mathbf{x}_k,\ \nu_k)\ , \tag{25}$$

where $f_k : \mathbf{R}^n \times \mathbf{R}^m \rightarrow \mathbf{R}^n$ is the state transition function at time step k, $h_k : \mathbf{R}^n \times \mathbf{R}^r \rightarrow \mathbf{R}^p$ is the measurement function, $\mathbf{x}_k \in \mathbf{R}^n$ is the state vector, $\mathbf{y}_k \in \mathbf{R}^p$ is the measurement vector, and $\varpi_k \in \mathbf{R}^m\ and\ \nu_k \in \mathbf{R}^r$ are zero mean, white noise sequences independent of the past and current states. In addition, $\varpi_k\ and\ \nu_k$ are independent, and their distributions are known. The equation (24) describes how the state vector evolves, and the equation (25) relates the state vector and the observation vector at time k. To estimate $\mathbf{x}_k$ sequentially, the Bayesian estimator in equation (23) can be extended to dynamical systems, namely the recursive Bayesian estimator, in which the state estimate is updated upon the arrival of each measurement [70].

The recursive Bayesian estimator calculates the posterior PDF recursively in two stages: a *prediction phase* and an *update phase* [56, 58, 70]. The prediction stage propagates the state vector $\mathbf{x}_k$ according to equation (24) and the update stage corrects the predicted state based on the current measurement $\mathbf{y}_k$. Let $\mathbf{D}_k = \{\mathbf{y}_i : i = 1, \ldots, k\}$ be the set of observations until time step k. Assuming that the prior PDF $p(\mathbf{x}_1|\mathbf{D}_0)$ is known, the PDF for the prediction step is obtained via the Chapman-Kolomogorov equation [70]:

$$p(\mathbf{x}_k|\mathbf{D}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1}|\mathbf{D}_{k-1})\, d\mathbf{x}_{k-1}\ , \tag{26}$$

where $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is obtained by equation (24). When the measurement $\mathbf{y}_k$ is available, the posterior PDF is obtained via Bayes' rule [46]:

$$p(\mathbf{x}_k|\mathbf{D}_k) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{D}_{k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{D}_{k-1})d\mathbf{x}_k} \tag{27}$$

where $p(\mathbf{y}_k|\mathbf{x}_k)$ is the likelihood PDF.

Other than some simple cases (e.g., linear time-invariant models), it is generally difficult to obtain an analytical solution for equation (27) [55]. However, experimental solutions for the more difficult problems can be solved by means of Monte Carlo methods. Monte Carlo methods are stochastic simulations that statistically sample experiments from a model utilizing random numbers [47]. Muller [54] showed that Monte Carlo methods can be used in the Bayesian analysis of general dynamics to approximate any nonlinear and/or non-Gaussian systems.

5.2 Sequential Monte Carlo Methods

Particle filtering is a sequential Monte Carlo method in which the posterior PDF is approximated by the importance weights of samples [68, 70]. For nonlinear/non-Gaussian problems, particle filtering has received much attention in recent years after the pioneering work of Gorden *et al.* [56]. The main drawback of particle filtering was the high computational cost associated with Monte Carlo methods. However, remarkable progress in computer technology makes particle filtering an attractive in many applications.

A brief review of the particle filter proposed in [56] is as follows. The objective of the particle filter is to approximate $p(\mathbf{x_k}|\mathbf{D}_k)$ by using samples and to generate new samples from the PDF. The authors in [55] showed that a posterior sample can be generated from a prior sample via the weighted bootstrap approach [48, 49, 60] based on the Bayes theorem. Assume that we have N samples generated from prior knowledge given by $p(\mathbf{x_1}|\mathbf{D}_0)$. In the prediction step, the state of each sample is propagated by equation (24). In the update step, the likelihood probability $p(\mathbf{y}_k|\mathbf{x}_k)$ is obtained upon the arrival of the measurement $\mathbf{y}_k$. This update step follows the weighted bootstrap approach. The normalized weight of each sample is calculated as,

$$\tilde{\omega}_k^i = \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)}{\sum p(\mathbf{y}_k|\mathbf{x}_k^i)} \ , \tag{28}$$

```
1. Prediction Step:
    - Draw N samples by $\mathbf{x}_k^i \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$, where i=1,...,N
2. Update Step:
    - Weight each sample by $\tilde{\omega}_k^i = p(\mathbf{y}_k|\mathbf{x}_k^i)$
    - Normalize each weight by $\omega_k^i = \tilde{\omega}_k^i \,/\, \sum_i \tilde{\omega}_k^i$
    - Resample by $\{\mathbf{x}_k^i, \omega_k^i\}_{i=1}^N$
```

Table 5: Sampling-Importance Resampling (SIR) algorithm

where $\tilde{\omega}_k^i$ is the sample weight. The weights approximate the posterior PDF. Hence the posterior PDF is approximated as [56],

$$p(\mathbf{x}_k|\mathbf{D}_k) \approx \sum_{i=1}^{N} \omega_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) \ . \tag{29}$$

where ω_k^i is the normalized weight for the i-th sample. Then, the *resampling* step is performed in which a new set of samples are generated from the obtained $p(\mathbf{x}_k|\mathbf{D}_k)$ [56]. The *resampling* step fits well in a sequential estimation scheme since the important samples in the previous time step are more likely to be selected in the next time step. *Resampling* eliminates samples with low importance weights and multiplies samples with high importance weights, which is more effective than just a random sampling. However, *resampling* causes loss of diversity among the particles; that is known as sample impoverishment [70]. The above steps comprise one iteration of the particle filter and also is known as sampling importance resampling (SIR) algorithm. Table 5 summarizes the generic process of SIR particle filtering.

Gordon *et al.* [56] introduced particle filtering to estimate nonlinear/non-Gaussian Bayesian state estimation as a type of bootstrap filtering. Avitzour [57] applied bootstrap filtering to the simple problem of multi-target tracking. Gordon [59] applied a hybrid bootstrap filter for the tracking of multiple targets using information from multiple sensors. Particle filtering is used in the problem of determining a mobile robot's pose with unknown starting position [86]. Other work and applications of particle filters are referred to [68, 70, 103]

5.3 Particle Filtering in Computer Vision

Particle filtering is a rapidly growing area in the field of computer vision. Major applications of particle filters in computer vision include for tracking in clutter, for tracking multiple targets, and for multiple model tracking.

Israd and Blake [58, 63] used particle filtering for contour tracking by means of the CONDENSATION filter. Here, "CONDENSATION" stands for *conditional density propagation.* The conditional PDF or posterior pdf is represented by the weights of N samples and is propagated over time. The filter showed robust results in tracking the outline of an object in dense clutter. However, the contour tracker only estimates the global motion (affine shape transformations) of the target's shape so that local deformation can not be estimated.

Particle filters fit well to multiple target tracking problems because of the ability of handling multi-modal PDF. However, there are several practical issues such as:

- the dominant peak problem,
- occlusion between the targets,
- an appearing/disappearing target.

First, the SIR or CONDENSATION algorithm can cause the dominant peak problem in which the PDF may increasingly bias towards the object with a dominant PDF as the estimation progresses over time [81]. To prevent the dominant peak problem, several extensions of the classical (e.g., SIR and CONDENSATION) particle filters have been proposed; Mixture particle filters assume that the posterior PDF is composed of multiple components and each component is assigned a fixed number of samples at every iteration [83, 84, 104]. Therefore, a certain number of samples corresponding to a component are always available even if the posterior PDF of the component is very low. Another approach for this problem is to construct Voronoi cells [79] within the posterior PDF and each cell represent one mode [69, 81].

Second, occlusion between the targets or clutter increases the uncertainty of the observation. Several particle filtering schemes have been designed for this problem. MacCormick *et al.* [66] proposed the probabilistic exclusion principle in which one measurement that belongs to a target should not be shared by the other target. This principle resolves the conflict for the origin of the measurement caused by the occlusion between two targets. Tao *et al.* [65] defined a hierarchial structure of samples to correctly track multiple targets in complicated interactions. The authors proposed a sampling algorithm in which possible configurations of targets are generated as the results of individual object dynamics. Tweed and Calway [69] introduced bindings (subordination) amongst particles and a hierarchial sampling algorithm that enables multiple occlusions to be handled in a natural way within the standard CONDENSATION framework.

Third, the target can be out of the camera's field of view during the course of tracking because of limited field of view, and the number of targets in a given image may be unknown in advance. Tracking target, in this nontrivial situation and estimation of the number of targets are important task in multiple target tracking. For the problem of a newly appearing (or disappearing) target, a number of models has been proposed such as the spatial birth and death model [65] and the initialization or re-initialization density model [64, 67].

Sometimes, during the course of tracking one set of models (system and observation models) may not be enough to describe various conditions of the target. In these cases, particle filtering may define multiple models and predicts and updates samples according to the model of each sample. For these cases, an additional sampling step is required to generate a model for each sample. This approach may be categorized as multiple model tracking. Multiple model particle filtering schemes have been applied in various problems such maneuvering target tracking [85], mixed state CONDENSATION tracking [82], track-before-detect (TBD) problem [62]. The TBD problem occurs when the existence of a target is uncertain because of the finite resolution of the measuring sensor.

For the real-time tracking problem, Isard *et al.* [64] utilized importance sampling to improve the efficiency and speed of sampling. An importance functional is formed by using a simple blob tracker, and it helps to get a better approximation by combining it with prior information in the prediction step. Then, a more complex and accurate contour tracker is used in the update step. The speed of this algorithm make it applicable to real-time applications.

For the problem of deformable targets, Rathi*et al.* [72, 81] formulated a particle filtering algorithm in the geometric active contour framework. This method can estimate detailed local deformation as well as global motion of objects in images. Furthermore, this method can handle topological changes of the contour by the nature of level set methods.

CHAPTER VI

PARTICLE FILTERS AND GEOMETRIC ACTIVE CONTOURS

In computer vision, particle filters that incorporate shape information of the target have become an emerging area of research because the system model and/or the observation model can easily be nonlinear/non-Gaussian in these cases [24, 64, 69, 83, 84, 81]. However, the computational burden associated with particle filters prevents them from performing at full capacity in real-time visual tracking applications. Note that, in general, the accuracy of estimation increases as the number of particles increases in particle filtering. This thesis has considered the problem of reducing the processing time of particle filters for real-time applications. Furthermore, particle filtering has extended to solve various problems encountered in our applications such as the tracking of appearing/disappearing targets and tracking multiple targets.

Section 6.1 presents particle filtering in the geometric active contour framework. Subsection 6.2.1 describes some problems encountered in our specific applications. Applications of particle filtering are presented in the following sections: air-to-air vision-based tracking in Section 6.3, air-to-ground vision-based tracking in Section 6.4, Near-horizon vision-based tracking in Section 6.5, multiple model tracking in Section 6.6, and multiple target tracking in section 6.7.

6.1 Particle Filtering in The Geometric Active Contour Framework

We begin by briefly describing the particle filter in the geometric active contour framework introduced in [81]. The authors in [81] proposed a particle filter that

- Prediction step:
  - Generate samples $\{\mathbf{A}_k^i, \widetilde{C}_k^i\}$ using $p(\mathbf{A}_k^i|\mathbf{A}_{k-1}^i)$ and $\widetilde{C}_k^i = \mathbf{A}_k^i(C_{k-1})$.
  - Evolve $\widetilde{C}_k^i$ for L times to obtain C_k^i on a given image I_k.
- Update step:
  - Calculate $\omega_k^i = \lambda_{cv}\frac{e^{-E_{CV}}}{\sum_{j=1}^N e^{-E_{CV}}} + \lambda_s(1 - \frac{E_S}{\sum_{j=1}^N E_S})$ and normalize the weights by $\tilde{\omega}_k^i = \omega_k^i / \sum_{i=1}^{N} \omega_k^i$, where $E_S = \int_\Omega (\Psi_1 - \Psi_2)^2 \frac{h(\Psi_1)+h(\Psi_2)}{2}\, dxdy$.
- Re-sampling step

Table 6: Basic particle filtering in the geometric active contour framework.

incorporates geometric active contours. The state vector is $\mathbf{x}_k = [\mathbf{A}_k, C_k]$, where $\mathbf{A}_k$ is the six-dimensional affine motion parameter vector for an active contour C_k at time k [81]. The C_k is embedded in the higher dimensional function Ψ introduced in Chapter 2. In the prediction step, $\mathbf{A}_k$ is propagated according to the affine motion transition probability $p(\mathbf{A}_k|\mathbf{A}_{k-1})$, and the contour C_k is transformed by the predicted affine motion parameters $\mathbf{A}_k$ as $\widetilde{C}_k = \mathbf{A}_k(C_{k-1})$ where $\widetilde{C}_k$ is the predicted contour. Then, the contour $\widetilde{C}_k$ is evolved over a small number of iterations to follow changes in the shape of the target in the image by using the Chan-Vese active contour model. In the update step, the weight of each sample is calculated by the combination of the energy term from the Chan-Vese's active contour model (equation (9)) and the energy term from the shape prior (equation (19)) [77]. The major advantage of this method is that local deformation as well as the global motion of the contour can be measured because the active contour can track various shape changes. New samples are generated by the re-sampling step like in the general particle filtering process. Table 6 shows the algorithm of particle filtering in the geometric active contour framework, and refer

[81] for detailed discussion for the algorithm.

6.2 Particle Filtering in the Blob-FICVM Framework

This section presents the basic algorithm of the particle filter for our applications, and shows how to improve the computational speed of the algorithm. In principle, we follow particle filtering in the geometric active contour framework [81]. In addition, this thesis proposes an advanced blob tracking scheme in which a blob contains shape prior of the target. This scheme simplifies the sampling process and quickly suggests the samples which have high probabilities of being the target as in [64]. The contour evolves only on these samples for more detailed state estimation. The curve evolution in the contour tracking is realized by the fast implementation of the Chan-Vese model (FICVM) presented in Chapter 3. The dissimilarity measure is calculated by the contour based shape modeling method in Chapter 4, and shape prior is updated when it satisfies certain conditions. This new method is named as the particle filtering int the Blob-FICVM framework.

This thesis employs blob tracking as in [64] within the particle filtering in the geometric active contour framework [81] to be more effective in real-time applications. Three major modifications of the new algorithm with respect to the particle filter in [81] are:

1. The blob tracking approach is used in the prediction step. N rectangular blobs containing the shape prior contour (C_{k-1}^{sp}) at time k is generated in the prediction step. The shape prior contour C_k^{sp} is estimated and updated if the candidate contour satisfies given conditions.

2. The original Chan-Vese's algorithm used for segmentation is replaced by the fast implementation of the Chan-Vese's method (FICVM) introduced in Chapter 3.

3. The contour-based shape difference measure is added to the dissimilarity measure as in equation (22).

The first modification is related to the construction of Ψ^s. To carry out curve evolution for each sample, the higher dimensional functional Ψ^s should be constructed N times. Although the FICVM utilizes a simple discrete structure of Ψ^s which can be constructed quickly, constructing Ψ^s N times is still computationally expensive. For the N times Ψ^s constructions, the simplest and fastest way while maintaining shape information of the target, may be by using the shape prior contour C^{sp}_{k-1} and by using two-dimensional translational motion. In this case, only one Ψ^s is required to be established based on C^{sp}_{k-1} for the first sample, and then the Ψ^s can be copied and translated for other N-1 samples. Note that if the variations in rotation and scaling are considered in addition to the translation, then N different constructions of Ψ^s are required.

The concept of blob tracking can be established in this step. In general, blob trackers use a simple structure which is easy to construct and multiply [64]. They can quickly but roughly extract useful data from a given image. This thesis defines a blob which is a small-windowed Ψ^s based on the shape prior contour C^{sp}_{k-1}. The samples are predicted by only translational motion of the blob. This blob contains more shape information than the ones in the other blob trackers as in [64, 101], but it has less variety than the contour transformed by the affine motion parameter vector as in [81]. With this simple prediction step, the tracker can quickly find samples which have high probability of being the target based on the image information (without curve evolution). Only for these highly probable samples, the curve evolutions are applied, and the dissimilarity measures are calculated.

For the second modification, the improvement in computational time of the segmentation using FICVM was presented in Chapter 3. Obviously, the FICVM reduces the computational time of curve evolution for each sample in particle filtering from the analysis of the algorithm. The third modification is that the contour-based shape energy compensates for the shortcomings originated from the FICVM in the calculations of dissimilarity measure; See the introduction of Chapter 4 for the shortcomings. For those samples having low probabilities, the dissimilarity measures are assumed to have uniform probability distribution.

The details of the new algorithm are described as follows: assume that we have obtained the initial contour for the shape prior C_0^{sp} and the initial affine parameter vector $\mathbf{A}_0$, using the FIVCM without particle filtering. Note that this thesis still use the affine motion parameter notation $\mathbf{A}$ without loss of generality. Note, since the prediction step uses only translational motion, the affine motion parameter vector is $\mathbf{A}_k = [x, y, 0, 0, 0, 0]^T$. The state vector is $\mathbf{x}_k = [\mathbf{A}_k, C_k^{sp}]$. In the prediction step, the affine motion parameter vector of each sample is propagated by the affine motion parameter transition probability $p(\mathbf{A}_k|\mathbf{A}_{k-1})$. The contour for each sample transforms its shape by $\widetilde{C}_k = \mathbf{A}_k(C_k^{sp})$. This prediction step generates the blob containing Ψ^s. Then the image dependent energy functional is calculated. The curve evolution by using the FICVM is applied L times to the highly probable samples. It means that $C_k = f_{CE}(\widetilde{C}_k, I_k)$ where f_{CE} is curve evolution function and I_k is the image at the time k. In practice, we determine the highly probable samples by $|c_i - c_o| \geq \varepsilon$, where c_i is the average value (e.g., intensity) of inside the contour, c_o to the opposite; See equations (10) and (11) for c_i and c_o.

In the update step, the dissimilarity between C_{k-1}^{sp} and C_k^i is calculated. Then the importance weight of each sample is calculated. The likelihood PDF, $p(\mathbf{y}_k|\mathbf{x}_k^i)$ is proportional to the product of $p(\mathbf{y}_k^{CV}|\mathbf{x}_k^i) \triangleq \mathrm{e}^{-E_{CV}(\Psi^s)}$ and $p(\mathbf{y}_k^S|\mathbf{x}_k^i) \triangleq \mathrm{e}^{-E_S(\Psi^s)}$. $E_{CV}(\Psi^s)$ is the energy from the Chan-Vese model given by equation (9), and $E_S(\Psi^s)$ is shape energy given by equation (21). Note that this thesis always considers the dissimilarity term in the weight calculation, but in [81] this is only applied to cope with occlusions. This is because shape information is very important, since the color

```
Initialization step:
    . Obtain the initial contour for shape prior $C_0^{sp}$ and the affine parameter
      vector $A_0$ from the results of segmentation process without
      particle filtering in multiple frames.
    . Initialize N samples using $C_0^{sp}$ and $A_0$.
Filtering steps:
  For i = 1 to N
      • Prediction step:
          . Propagate the affine motion parameter vector using $p(\mathbf{A}_k^i|\mathbf{A}_{k-1}^i)$,
          . Transform by using $\widetilde{C}_k^i = \mathbf{A}_k^i(C_k^{sp})$ and construct the corresponding
            blob and calculate $E_{CV}$.
          . Evolve $\widetilde{C}_k^i$ for L times to obtain $C_k^i$ if the sample has large $e^{-E_{CV}}$.
      • Update step:
          . Calculate $E_S$,
          . Calculate $\omega_k^i$ using equation (30).
  End of for loop.
  . Calculate the normalized weights by $\tilde{\omega}_k^i = \omega_k^i / \sum_{i=1}^{N} \omega_k^i$,
  . Update the shape prior contour $C_k^{sp}$ if the candidate contour satisfies
    given conditions.
  Resampling step
```

Table 7: Basic particle filtering in the Blob-FICVM framework.

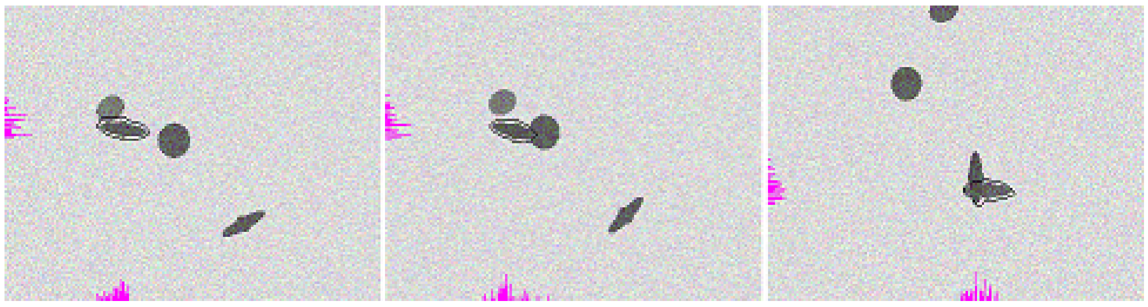


Figure 14: Sequences of ellipse tracking using the Blob-FICVM particle filtering method

information may not be available and the target is very small and thin in the most of our applications.

The two likelihood PDFs are assumed to be mutually independent. Hence, the weight of each sample is calculated as given in Rathi *et al.* [81] by,

$$\omega_k^i = \lambda_{cv} \frac{e^{-E_{CV}}}{\sum_{j=1}^{N} e^{-E_{CV}}} + \lambda_s \left(1 - \frac{E_S}{\sum_{j=1}^{N} E_S}\right), \tag{30}$$

where $\lambda_{cv} \geq 0,\ \lambda_s \geq 0$, and $\lambda_{cv} + \lambda_s = 1$. Then, the sample which has the maximum a posterior (MAP) probability is selected from the approximated posterior PDF as the estimator for time step k. If C_k^i corresponding to the MAP sample satisfies pre-determined conditions, the shape prior contour C_{k-1}^{sp} is updated to C_k^i corresponding to the MAP sample. The conditions are the high contrast between the target and background, and the temporal coherence between C_{k-1}^{sp} and C_k^i belonging to the MAP sample. C_k^{sp} is used as the shape prior in the next frame. Table 7 shows the basic particle filtering algorithm in the Blob-FICVM framework. Figure 14 shows three frames for tracking results of an elliptic-shaped target using the algorithm in Table 7. This tracker could track the target in the presence of partial occlusions. The pink bars on the bottom and left wall of the image show the marginal posterior PDF in the horizontal and vertical axis.

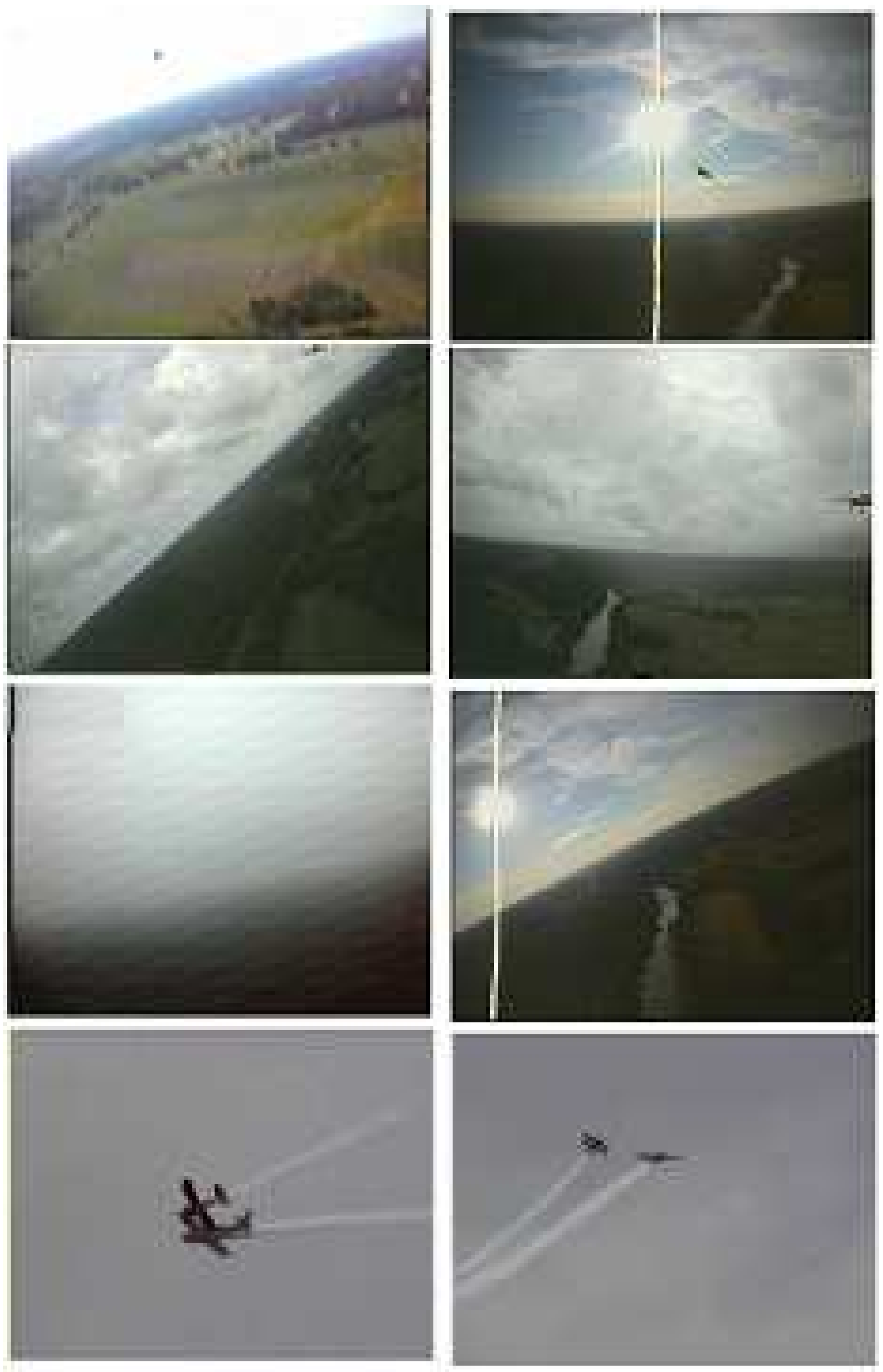


Figure 15: 1 and 2: the effect of direct sunlight, 3 and 4:out-of-the camera's field of view, 5 and 6:low signal to noise ratio, 7 and 8:the presence of multiple targets.

6.2.1 Typical Problems in Our Applications

Before moving on to the applications of particle filtering to our research, let us address some typical problems frequently encountered during flight tests which may cause a loss of tracking. The basic setup for the tests consists of a leader-follower formation flight. Details of the formation flight are described in Chapter 7. Some typical problems for visual tracking during flight may be categorized as follows:

- Occlusions in a dense cluttered environment,
- Low signal-to-noise ratio,
- Problems due to targets lying out of the camera's field of view,
- The presence of multiple targets.

Figures 15.1 and 15.2 show the effect of direct sunlight in an image which can make the target disappear from the image for several seconds. Because the wings in these images appear to be very thin, they are easily disrupted by the background. Figures 15.3 and 15.4 show cases for when the target goes out of the camera's field of view. Figures 15.5 and 15.6 show cases for when the signal-to-noise ratio is very low. Finally, Figures 15.7 and 15.8 demonstrate the presence of multiple targets. In general, these sources of error come together and make it hard to track a target.

6.3 *Air-to-Air Visual Tracking*

The first application of particle filtering in our project has been to track a target that stays above the horizon. The basic setting of this application is to track a leader flying aircraft located above the following aircraft. The algorithm in Table 7 tracks the target during most of the engagement between the two aircraft. The most unique aspect of the air-to-air visual tracking problem is that the color information of the target may not be available because the target is shadowed by sunlight and the two aircraft are distinctly separated ($\approx$ 100 feet) from each other in order to avoid collisions. As a result, shape information becomes important in this application.

The initial contour for the shape prior, C_0^{sp}, is obtained from clean images. The variations in the rotation and the scaling of the target are observed to be much smaller than its translation in the image plane. This is because the target is a rigid body, and the relative motion between the target and the follower has more effect on the translation than the rotation and the scaling of the target in the image plane. Therefore, the propagation equation only considers the translational motion of the target which is modeled as a two-dimensional random walk. In other words, the prediction of the affine motion model is $p(\mathbf{A}_k|\mathbf{A}_{k-1}) = \mathcal{N}(\mathbf{A}_{k-1}, \Lambda)$, where $\mathcal{N}$ denotes the Gaussian distribution, and $\Lambda = diag(\sigma_x^2, \sigma_y^2) = (7, 7)$ in our typical tracking sequences.

In the update step, $\lambda_{cv} = 0.5$, $\lambda_s = 0.5$, and $L = 5$. Only those samples that have $|c_i - c_o| \geq (\varepsilon = 0.1)$ are evolved L times. Otherwise, we would have to spend significant time on samples which are not important. After the contour has been evolved L times, two feature points which correspond to the wing-tips of the target are detected by using the feature point extraction algorithm in Section 4.2. These two points are used to resolve the correspondence problem between C_k^i and C_{k-1}^{sp}. The rotation of C_k^i is calculated using these two feature points. Then, C_k^i is rotated to have the two feature points aligned with the horizontal axis and to have the origin at the center point of C_k^i. From the rotated contour, the orientation histogram introduced in Section 4.3 is obtained. The weight of each sample is calculated by equation (30). C_{k-1}^{sp} is updated if C_k^i corresponding to the MAP sample which has the maximum a-posteriori (MAP) probability for meeting the predetermined conditions. The conditions are a large difference between c_i and c_o (equation (10)) after the curve evolution, which represents a high contrast between the target and the background, and a small difference between C_{k-1}^{sp} and C_k^i belonging to the MAP sample.

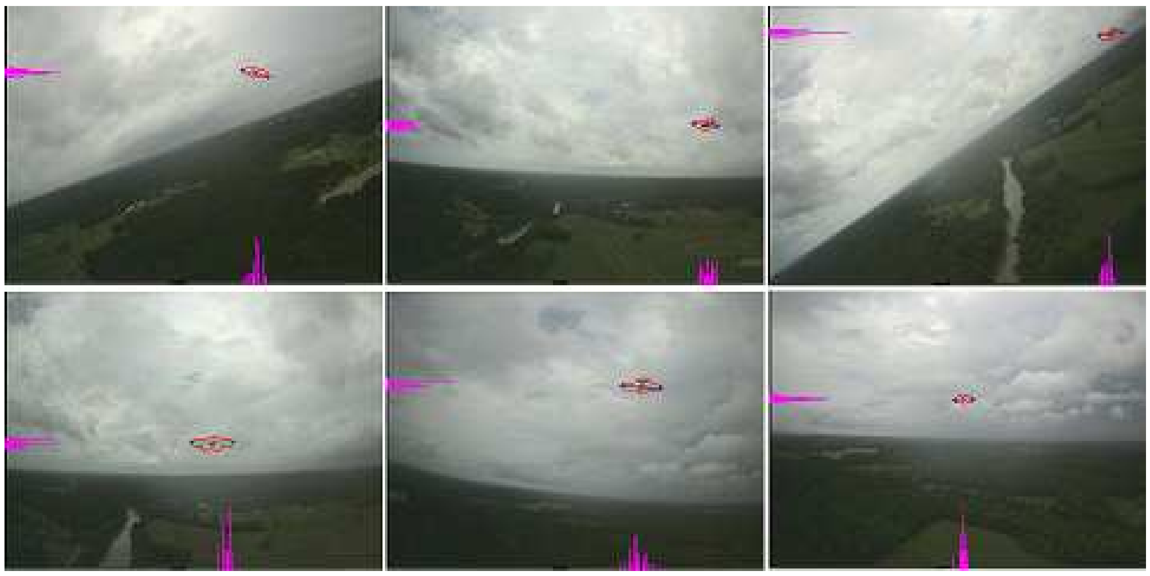


Figure 16: Sequences of air-to-air visual tracking mission by using particle filtering.

Figure 16 shows some frames for a sequence of air-to-air video applied the particle filter. The number of samples is 50 and the average processing time of the particle filter is 27.58 frames/seconds. The computing system is a Intel(R)Xeon(TM) with 3.19 GHz of CPU, 2 GB of RAM. More detailed performance analysis along with GPS/INS data can be found in Figure 32 in Chapter 7. In this simulation test, the tracker could follow the aggressively maneuvering target for about three minutes without any intervention by a user. Note that this automatic tracking ability is obtained by combining the particle filter presented in this section and multi-mode tracking method presented in Section 6.6. The initial position of the tracking is obtained by an user (mouse-clicking) for the convenience of the simulation.

6.4 Air-to-Ground Tracking

The second application was to track a target below the horizon. The target is an aircraft located below the following aircraft. The algorithm in Table 7 is used. The measurements are susceptible to visual noise and occlusions because many other objects exist on the ground. The prediction model uses the same translational random

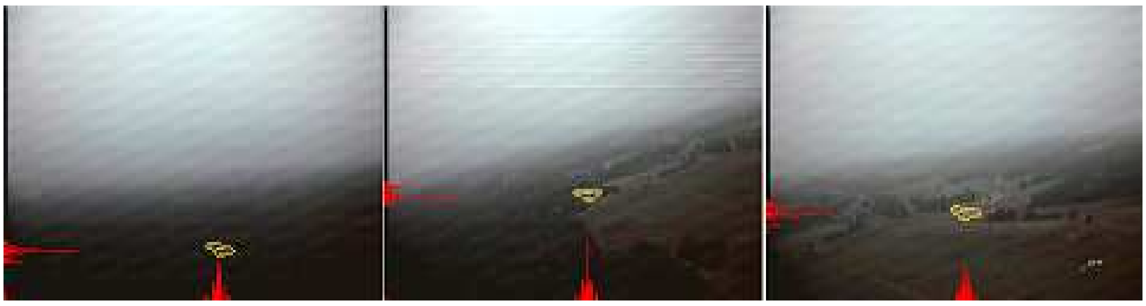


Figure 17: Sequences of tracking from air-to-ground tracking

walk model as in the air-to-air tracking problem with $\Lambda = diag(\sigma_x^2, \sigma_y^2) = (5,5)$. The procedure and the parameter values in this application also remain the same as those in the air-to-air tracking. Figure 17 shows some results of particle filtering in the air-to-ground tracking problem.

6.5 Near the Horizon Tracking

This section removes the limitation that the target stays within a certain part of each image. The main idea of the particle filter in this section is as follows: if the camera doesn't have a clear view of the target, the background is removed from the image. What remains is highly likely to be the moving target. To remove the background, the alignment between two images is required. The tracker uses easy-to-find objects within each given image, such as the horizon and the direct sunlight, for alignment. Note that the direct sunlight in the image is considered as an object. Furthermore, some additional information from these objects can be used in the prediction step to improve the prediction probability.

The third and most difficult application has been to track a target that stays near the horizon or that crosses it. Two major sources of difficulty in this application are as follows:

1. The horizon is the most highly contrasted region for most tracking images. Therefore, it attracts more samples than any other areas for a given image

regardless of the target's presence.

2. In general, the outline of the target becomes hard to distinguish from the background near the horizon. This problem gets worse as the distance between the target and the follower increases because the contour of the target appears small and thin in the image plane. Figure 19 and 20 show some sample images exhibiting this problem.

Because the outline of the target may not be observable, the shape information is excluded from the calculation of the weight for each sample. An additional step is used to estimate the shape of the target. The blob tracking estimates temporal changes and edge information without shape information of the blob.

This thesis uses two assumptions to track the target in these severe conditions:

1. The temporal change of the target is assumed to be much larger than for any other object in the background.

2. The target is assumed to have a slender body.

Consider an ideal static camera problem in which the only moving object in the images is the target. The motion of the target can be easily detected by subtracting the present frame from the previous frame; this is background subtraction method. In dynamic problems, the above situation may not be satisfied. However, the concept of background subtraction can be exploited by assuming slower changes in the background as in the first assumption listed above. This method can improve the performance of tracking by eliminating many outstanding objects (that do not have significant volumes) fixed in the background (e.g., the horizon, a river, and some bright parts of the ground).

First of all, some easy-to-find objects (e.g., the horizon and direct sunlight) are registered. Based on the temporal motions of these objects, this method roughly

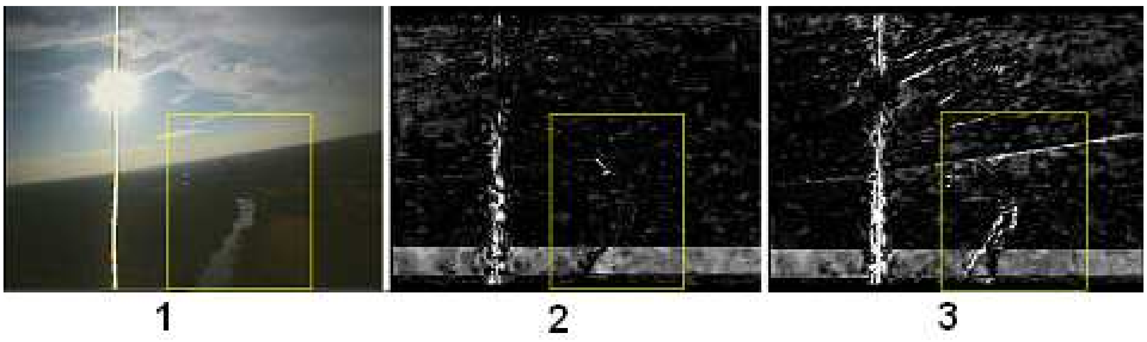


Figure 18: Comparison of temporal differences between a case with correspondence and a case without correspondence. 1: the original image, 2: temporal difference with alignment, and 3: temporal difference without alignment.

aligns the background of the present image with that of the previous image, and calculates the temporal difference between them. This rough alignment procedure considers only the translational motion of the background. In this procedure, two horizontal and two vertical lines are first selected from the previous frame, and these lines are shifted from -5 pixels up to +5 pixels and are subtracted from the lines in the present frame. Then, the horizontal and vertical displacements are found by minimizing the error. The temporal change of the image is modeled by,

$$E_t = \sum_{n=1,m=1}^{N,M} (I_k(n,m) - I_{k-1}^c(n,m))^2. \tag{31}$$

where $I_{k-1}^c(n,m)$ is the aligned gray-scale intensity value of the point (n, m).

Figure 18 compares procedure results of the temporal difference calculation between the case with alignment and that without the alignment. In these images, the larger the temporal difference, the brighter it appear in the image. The river and horizon in Figure 18.2 do not have large temporal difference as in Figure 18.3, but the area corresponding to the target has large differences in Figure 18.2. Although this thesis used a rough alignment, this method is very promising since the attitude and position data of the camera are available in real flight tests so that the alignment procedure can be made much more accurate.

From the second assumption, the slender body of the target produces concentrated large-valued edges around the target in these images. The edge information for each sample is measured by,

$$E_e = \sum_{n=1,m=1}^{N,M} ((I_k(n+1,m+1) - I_k(n,m))^2 \\ +(I_k(n+1,m) - I_k(n,m+1))^2), \tag{32}$$

where $I_k(n,m)$ is the gray-scale intensity value at the point (n, m) and N×M is the size of the blob. Note that in the calculation of equation 20, the horizon and direct sunlight also have large values. Samples nearby the horizon and samples nearby the direct sunlight in the image will have large weights regardless of the actual position of the target. To prevent this event, a penalty term p_{hs} is considered.

First, the probability of being the horizon can be modeled by a one-dimensional uniform distribution such as $P_{horizon} = \mathcal{U}(h)$, where $h = \{y \pm dh|$ the vertical coordinate of the point on the horizon $\}$. Similarly, the probability of being at the location of direct sunlight can be modeled by a one-dimensional uniform distribution such as $P_{sun} = \mathcal{U}(s)$, where $s = \{x \pm ds|$ the horizontal coordinate of the point on the location of direct sunlight in an image $\}$. Remember that the pixel coordinates for the horizon and the direct sunlight are obtained before alignment. In this application, $dh = 1$, $ds = 7$, $P_{horizon} = 0.5$, and $P_{sun} = 1.0$. Therefore, the penalty for a sample located

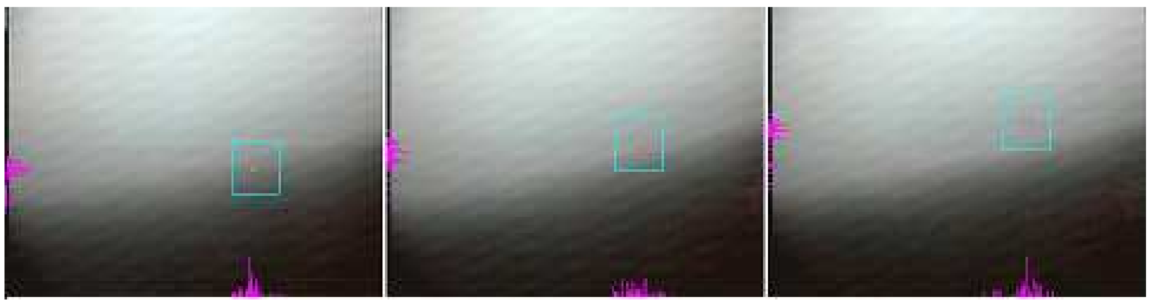


Figure 19: Sequences of tracking a target near the horizon

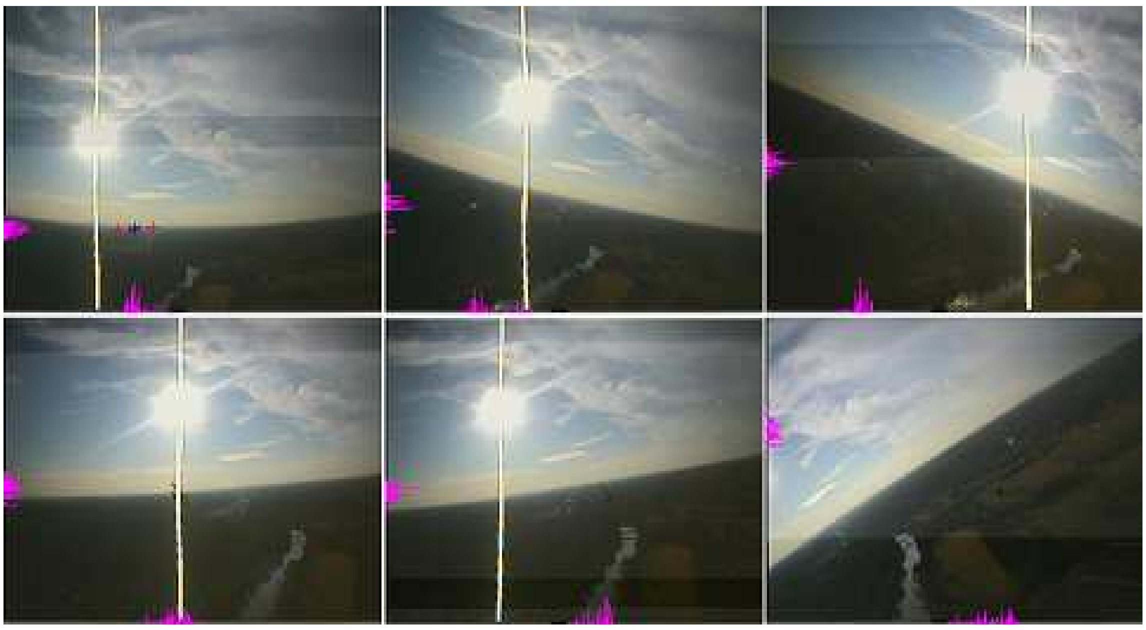


Figure 20: Sequences of particle filtering in tracking the target staying nearby the horizon.

on the horizon or nearby direct sunlight can be modeled as,

$$p_{hs} = 1 - MAX(P_{horizon}, P_{sun}). \tag{33}$$

Multiplying this penalty into the calculation of the weight of the sample prevents the horizon and direct sunlight from attracting most of the samples.

The weight of each sample is calculated as,

$$\omega_k^i = \lambda_e\left(\frac{E_e \cdot p_{hs}}{\sum_{j=1}^{N}(E_e \cdot p_{hs})}\right) + \lambda_t\left(\frac{E_t}{\sum_{j=1}^{N} E_t}\right), \tag{34}$$

where $\lambda_e \geq 0, \lambda_t \geq 0$, and $\lambda_e + \lambda_t = 1$. In the prediction step, the motions of the horizon and the direct sunlight are added to have a better prediction of the samples. In flight tests and in simulation, the actual attitude and position of the camera data can be used for the prediction. Then, the curve evolution is applied only to the sample which has the MAP probability. The shape of the contour is examined only for this sample. If the contour satisfies the predetermined condition, C_k^{sp} is updated.

Figure 19 and Figure 20 show results of using particle filtering for tracking a

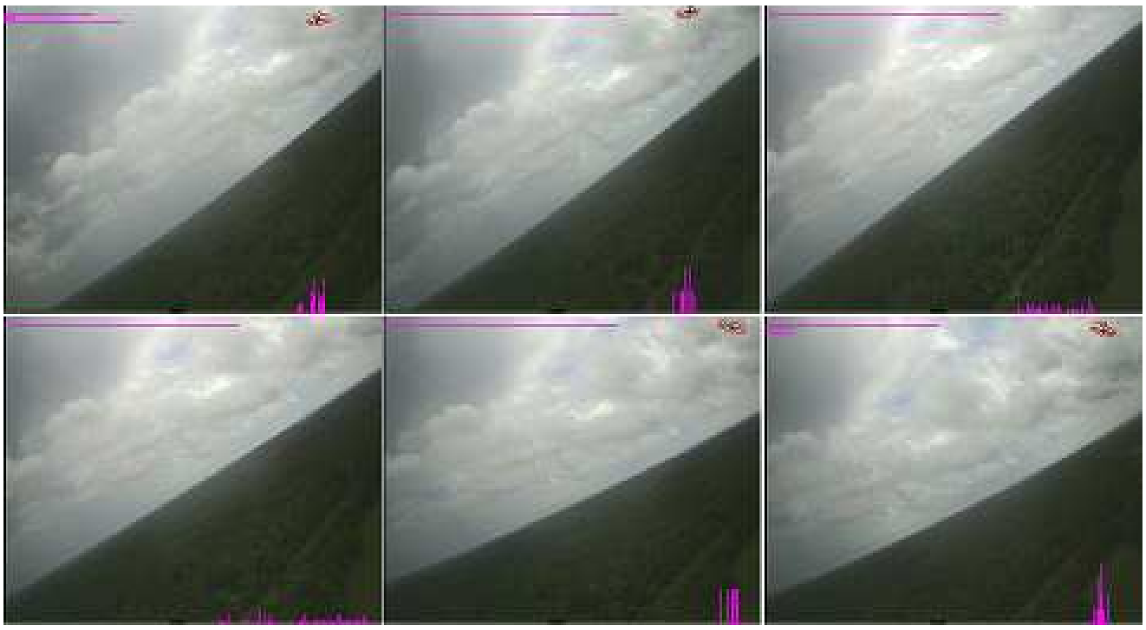


Figure 21: Sequences of the target close to the top-boundary of the image

target that stays near the horizon. The marginal probability densities are multi-modal distributions in these figures.

6.6 *Out-of-the-Camera's Field of View Problem: Multiple Model Tracking*

The particle filtering method proposed in Section 6.1 may not be effective in some cases. One of the important practical issues of visual tracking is that the view of the camera is finite which may result in targets being out of sight. Moreover, the random walk propagation model may predict some samples outside of the image plane as the target moves towards the image boundaries. Obviously, no measurements are available from these samples. In addition, samples which are close to the boundaries may produce incomplete measurements. One solution for these problems is to generate samples until the sample is predicted to be within the image plane. However, this sampling method can take a significant portion of the processing time. With this in consideration, this thesis considers multiple model tracking in which a different model is considered in place of the ordinary tracking model as the target approaches the boundaries of the image. Details and references of multiple model tracking are in [82, 62, 85, 70]. In this application, the tracker switches from the ordinary tracking model to a model which is effective at tracking targets close to the boundaries. Furthermore, the new sampling method is effective at tracking disappearing targets and detecting re-entering targets. Figure 21 shows a target moving close to the top-boundary of the image, then moving completely out of the field of view, and re-entering the image.

In the multiple model tracking problem, a different propagation model can be applied according to the model of each sample. A different observation model may be used as well. This thesis considers two models: the generic model and the boundary model. The generic model represents ordinary tracking when the target stays far from the boundaries and close to the center of the image and this uses the algorithm in Table 7. The boundary model is activated as the target moves close to the boundaries of the image plane. The models are denoted by a two state Markov chain $M_k = \{0, 1\}$ [62]. 0 denotes the generic model, 1 otherwise. Each sample belongs to one of the two models, and sampling using transitional probabilities determines the model of the sample in the next time step. The transitional probabilities denoted by P_b (the boundary model) and P_g (the generic model) are defined as,

$$\begin{aligned} P_b &\triangleq P\{M_k = 1 | M_{k-1} = 0\} = \lambda e^{-\lambda d}, \\ P_g &\triangleq P\{M_k = 0 | M_{k-1} = 1\} = \lambda e^{-\lambda(1-d)}, \end{aligned} \tag{35}$$

where d represents the minimum distance from the boundaries to the position of the sample which has the MAP probability as in Figure 22. λ is a positive constant. Hence, the transitional probability matrix between the two states is given by

$$\prod = \begin{bmatrix} (1 - P_b) & P_b \\ P_g & (1 - P_g) \end{bmatrix}. \tag{36}$$

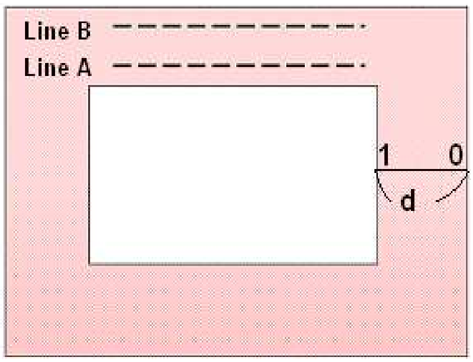


Figure 22: Region of the interest for the appearing/disappearing target.

Once the model of each sample is determined, the corresponding propagation and observation model can be applied. There are four possible classes of samples (particles) as follows:

Generic particles ($M^i_{k-1} = 0,\ M^i_k = 0$): This is a group of particles that continues to stay away from the boundary. The particle filter in the Table 7 is used for these.

Disappearing particles ($M^i_{k-1} = 0,\ M^i_k = 1$): This is a group of particles that moves towards the boundaries. The particles in this group are sampled uniformly along the approaching boundary (Line B in Figure 22) . The observation model only considers E_{cv}, and shape information is ignored.

Appearing particles ($M^i_{k-1} = 1,\ M^i_k = 0$): This is a group of particles that moves towards the center from the boundary. The particles in this group are propagated uniformly along the starting line of the boundary area (Line A in Figure 22). The observation model is the same as for the disappearing particles.

Outside particles ($M^i_{k-1} = 1,\ M^i_k = 1$): This is a group of particles that are out of the field of view. These particles are considered as disappearing particles.

The procedure for the multiple-model tracking is as follows:

1. Model transition: generate a random sample set based on the previous model and the transitional probability matrix $\prod$ [70].

2. Model conditioned filtering: perform particle filtering in the Blob-FICVM framework according to the model of each sample.

6.6.1 Sampling for A Target Complete Outside of The Camera's View

It is possible for a target to lie completely outside of the image plane. Thus no measurements are available in this case. When there is no significant sample to perform the curve evolution on multiple consecutive frames (3 frames in this application), the target state is considered to be outside of the image. In this case, the transition probability matrix is,

$$\prod = \begin{bmatrix} 0.02 & 0.98 \\ 0.02 & 0.98 \end{bmatrix}, \tag{37}$$

where the probability of outside particles is $P(M_{k-1}^i = 1,\ M_k^i = 1)$=0.98, and the probability of disappearing particles is $P(M_{k-1}^i = 0,\ M_k^i = 1)$=0.98. Therefore, most of the particles will be considered as disappearing and to lie outside of the images.

One assumption is made to make sampling more effective for detecting a re-entering target. The assumption is that the target is likely to re-enter somewhere close to the point it had excited. As the number of frames that the target is absent increases, the interval of the uniform distribution increases while remaining centered at the target's last known position. This increases the probability of detection for the re-entering target.

6.7 Multiple Target Tracking

Probability distributions have multi-modality in the multiple target tracking problem. Maintaining multi-modality during tracking is an important issue in this problem because of "the dominant peak problem" mentioned in Section 5.3. This thesis deals

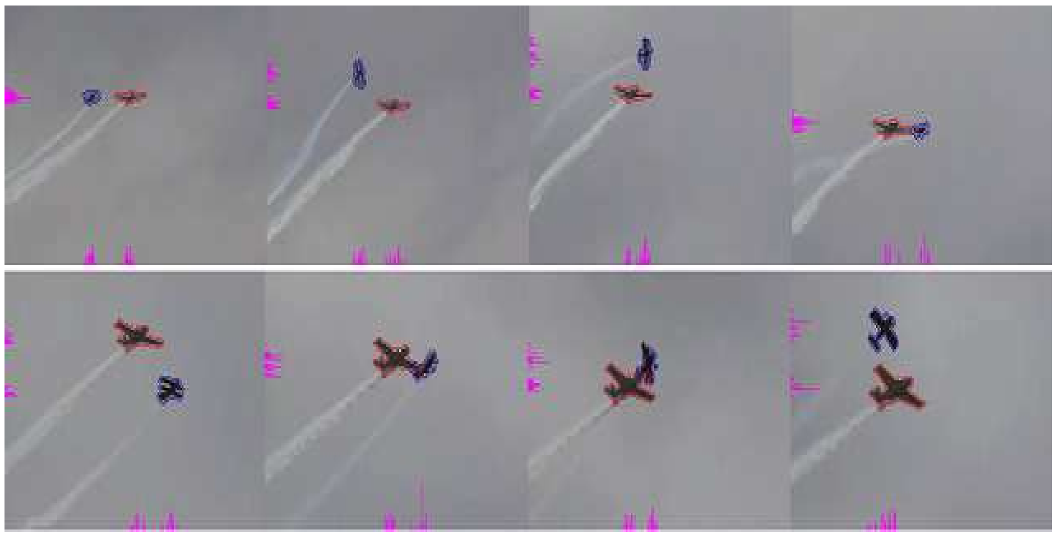


Figure 23: Sequences of multiple target tracking using particle filtering in the geometric active contour framework, 50 samples. The original images from www.metacafe.com.

with the multiple target tracking problem by using mixture particle filtering as given in [83, 84]. In this approach, the posterior PDF consists of multi-component mixture models, and the contributions of each component to the PDF is weighted as given in [83],

$$p(\mathbf{x}_k|\mathbf{D}_k) = \sum_{m=1}^{M} \pi_{m,k}\, p_m(\mathbf{x}_k|\mathbf{D}_k). \tag{38}$$

where $\pi_{m,k}$ is the weight of the m-th component and $\sum_{m=1}^{M} \pi_{m,k} = 1$. The authors in [83] showed that the prediction step and update step can be carried out by each component individually, and each component interacts only in the calculation of the weight for each component $\pi_{m,k}$. Furthermore, the multi-modality is maintained by re-sampling particles in each component individually. Note that even if one mode has relatively lower probability than other modes, a certain number of particles are always assigned for that mode, otherwise all the samples quickly move towards the dominant mode. Furthermore, this filter refines the multi-modality by recomputing the mixture representation and by applying K-means algorithm in multiple times.

The mixture particle filtering approach incorporated with the Blob-FICVM particle filter fits well in multiple-UAV tracking problems, in particular maneuvering ones. Since the shape prior in the Blob-FICVM particle filter is updated, a maneuvering target which constantly changes its shape can be tracked nicely. In addition, because the mixture particle filter can treat multiple modes individually, the new filter can track multiple targets doing various different maneuvers. Figure 23 shows sequences of multiple target tracking by using the mixture particle filtering in the Blob-FICVM framework. In this sequence, one target moves forward with almost a constant speed while the other target doing severe maneuvers. Two targets partially occludes in frames and the particle filter correctly follows each target. The multi-modality is maintained throughout the tracking.

CHAPTER VII

APPLICATIONS

This chapter presents how the computer vision algorithms developed in the previous chapters are applied to our specific applications. Section 7.1 describes formation flight between UAVs. Section 7.2 presents how the computer vision algorithms are used for formation flight. Section 7.3 shows test results obtained from simulations and flight tests.

7.1 Formation Flight

The ultimate goal of this research is to track multiple targets in uncertain complex environments purely using a visual sensor. The ability to track multiple objects in adversarial conditions will open the door to various research opportunities related to UAVs (e.g., formation flight between multiple and multi-purposes UAVs). Figure 24 sketches the mission of the UAV.

The performance of the vision system is tested using the test-bed UAVs of the Georgia Tech (GT) UAV Laboratory which are shown in Figure 25. The basic setup

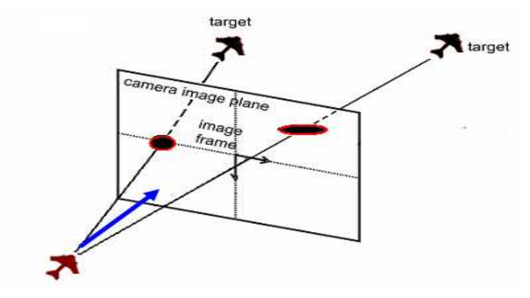


Figure 24: A UAV tracking multiple targets

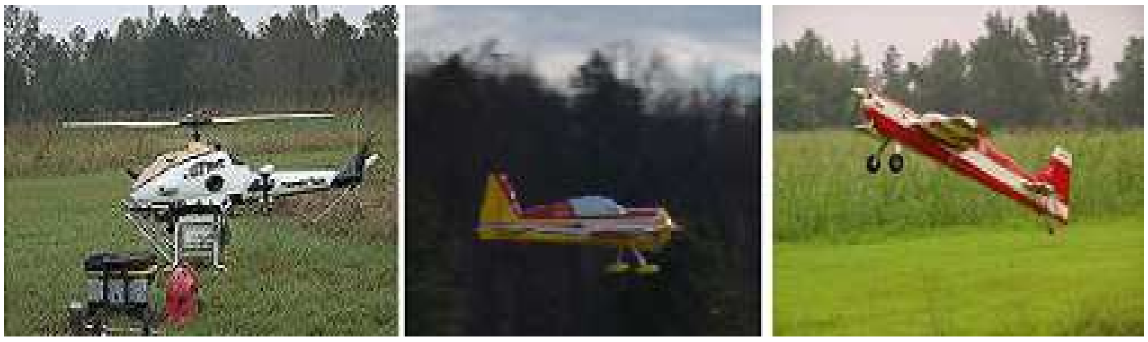


Figure 25: Left: GTMax helicopter. Right: GTEdge fixed-wing aircraft

for a flight test is the leader-follower formation flight in which the follower detects and tracks the leader using only a single vision sensor. The leader aircraft is a 1/3 scale fixed-wing aircraft, the GTEdge 540T, with a global positioning system/inertial navigation system (GPS/INS)-based autopilot. This aircraft can perform various maneuvers given from the ground control station. For the follower aircraft, the GTMax research helicopter was used with on-board image processing until 2006 [23]. Then, it was replaced with a fixed wing aircraft to have dynamics similar with the leader aircraft. In addition, the computing system was upgraded from a 850 MHz CPU and 520 MB of RAM to a 1.6 GHz CPU and 1 GB of RAM.

In flight tests, the follower initially loiters at a remote point outside of the path which the leader fixed wing aircraft is flying. At a certain time, the follower is commanded to maintain a fixed relative position from the leader aircraft, which makes the start of the formation maneuver. The initial relative position is selected as 20 feet below and 100 feet behind the leader to reduce the risk of collision. After this initial positioning maneuver, the follower relies on only the vision sensor. The follower aircraft estimates the states of the leader aircraft by utilizing on-board image processing. The follower determines its relative position with respect to the leader by estimating the relative states with respect to the leader. See Johnson *et al.* [23] for details of the estimation of the relative states. Figure 26 shows a scene from the

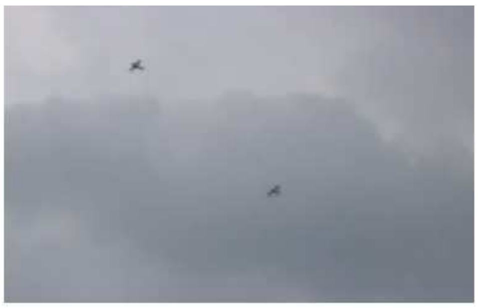


Figure 26: Formation flight between two fixed-wing aircrafts.

formation flight and two aircraft.

The following section presents how the developed computer vision algorithms are used to detect and track the targets. Before moving to the subject of compute vision, let us address a couple of observations we have made during several flight tests. Unlike simulated or indoor images, the surveillance and tracking images have some unique properties as follows.

- Color information may not be reliable.
- Severe variations in illumination can occur.
- Objects can be small and thin.

7.2 Visual Detection and Tracking Algorithms

One of the fundamental requirements of the active vision system is to detect and to track targets in uncertain complex environments. The main objective of the detection step is to quickly search and identify a target in within wide region. The detection process uses the assumption that the tracker has some amount of knowledge about the shape of the target as stated in Section 1.4. The detection step quickly separates the object from the background using segmentation, and then extract the outline of the object. Shape information is extracted from the outline of the object to investigate the proximity of it with the prior shape knowledge of the target. After multiple consecutive identifications of the target, the tracking step is initiated. The shape information and the position of the target obtained from the detection step are used in the tracking step.

Tracking employs recursive processing of measurements of the states of the target. Because the tracker may encounter various situations during tracking, the measurement from the vision sensor may be noisy. The tracker needs robust and efficient computer vision algorithms to cope with these noisy signals. This thesis concentrates on the development of fast and robust computer vision algorithms to detect and track the targets in complicated situations during flight.

The following subsections present detection and tracking algorithms based on the algorithms proposed in the previous chapters. Subsection 7.2.1 presents the algorithms based on fast marching methods. The limitations of the algorithm are presented at the end of this subsection. Subsection 7.2.2 presents those based on the particle filtering in the fast implementation of the Chan-Vese model framework.

7.2.1 Using Fast Marching Methods

7.2.1.1 Detection Step

The detection step is a computationally expensive process since it requires searching for a target in a wide region. To speed up the searching process, fast marching methods are used. Two strategies that reduce computational loads of the methods are proposed in this step. Two strategies are,

1. adjusting the stopping condition,

2. modifying the frequency of heap sorting.

The first strategy is to stop the fast marching right after the sky is completely separated from the ground in a given image. For this moment, the target is assumed to stay above the horizon (this assumption will be removed in a later algorithm). In general, the curve first moves to smooth areas that have small values of gradient of

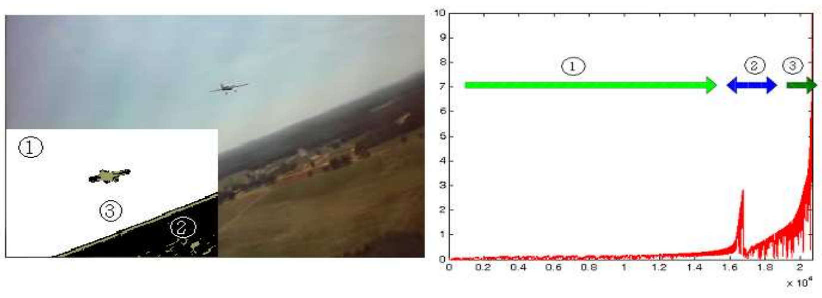


Figure 27: Left: A result from marching methods, Right: magnitude of image gradient with respect to the evolution of the curve. Searching area = 120 by 180 pixels

intensity in the fast marching methods. Figure 27 shows the relationship between the order of fast marching and the values of the gradient of intensity at the corresponding points in the image. As you can see in Figure 27, the curve marches in the order of region 1, 2, and 3. Region 1 corresponds to the sky in the image, region 2 to the smooth area in the ground, and region 3 to the areas that have large values of the gradient of intensity. Therefore, if the fast marching is stopped when region 1 is revealed, the target and the sky would be separated as in Figure 27. Following the motion of the curve at every iteration, the corresponding gradient value is checked. The stopping condition terminates the curve evolution after finding region 1 in Figure 27 by checking the gradient value at each iteration. This method is less sensitive to variations in illumination than when using thresholding or edge detection operators to find the outline of the target. This is because, in general, variations in illumination affect the entire image and the relative value of the gradient of intensity is still large around the target.

For the second strategy, the frequency of heap sorting changes according to the number of points in the heap. As the number of points in the heap increases, the computational cost of the sorting algorithm also increases. In this application, the heap sorting algorithm is called at every iteration until the total number of points in the heap exceeds a given number (500 points). When the number of points in

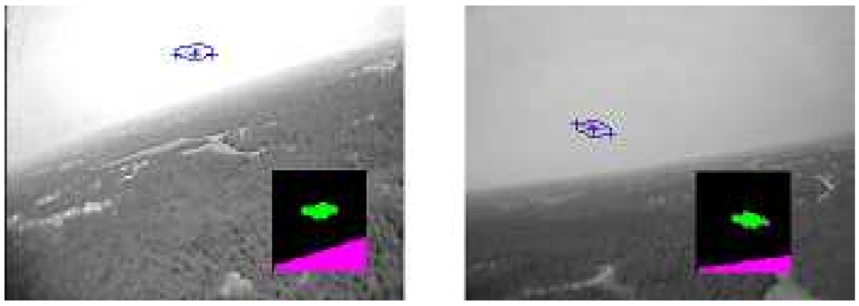


Figure 28: Typical tracking sequences. Average time for tracking 0.07sec, ROI = 80 by 80 pixels

the heap exceeds a given value, the heap sorting algorithm is called only once every several iterations (20 iterations). This strategy improves the speed of the detection step dramatically by not having to sort large amount of data in the heap at every iteration. The size of the region of interest (ROI) is 120 by 180 pixels in this step, and the averaging processing time is 10 frames/seconds.

7.2.1.2 Identification Step

After the evolution of the curve is stopped, the result of this evolution is converted into a binary image. The "Known" set is converted to 0 and the "Far" and the "Trial" sets are converted to 1. From this binary image, several contours are extracted. Then, three feature points (center, left-wing tip, right-wing tip) and several geometric properties such as the aspect ratio of the contour and the span based on the major axis of the target are extracted from each contour. These properties are then used to determine which contour corresponds to the target aircraft. The expected value of the span is calculated based on knowledge of the actual target aircraft span.

7.2.1.3 Tracking Step

Once a target is identified in multiple consecutive frames, the tracking step is initiated. One simple choice of the algorithm for the tracking step is to use the same fast marching method used in the detection step, but with a smaller ROI (80 by 80

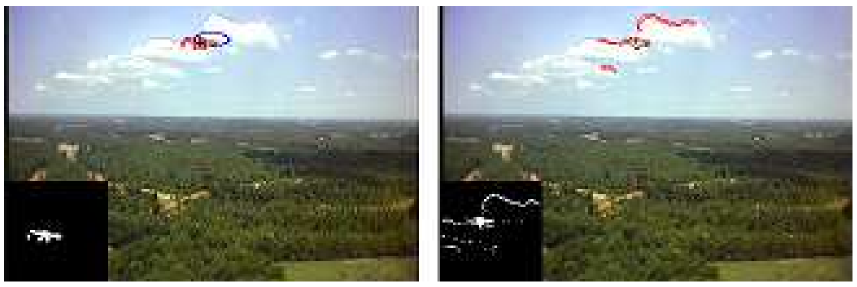


Figure 29: Left: A result from the region-based approach, Right: A result from fast marching methods

pixels). Figure 28 shows typical tracking scenes. Using the proposed algorithm, the UAVs succeeded in performing successful formation flights in flight tests.

7.2.1.4 Limitations

One limitation is that the fast-marching-based tracker is not robust to a noisy input image because fast marching methods are based on edge-based geometric active contours. Edge-based geometric active contours are more susceptible to noise because their external energy terms are constructed by the local gradient of intensity information of the image. Influences of noise can be reduced by applying region-based geometric active contours. For example, Figure 29 shows a comparison between the result from the region-based geometric active contour model (Figure 29, left) and the result from the fast marching method described in the previous sections (Figure 29, right). The region-based geometric active contour model produces robust result while the fast-marching-based method produces spurious contours caused by thick clouds. The blue ellipse in the left figure represents the initial contour.

The second limitation is that the tracker does not effectively use data from the previous frame. The fast-marching-based tracker uses a rectangular with a fixed size. This shape of the ROI limits improvements in the computational speed of the algorithm because the tracker may not effectively use the shape data obtained from previous frames. The final contour in a previous frame contains useful data of the

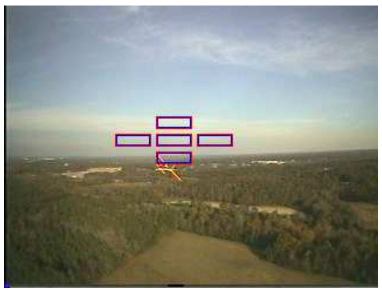


Figure 30: Detection using the fast implementation of the Chan-Vese model

target. If we can use the final contour in the previous frame as the initial contour in the next frame, the curve evolution may converge faster [100].

7.2.2 Using Particle Filtering and The Fast Implementation of Chan-Vese's Model

7.2.2.1 Detection Step

A fast implementation of the Chan-Vese model (FICVM) was proposed in Chapter 3. This new implementation is very effective in real-time tracking problems and it resolves the limitations described in the previous subsection. The computational speed of this implementation has reached 100 frames/seconds in typical tracking scenes for our project while maintaining robustness of the segmentation with respect to local minima. We have designed a new detection scheme based on the FICVM. Taking advantage of the fast speed of the FICVM, we used multiple initial contours in a frame to cover a wide region.

Figure 30 shows the multiple initial contours for the detection of the target. First the center initial contour is tested. Remember that region-based geometric active contours aim to separate the values (e.g., c_i and c_o) from each segmented region as much as possible as the curve evolution progresses. Therefore, if the difference between two segmented region is not large enough, $|c_i - c_o| < \epsilon_d$, after several iterations, then the evolved contour is considered to be part of the background. In addition, if the curve evolution does not converge, this means that the object is too large to be the target and this contour is also considered to be part of the background. Then, the next initial contour is translated to the right from the initial contour at the center. The order of the translation of the initial contour is center-right-left-up-down. At each translation, the center position of the initial contour moves 15 pixels in the vertical direction and 30 pixels in the horizontal direction. If the curve evolution converges at a certain iteration and $|c_i - c_o| > \epsilon_d$, then the contour is sent to the identification step as in the previous subsection.

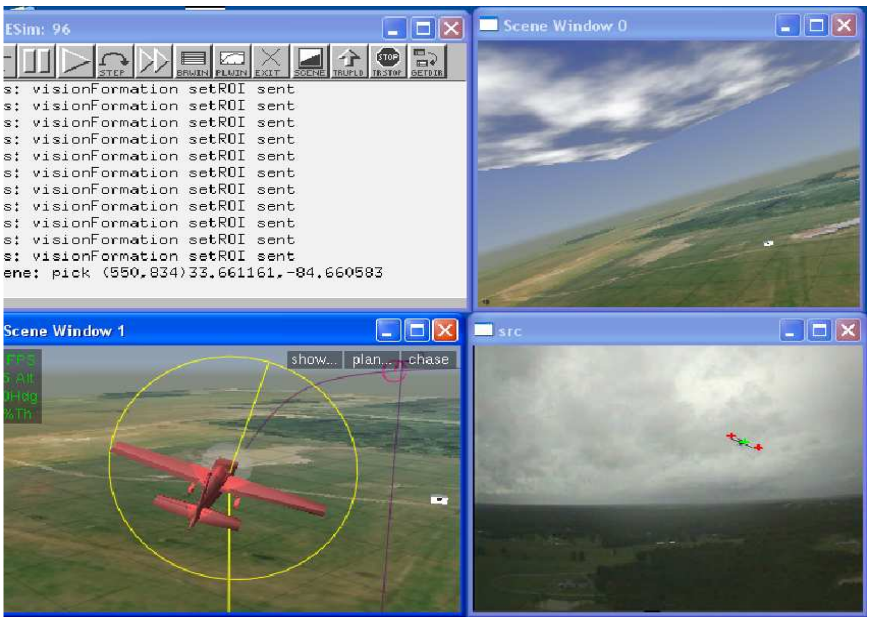
7.2.2.2 Tracking Step

Once a target is identified in multiple consecutive frames, the tracking step is initiated. At the beginning of the tracking step, the contour belonging to the target obtained from the detection step is saved for use as the shape prior. Shape information (feature points, the orientation of the connectivity) is also saved for the calculation of energy from the dissimilarity measure 21. Then, particle filtering takes over in the tracking step. Particle filtering copies the saved contour N times and predicts the new samples in the prediction step. The details of particle filtering are presented in Chapter 5.

7.3 Test Results

7.3.1 Results from Simulation

Computer simulation has been a useful tool for developing computer vision algorithms. Figure 31 is a screen capture of the display of the vision-based flight simulator we have used. This simulator includes two dynamic models and graphics for a 14 foot wingspan fixed wing aircraft, configured as either a leader or a follower. A flight controller and guidance system for each dynamic model is implemented in the simulation [23]. Since the aircrafts use similar dynamical models and control-guidance systems, additional aircraft can be easily added for multiple target tracking.



```
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
s: visionFormation setROI sent
ene: pick (550,834)33.661161,-84.660583
```

Figure 31: Vision-based flight simulator.

The follower aircraft has an onboard camera and its output is simulated by synthetic images. For testing of computer vision algorithms, the simulator provides artificially generated scenes for various flight conditions. For a more realistic simulation, it can use provides streaming videos saved from flight tests. Streaming videos are created by saving several sequences of pure images along with other data such as positions, velocities, and attitudes of the leader and the follower so that the data is time-synchronized with the images. The computer vision algorithms process the images and provide the same type of output that would be expected in an actual flight. An extended Kalman filter estimates the relative states between the two aircraft by using the results from the vision system. Then, the controller determines actuator commands based on the navigation system output and relative position, velocity, and attitude commands [23].

Figure 32 shows the results from the flight simulator using streaming video recorded on July 11, 2007. The algorithm used in this test is the particle filter in Table 7 with multiple mode tracking capabilities. The number of samples was taken to be 50, $(\sigma_x^2, \sigma_y^2) = (7, 7)$, $\lambda_{cv} = 0.5$, and $\lambda_s = 0.5$. In this video, the UAV maneuvers aggressively and appears/disappears multiple times from the camera's field of view. The tracking data was saved for about 182 seconds (the entire video is about 236 seconds long.) The altitude change is about 19.2 feet during the entire maneuver. The computing system is a Intel(R)Xeon(TM) with 3.19 GHz of CPU, 2 GB of RAM.

Figure 32.1 shows the results of target acquisition where 1 represents that the target is acquired and is 0 otherwise. The total target acquisition rate was 84.2 percent. Figure 32.2 displays the trajectory of the leading UAV obtained from GPS in the (NED) inertial frame. Figure 32.3 shows a comparison between the x (vertical) and y (horizontal) image coordinates of the center position predicted from GPS/INS with measurements obtained from the image processor. Here, the data from the GPS/INS is transformed from the inertial frame to the camera frame for the comparison. Figure 32.4 shows the histogram for the computational times for data processing spent in the vision system which includes segmentation, particle filtering, and relative state estimation by EKF. From the histogram, 99.5 percent of all frames took less than 0.05 seconds, per frame and the average computational time is 0.0369 seconds per frame (or ≈ 27 frames/seconds.)

As shown in Figure 32.3, the x-coordinate of the GPS target is out of the range of the image plane in case A and case B. The multi-mode particle filter in Section 6.7 handles the appearing/disappearing target nicely. Except for the appearing/disappearing cases, the particle filtering tracks the target consistently.

7.3.2 Results from Flight Tests

After completing extensive trials in the simulator, we have conducted several flight tests to verify that the proposed vision algorithms can be applicable in real-world tracking problems and to gather data for further analysis. The flight tests have been designed around the leader-follower formation flight described in Section 7.1.

Figure 33 shows the results from the flight test held on July 15, 2006. The tracking algorithm in this test was based on the fast marching method presented in Subsection 7.2.1. The altitude change was about 33.4 feet during the maneuver. Figure 33.1 shows the results of target acquisition. The total target acquisition rate was 95.3 percent. Figure 33.2 displays the trajectory of the leading UAV obtained from GPS in the inertial frame. Figure 33.3 shows a comparison between the x (vertical) and y (horizontal) image coordinates of the center position predicted from GPS/INS with measurements obtained from the image processor. Figure 33.4 shows the histogram for the computational times for data processing spent in the vision system including segmentation, particle filtering, and relative state estimation by the EKF. From the histogram, 97.1 percent of the entire tracking frames took less than 0.1 seconds per frame and the average computational time was 0.0997 seconds per frame (or $\approx$ 10 frames/seconds.) The computing system has 850 MHz of CPU power and 520 MB of RAM. We can see that the processing time with the fast-marching-based tracking algorithm is much slower than the one with FICVM. Remember that the computational times in Figure 32.4 each represent the average from 50 samples.

Figure 34 shows the results from a flight test held on November 11, 2007. The tracking algorithm in the test was used in this test, or the particle filter in Table 7 with multiple mode tracking ability. The number of samples was taken to be 50, $(\sigma_x^2, \sigma_y^2) = (7, 7)$, $\lambda_{cv} = 0.5$, and $\lambda_s = 0.5$. The altitude change was approximately 5.7 feet.

Data was saved at three different rates:

1. 30 Hz: computational time of estimation, position, velocity, acceleration, and attitude of both the leader and follower aircrafts,

2. 20 Hz: results of target acquisition,

3. 10 Hz: the positions of feature points of the leader aircraft in the camera frame.

Figures 34.1,2,3, and 4 show results of the test. The total target acquisition rate was 98.9 percent. From Figure 34.4, the average computational time was 0.0606 seconds per frame (or $\approx$ 17 frames/seconds.) The computing system is a 1.6 GHz CPU with 1 GB of RAM.

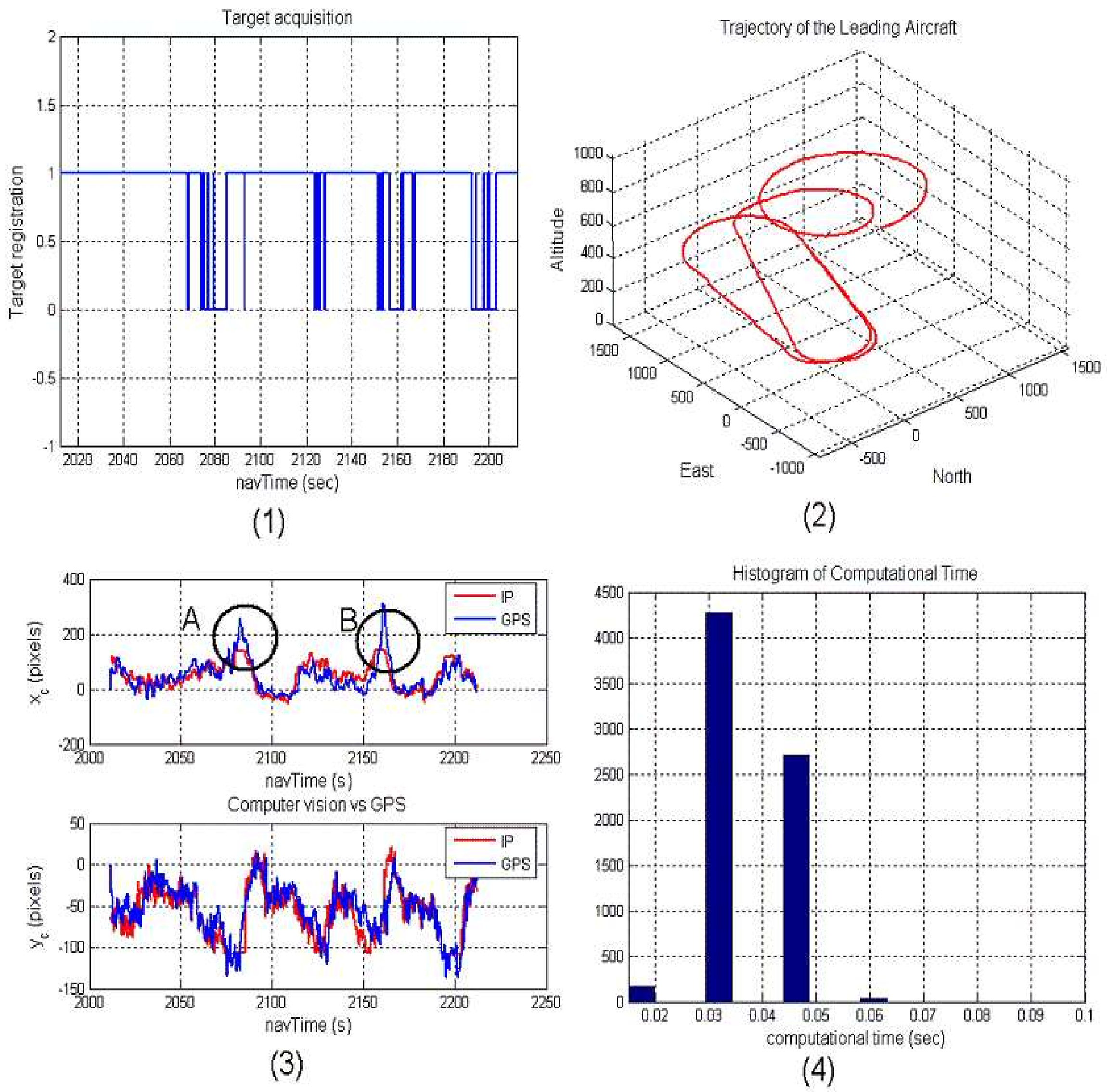


Figure 32: Simulation results on the aggressively maneuvering UAV video using the particle filter. (1) target acquisition, (2) the trajectory of the target in inertial frame, (3) comparison between the results from GPS/INS vs from computer vision in image plane, (4) the histogram for computational time spent in particle filtering and relative state estimation by EKF.

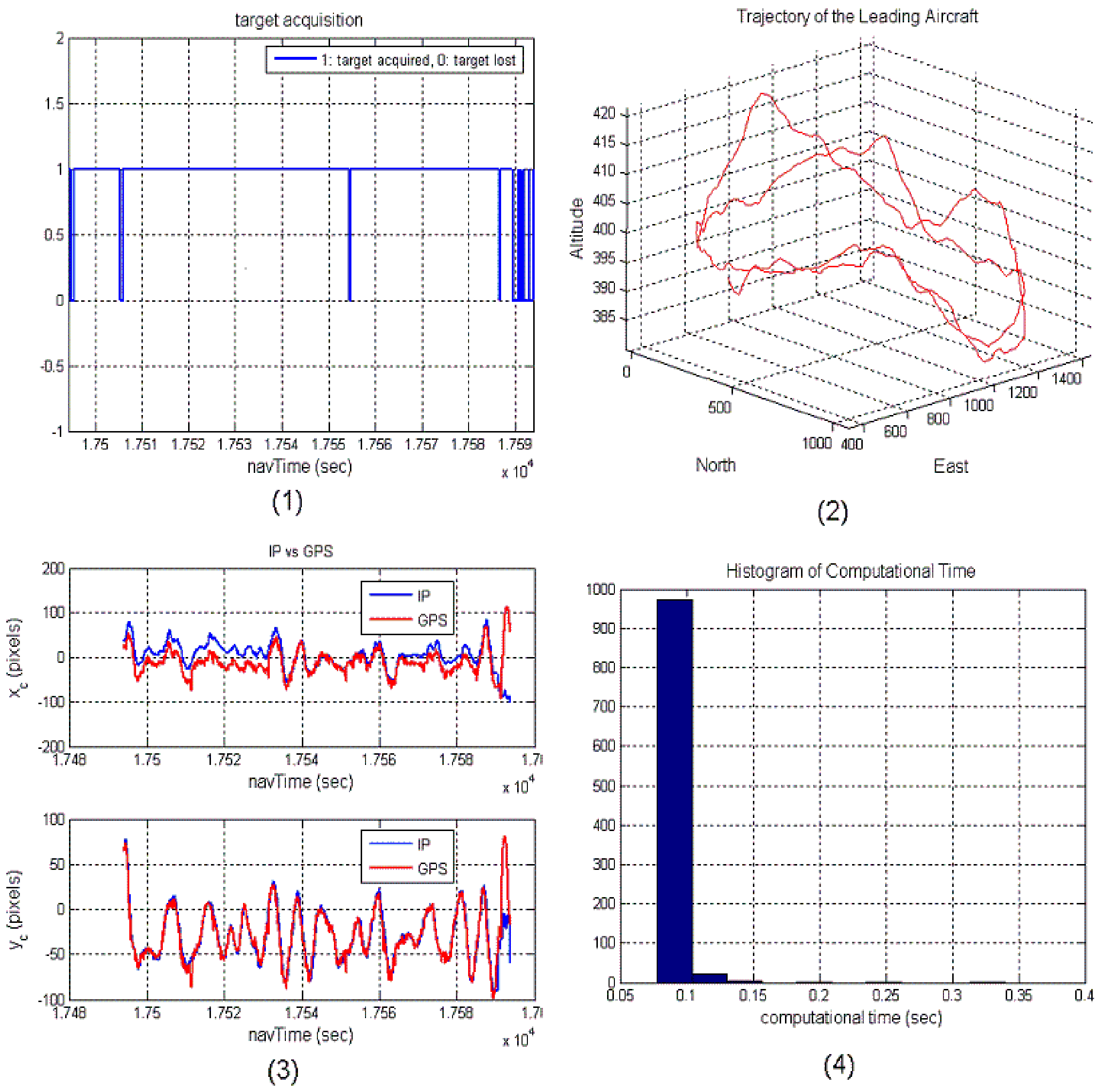


Figure 33: Flight test results using the particle filter on June 15, 2006. (1) target acquisition, (2) the trajectory of the target in inertial frame, (3) comparison between the results from GPS/INS vs from computer vision in image plane, (4) the histogram for computational time spent in particle filtering and relative state estimation by EKF.

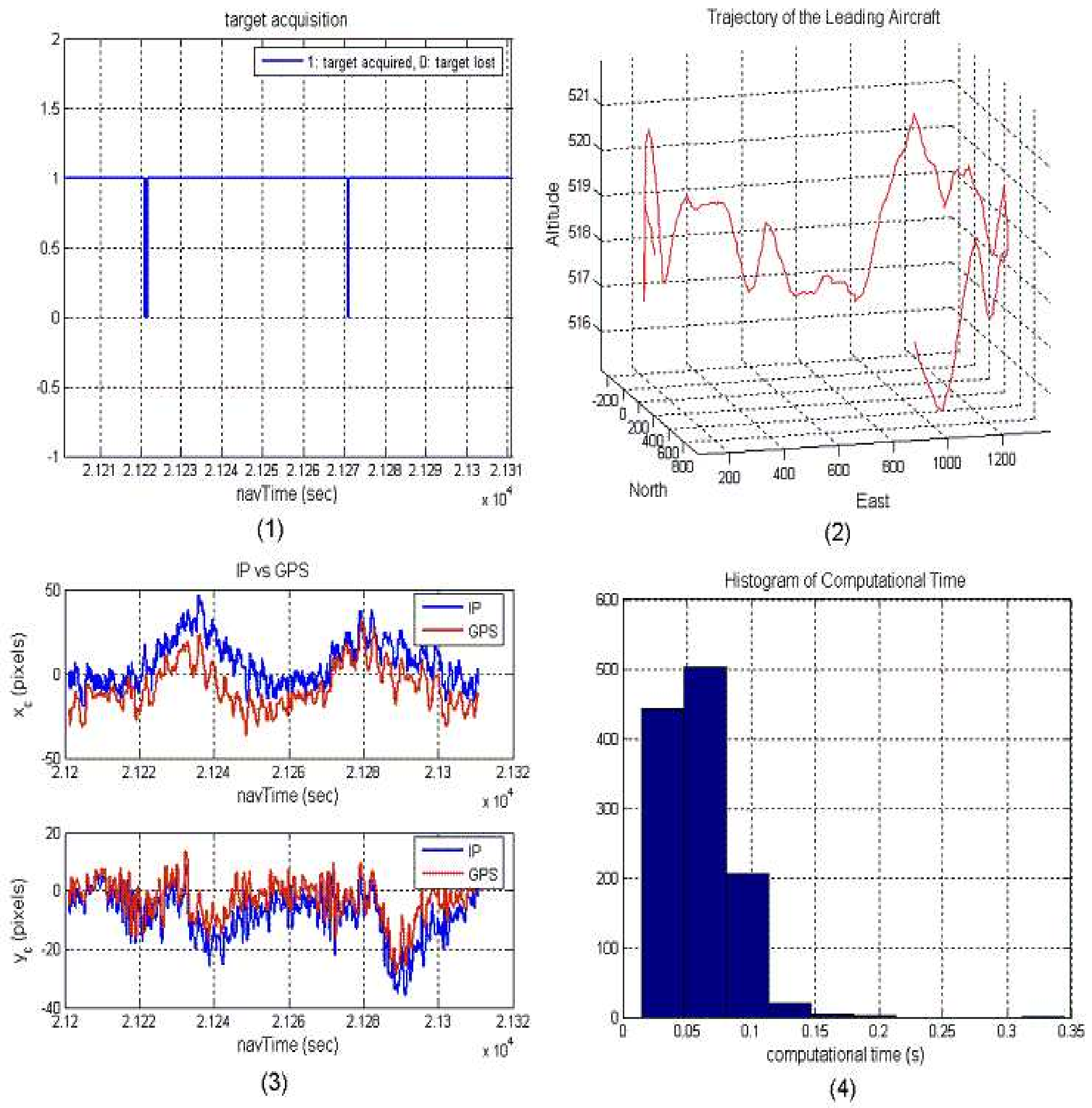


Figure 34: Flight test results using the particle filter on November 08, 2007. (1) target acquisition, (2) the trajectory of the target in inertial frame, (3) comparison between the results from GPS/INS vs from computer vision in image plane, (4) the histogram for computational time spent in particle filtering and relative state estimation by EKF.

CHAPTER VIII

CONCLUSIONS AND FUTURE RESEARCH

The purpose of this thesis was to present the development of real-time computer vision algorithms in order to track targets in uncertain complex environments purely based on a visual sensor. The main constraint of this research has been that the algorithms should work in real-time with limited computing power on the onboard computer in a small UAV. The goals of this research have been to develop fast and robust image processing and sound nonlinear filtering algorithms that are able to search, automatically detect, and track targets in adversarial conditions. This thesis has focused mainly on a contour tracker which tracks the outline of the target by using deformable contours in the image plane. This thesis was concerned with three specific subjects namely, image segmentation, shape modeling, and particle filtering.

We have designed the image segmentation algorithm based on the geometric active contour approach which is implemented via level set methods. A large part of the thesis has been devoted to investigating various methods to reduce the processing time of geometric active contours. At the beginning, the fast marching method was used for segmentation. The fast-marching-based vision system was effectively used in detecting and tracking targets in a wide region of interest. However, the system had some limitations in reducing the computational time of detection or tracking, and the algorithm was sensitive to noise. As an alternative, a fast implementation of the Chan-Vese model (FICVM) was proposed. The FICVM has combined a fast level set optimization method with a fast level set implementation. In addition, this thesis has added a regularizing term that ensures the smoothness of the evolving curve. This method has been robust to noise, since it is based on region-based geometric active contours. Furthermore, the processing time of this method has reached average computational times up to 100 frames/seconds in tracking scenes in typical to our applications.

This thesis has examined a fast method for estimating differences between the known shape and the shape measurements obtained from the segmentation process on a given image (shape dissimilarity measures). This thesis has proposed a shape modeling method based on the contour or zero level set of a simple discrete higher dimensional functional Ψ^s, and has used it in detecting multiple objects of interest quickly. Since the FICVM utilized Ψ^s, the dissimilarity measure in the FICVM is not as accurate as the one provided by using the traditional Ψ in [77]. Instead of reconstructing Ψ, which is computationally expensive, this thesis compensates for inaccuracies in the calculation of the dissimilarity measure by the shape modeling. Furthermore, because this method computes the dissimilarity measure quickly, it contributes to reducing the processing time of the particle filter while still providing a wealth of information about the shape of the target.

Finally, particle filtering has improved the tracking ability of the vision system in varied uncertain conditions. Particle filtering within a vision system has provided a probabilistic framework for a decision making procedure based on samples. The combination of blob tracking and contour contour tracking has greatly improved the processing time for real-time applications. The FICVM and the contour based shape modeling method have been useful for reducing the processing time of the contour tracking part of the particle filter. Furthermore, the the geometric active contour framework of the particle filter has allowed for estimation of local deformation as well as global motion of the target's shape. The new particle filter has been applied to low contrast and severe sunlight conditions, and to cluttered environment cases, to the appearing/disappearing target tracking problem using by the multiple model tracking method. In these applications, particle filtering demonstrated promising results in both simulations using streaming video and in actual flight tests. We have also demonstrated the utility of the filter for multiple target tracking in the presence of occlusions. The proposed algorithms in this thesis will open the door to various future research opportunities related to UAVs (e.g., various low-cost missions by using UAV and vision, see and avoid missions, and formation flight between multiple multi-purpose UAVs).

REFERENCES

[1] B. K. P. Horn, Robot Vision, MIT press, 1986.

[2] E. R. Davies, Machine Vision: Theory, Algorithms, Practicalities, 2nd Edition, Academic Press, 1996.

[3] G. W. Awcock and R. Thomas, Applide Image Processing, McGraw-Hill, Inc., 1996.

[4] J. Hollingum, Machine Vision: The Eyes of Automation, IFS Ltd., 1984.

[5] R. Bajcsy, "Active Perception," Proceedings of IEEE, pp. 996-1005, 1988.

[6] J. Aloimonos, I. Weiss, and A. Bandpodahay, "Active Vision," International journal on Computer Vision, vol. 7, pp. 333-356, 1988.

[7] A. Blake and A. Yulle, Active Vision, MIT Press, Cambridge, Mass., 1992.

[8] H. Oh, C. Lee, and I. Mitsuru, "Navigation Control of a Mobile Robot Based on Active Vision," Proceedings of Industrial Electronics, Control and Instrumentation (IECON), pp. 1122-1126, 1991.

[9] J. Peng, A. Srikaew, M. Wilkes, K. Kawamura, and A. Peters, "An Active Vision System for Mobile Robots," IEEE Conference on Systems, Man, and Cybernetics, pp. 1472-1477, 2000.

[10] T. Rabie, G. Auda, A. El-Rabbany, A. Shalaby and B. Abdulhai, "Active-Vision-Based Traffic Surveillance and Control," Proceedings of the International Conference on Vision Interface, Ottawa, Canada, 2002.

[11] H. Araujo, J. Batista, P. Peixoto, and J. Dias, "Pursuit Control In a Binocular Active Vision System Using Optical Flow," IEEE International Conference on Pattern Recoginition, pp. 1465-1469, 1997.

[12] S. Rougeaux and Y. Kuniyoshi, "Robust Real-Time Tracking on an Active Vision Head," IEEE Intelligent Robots and Systems, pp. 873-879, 1997.

[13] X. Yuan, C. Qiu, R. Chen, Z. Hu, and P. Liu, "Vision System Research for Autonomous Underwater Vehicle," IEEE International Conference on Intelligent Processing Systems, pp. 1465-1469, 1997.

[14] J. Miura, T. Kanda, and Y. Shirai, "An Active Vision System for Real-Time Traffic Sign Recognition," IEEE Intelligent Transportation Systems, pp. 52-57, 2000.

[15] A. Bakhtari and B. Benhabib, "Active Vision System for Multi-Target Surveillance," IEEE International Conference on Mechatraonics and Automation, pp. 1169-1174, 2005.

[16] " UAS Roadmap 2005," http://www.acq.osd.mil/usd/.

[17] S. Bruno, M. Micheli, G. Donato, and T. J. Koo, "Vision Based Navigation for an Unmanned Air Vehicle," IEEE International Conference on Robotics and Automation, pp. 1757-1765, 2001.

[18] S. Srikanth, J. F. Montgomery, and G. S. Sukhatme, "Vision-Based Autonomous landing of an Unmanned Aerial Vehicle," IEEE International Conference on Robotics and Automation, pp. 2799-2804, 2002.

[19] C. D. Wagter, A. A. Proctor, and E. N. Jonson, "Vision-only Aircraft Flight Control," AIAA Digital Avionics Conference, No. 8B2, Indianapolis, IN, October, 2003.

[20] A. A. Proctor and E. N. Jonson, “Vision-only Aircraft Flight Control Methods and Test Results,” Proceedings of the AIAA Guidance, Navigation and Control Conference, 2004.

[21] M. Niethammer, P. Vela, and A. R. Tannenabaum, “Geodesic Observers for Dynamically Evolving Curves,” International Conference on Decision and Control, 2005.

[22] E. N. Johnson, A.A. Proctor, J. Ha, and A. R. Tannenbaum, “Visual Search Automation for Unmanned Aerial Vehicles,” IEEE TRansactions on Aerospace and Electronic Systems, vol. 31, No. 1, pp. 219-232, 2005.

[23] E. N. Johnson, A. J. Calise, Y. Watanabe, J. Ha, and J. C. Neidhoefer, “Real-Time Vision-Based Relative Navigation,” Proceedings of the AIAA Guidance, Navigation, and Control Conference, August 2006.

[24] A. Blake and M. Isard, Active Vision, eds, Springer, 1998.

[25] Y. Bar-Shalom and T. E. Fortmann, Tracking and Data Association, vol. 179, Academic Press, 1988.

[26] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active Contour Models,” International Journal of Computer Vision, 1(4), 321-331, 1987.

[27] V. Caselles, F. Catte, T. Coll, and F. Dibox, “A Geometric Model for Active Contours in Image Processing,” Numerische Mathematik, 66, pp. 1-31, 1993.

[28] R. Malladi, J. A. Sethian, and V. C. Vemuri, “Shape Modelling with Front Propagation: A Level Set Approach,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995.

[29] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, “Gradient Flows and Geometric Active Contour Models,” International Conference on Computer Vision (ICCV), 1995.

[30] V. Caselles, R. Kimmel, and J. Sapiro, “Geodesic Active Contours,” International Journal of Computer Vision, 22, 61-79, 1997.

[31] R. Malladi and J. A. Sethian, “An O(N log N) Algorithm for Shape Modeling,” Proceedings of National Academy of Sciences, vol. 93, pp. 9389-9392, 1996.

[32] R. Malladi and J. A. Sethian, “Level Set and Fast Marching Methods In Image Processing and Computer Vision,” Proceedings of International Conference on Image Processing, 1996.

[33] J. A. Sethian, “A Fast Marching Level Set Method for Monotonically Advancing Fronts,” In Proceedings of National Academy of Science of the USA 93. 93(4):1591-1595, 1996.

[34] D. Adalsteinsson and J. A. Sethian, “A Fast Marching Level Set Method for Propogating Interfaces,” Journal of Computational Physics, 118(2):269-277, 1995.

[35] J. A. Sethian, Level Set Methods and Fast Marching Methods, Cambridge Press, 1996.

[36] T. F. Chen and L. A. Vese, “Active Contours Without Edges,” IEEE Transaction on Image Processing, vol. 10, No 2, 2001.

[37] B. Song and T. Chan, “Fast Algorithm for Level Set Based Optimization,” UCLA CAM Report, 02(68), 2002.

[38] A. Tannenbaum, “Three Snippets of Curve Evolutiontheory in Computer Vision,” Mathematical and Computer Modelling Journal, vol. 24, pp. 103-119. 1996.

[39] N. Paragios and R. Deriche, "A PDE-Based Level-Set Approach for Detection and Tracking of Moving Objects," International Conference on Computer Vision (ICCV), pp. 1139-1145, 1998.

[40] N. Paragios and R. Deriche, "Geodesic Active Regions for Motion and Tracking," International Conference on Computer Vision (ICCV), pp. 688-694, 1999.

[41] S. Zhu, T. S. Lee, and A. L. Yuille, "Region Competition: Unifying Snake/Balloon, Region Growing, and Bayes/mdls/energy for Multi-band Image Segmentation," IEEE Transactions on Pattern Analysis Machine Intelligence, vol. 18, no. 9, pp. 884-900, September, 1996.

[42] A. Yezzi, A. Tsai, and A. Willsky, "A Statistical Approach to Image Segmentation for Bimodeal and Trimodal Imagery," Proceedings of International Conference on Computer Vision (ICCV), September, 1999.

[43] A. Yezzi, A. Tsai, and A. Willsky, "A Fully Global Approach to Image Segmentation via Coupled Curve Evolution Equations," Journal of Visual Communication and Image Representation, 13, pp. 195-216, 2002.

[44] Y. Shi and W. C. Karl, "Real-time Tracking Using Level Sets," In Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 34-41, 2005.

[45] N. Peterfreund, "Robust Tracking of Position and Velocity With Kalman Snakes," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, pp. 564-569, 1999.

[46] A. Popoulis, Probability and Statistics, Prentice Hall, Inc., New Jersey, 1990.

[47] R. Y. Rubinstein, Simulation and the Monte Carlo Method, J. Wiley, New York, 1981.

[48] B. Efron, “The Bootstrap, Jackknife and Other Resampling Plans,” Philadelphia: Society of Industrial and Applied Mathematics, 1982.

[49] D. B. Rubin, “Using the SIR Algorithm to Similulate Posterior Distributions,” In Bayesian Statistics 3, Oxford University Press, pp. 395-402, 1982.

[50] R. G. Brown and P. Y. C. Hwang, Introduction To Random Signals and Applied Kalman Filtering, 3rd Edition, John Wiley and Sons, 1997.

[51] M. S. Grewal and A. P. Andres, Kalman Filtering, 2nd Edition, John Wiley and Sons, 2001.

[52] E. L. Lehmann and G. Casella, Theory of Point Estimation, 2nd Edition, Springer, 1998.

[53] A. E. Bryson and Y. Ho, Applide Optimal Control, Taylor and Francis, 1975.

[54] P. Muller, “Monte Carlo Integration in General Dynamic Models,” Contemporary Mathematics, 115, pp. 145-163, 1991.

[55] A. F. M. Smith and A. E. Gelfand, “Bayesian Statistics Without Tears: A Sampling-Resampling Prospective,” The American Statistician, vol. 46, No. 2, pp. 84-88, 1992.

[56] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, “Novel Approach to Nonlinear/Nongaussian Bayesian State Estimation,” IEEE Proceedings of Radar, Signal Processing, 140, pp. 107-113, 1993.

[57] D. Avitzour, “A Stochastic Simulation Bayesian Approach to Multitarget Tracking,” Cambridge University Press, 1997.

[58] M. Israd and A. Blake, “Contour Tracking by Stochastic Propagation of Conditional Density,” In European Conference on Computer Vision (ECCV), pp. 343-356, 1996.

[59] N. Gordon, “A Hybrid Bootstrap Filter for Target Tracking in Clutter,” IEEE Transactions on Aerospace and Electronic Systems, 33, pp. 353-358, 1997.

[60] A.C. Davison and D. V. Hinkley, Bootstrap Methods and their Applications, Cambridge University Press, 1997.

[61] A. Doucet, S. Godsill, and C. Andrieu, “On Sequential Monte Carlo Sampling Methods for Bayesian Filtering,” Statistics and Computing, vol. 10, no. 3, pp. 197-208, 2000.

[62] D. J. Salmond and H. Birch, “A Particle Filter for Track-Before-Detect,” In Proceedings of American Control Conference (ACC), pp. 3755-3760, 2001.

[63] M. Isard and A. Blake, “CONDENSATION - Conditional Density Propagation for Visual Tracking” International Journal of Computer Vision, 29(1), pp.5-28, 1998.

[64] M. Israd and A. Blake, “ICONDENSATION: Unified Low-Level and High-Level Tracking in A Stochastic Framework,” In European Conference on Computer Vision (ECCV), pp. 893-908, 1998.

[65] H. Tao, H. Sawhney, and R. Kumar, “A Sampling Algorithm for Tracking Multiple Objects,” International Conference on Computer Vision (ICCV), 1999.

[66] J. MacCormick and A. Blake, “A Probabilistic Exclusion Principle for Tracking Multiple Objects,” International Conference on Computer Vision (ICCV), pp. 572-578, 1999.

[67] B. K. Esther and A. Frank, “Tracking Multiple Objects Using the Condensation Algorithm,” Robotics and Autonomous Systems, 34, pp. 93-105, 2001.

[68] A. Doucet, N. de Freitas, and N. Gordon, eds., Sequential Monte Carlo Methods in Practice, New York: Springer, 2001.

[69] D. Tweed and A. Calway, “Tracking Many Objects Using Subordinated Condensation,” In The British Machine Vision Conference, pp. 283-292, 2002.

[70] B. Ristic, S. Arulampalam, and N. Gordon, Beyond the Kalman Filter: Particle Filters for Tracking Applications, Artech House Publisher, 2004.

[71] F. Gibou and R. Fedkiw, ”A Fast Hybrid k-Means Level Set Algorithm for Segmentation,” 4th Annual Hawaii International Conference on Statistics and Mathematics, pp. 281-291, 2002.

[72] Y. Rathi, N. Vaswani, A. Tannenbaum and A. Yezzi, “Particle Filtering for Geometric Active Contours with Application to Tracking Moving and Deforming Objects,” In Computer Vision and Pattern Recognition (CVPR), vol. 2, pp. 2-9, 2005.

[73] Y. Pan, J. D. Birdwell, and S. M. Djouadi, “Efficient Implementation of the Chan-Vese Models Without Solving PDEs,” Proceedings of the CDC, pp. 350-354, 2006.

[74] J. Ha, E. N. Johnson, and A. R. Tannenabaum, “Real-time Visual Tracking Using Geometric Active Contours for the Navigation and Control of UAVs,” In Proceedings of American Control Conference (ACC), 2007.

[75] M. E. Leventon, W. E. L. Grimson, and O. Faugera, “Statistical Shape Influence in Geodesic Active Contours,” In Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 316-323, 2000.

[76] M. Rousson and N. Paragios, “Shape Priors for Level Set Representation,” In European Conference on Computer Vision (ECCV), pp. 78-92, 2002.

[77] D. Cremers and S. Soatto, “A Pseudo-Distance for Shape Priors in Level Set Segmentation,” In IEEE Workshop on Variational Geometric and Level Set Methods in computer Vision. IEEE, 2003.

[78] J. Ha , E. N. Johnson, and A. R. Tannenabaum, “Real-time Visual Tracking Using Geometric Active Contours and Particle Filters,” Submitted to AIAA Journal of Aerospace Computing, Information, and Communication, 2007.

[79] A. Okabe, B. Boots, and K. Steiglitz, Spatial Tessellations; Concepts and Applications of Voronoi Diagrams, Wiley, Chichester, UK, 1992.

[80] J. Berg, A. Tannenbaum, and A. Yezzi. “Curve evolution models for real-time identification with application to plasma etching,” IEEE Transactions on Automatic Control, vol. 44, pp. 99-104, 1999.

[81] Y. Rathi, N. Vaswani, A. Tannenbaum and A. Yezzi, “Tracking Deforming Objects Using Particle Filtering for Geometric Active Contours,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, pp. 1470-1475, 2007.

[82] M. Isard and A. Blake, “A Mixed-State Condensation Tracker with Automatic Model-Switching,” IEEE International Conference on Computer Vision, pp. 107–112, 1998.

[83] J. Vermaak, A. Doucet, and P. Perez, “Maintaining Multi-Modality through Mixture Tracking,” International Conference on Computer Vision, 2003.

[84] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe, “A Boosted Particle Filter: Multitarget Detection and Tracking,” citeseer.ist.psu.edu/okuma04boosted.html, 2004.

[85] D. S. Angelova, Tz. A. Semerdjiev, V. P. Jilkov, and E. A. Semerdjiev, "Application of a Monte Carlo Method for Tracking Maneuvering Target in Clutter," Mathematics and Computers in Simulation, vol 55, pp. 15-23, 2001.

[86] A. Milsteain, J. N. Snchez, and E. T. Williamson, "Robust Global Localization Using Clustered Particle Filtering," In Proceedings of AIAA/IAAI, pp. 581-586, 2002.

[87] J. Driessen, "Object Tracking in a Computer Vision based Autonomous See-and-Avoid System for Unmanned Aerial Vehicles," Master's Thesis in Computer Science at School of Vehicle Engineering, Royal Institute of Technolody, 2004.

[88] B. Kimia, A. Tannenbaum, and S. Zucker, "Shapes, Shocks, and Deformations i: the Components of Two-Diemensional Shape and the Reaction-Diffusion Space," In International Journal of Computer Vision, vol. 15, pp. 189-224, 1995.

[89] S. Soatto and A. Yezzi, "Deformation: deforming motion, shape average and the joint segmentation and registration of images," International Journal of Computer Vision, vol. 53, pp. 153-167, 2003.

[90] J. Jackson, A. Yezzi and S. Soatto, "Tracking Deformable Moving Objects under Severe Occlusions," IEEE Conference on Decision and Control, vol. 3, pp. 2990-2995, 2004.

[91] T. F. Cootes, C. J. Taylor, D. H. Cooper and J. Graham, "Active Shape Models-Their Training and Applications," Computer Vision and Image Understanding, vol. 61, pp. 38-59, 1995.

[92] T. F. Cootes, C. Beeston, G. J. Edwards and C. J. Taylor, "A Unified Framework for Atlas Matching Using Active Appearance Models," In Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer-Verlag, pp. 322-333, 1999.

[93] Y. Wang and L. H. Staib, "Elastic Model Based Non-Grid Registration Incorporating Statistical Shape Information," In Medical Image Computing and Computer-Assisted Intervention, pp. 1162-1173, 1998.

[94] L. H. Staib and J. S. Duncan, "Boundary Finding with Parametrically Deformable Models," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, pp. 1061-1075, 1992.

[95] P. Stec and M. Domanski, "Efficient Unassisted Video Segmentation Using Enhanced Fast Marching," In Proceedings of International Conference on Image Processing, vol. 2, pp. 427-430, 2003.

[96] E. Pichon, A. Tannenbaum, and R. Kikinis, "A Statistically Based Surface Evolution Method for Medical Image Segmentation: Presentation and Validation," In Medical Image Computing and Computer-Assisted Intervention (MICCAI), vol. 2, pp. 711-720, 2003.

[97] A. Tsai, A. Yezzi, W. Wells, C. Tempany, C. Tucker, A. Fan, W. E. Grimson, and A. Willsky, "Model-Based Curve Evolution Technique for Image Segmentation," In Proceedings of Computer Vision and Pattern Recognition, vol. 1, pp. 463-468, 2001.

[98] D. Cremers, N. Sochen, and C. Schnorr, "Towards Recognition-Based Variational Segmentation Using Shape Prior and Dynamic Labeling," In L. Grifith, editor, International Conference on Scale Space Theories in Computer Vision, vol. 2965 of LNCS, pp. 388-400, 2004. Springer.

[99] T. Chan and W. Zhu, "Level Set Based Shape Prior Segmentation," Technical Report, Computational Applied Mathematics, UCLA, 2003.

[100] M. Moelich and T. Chan, "Tracking Objects with The Chan-Vese Algorithm," Technical Report 03-14, Computational Applied Mathematics, UCLA, 2003.

[101] C. J. Needham and R. D. Boyle, "Tracking Multiple Sports Players through Occlusion," In 12th British Machine vision Conference, BMVC01, pp. 93-102, Manchester, UK, 2001.

[102] M. Isard and J. MacCormick, "BraMBLe: A Bayesian Multiple-Blob Tracker," In International Journal of Computer Vision, vol. 15, pp. 34-41, 2001.

[103] S Arulampalam, S.Maskell, N. J. Gordon and T. Clapp, "A Tutorial on Particle Filters for On-Line Non-Linear/Non-Gaussian Bayesian Tracking," IEEE Transactions on Signal Processing, vol. 50, pp. 174-188, 2002.

[104] W. Du and J. Piater, "Tracking by cluster analysis of feature points using a mixture particle filter," IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 165-170, 2005.