Automatic Detection and Tracking in Underwater Environments with Marine Snow

by

Catherine A. Gamroth

B. Engineering, University of Victoria, 2003

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES
(Computer Science)

The University Of British Columbia
(Vancouver)

December 2010

Abstract

This project addresses the issue of automatic detection and tracking of man-made objects in subsea environments with poor visibility and marine snow. Underwater research and engineering is a quickly growing field and there are few computer vision techniques that specifically address these challenges.

The proposed system involves minimizing noise and video artifacts, estimating camera motion, detecting line segments and tracking targets. Overall, the system performs well under the conditions in the test video and the equal error rate is approximately 16%. Tests show how parameters may be tuned to account for changes in environmental conditions and to trade off the number of false negatives and false positives. System performance is affected by many factors. Poorest performance occurs under conditions of heavy marine show, low-contrast targets, and fast camera motion. Performance also suffers if the background conditions in the image change.

This research makes two contributions. First, we provide a survey of techniques that address similar problems and evaluate their suitability for this application, Second, we integrate existing techniques into a larger system. Techniques include median filtering, Canny edge detection, Hough transforms, Lucas-Kanade first-order optical flow and particle filtering. Where gaps exist between system components, new methods are developed. Testing evaluates the effects of system parameters and the conditions under which the system is effective.

Table of Contents

List of Tables

List of Figures

Glossary

1D one dimensional

AUV Autonomous Underwater Vehicle

BP British Petroleum

CLAHE Contrast Limited Adaptive Histogram Equalization

CPU Central Processing Unit

GUI Graphical User Interface

HMM Hidden Markov Model

NEPTUNE North-East Pacific Time-series Undersea Networked Experiments

ROV Remotely Operated Vehicle

SMC Sequential Monte Carlo

VENUS Victoria Experimental Network Under the Sea

Acknowledgments

I would like to thank...

My supervisors, Dr. Little and Dr. Woodham, for their guidance and support during my graduate studies.

Dr. Nando de Freitas for his expertise in SMC methods.

Members of the Laboratory for Computational Intelligence (LCI) for their insight, advice, and bad puns.

My husband, Emmett, for his wisdom and unfailing support.

The Natural Sciences and Engineering Research Council of Canada (NSERC) for their financial support.

Dedication

To Emmett.

Chapter 1

Introduction

1.1 Motivation

The recent increase in underwater research and engineering has led to a greater demand for scientific instruments that can operate under the harsh and challenging subsea conditions. The British Petroleum (BP) oil spill remediation, underwater logging, the Victoria Experimental Network Under the Sea (VENUS), and the North-East Pacific Time-series Undersea Networked Experiments (NEPTUNE) are a few prominent examples of recent projects in the underwater environment, all of which would benefit from improved computer vision systems.

The BP oil spill in the Gulf of Mexico in April, 2010, brought international attention to the difficulties of working underwater and the need for better understanding and protection of subsea environments. Much of the work to stop the spilling oil was performed by operators controlling Remotely Operated Vehicles (ROVS) with specialized arms and manipulators. The operators relied heavily on video feedback to control the vehicles.

Triton Logging is a BC-based company that specializes in logging underwater forests that were created when hydro dams were built [Triton, 2010]. Triton estimates that there are 300 million trees world-wide in such reservoirs and claims these trees can be harvested in a more economical and safe manner than traditional logging. At the core of their operation is an ROV equipped with large graspers and saw blades. An operator relies on eight video cameras and sonar data to lo-

Figure 1.1: Seaeye Falcon. This inspection class ROV has a forward-mounted camera and two lights.

cate trees, plan and execute cuts, and attach flotation bags to return the trees to the surface.

The NEPTUNE and VENUS projects are large-scale ocean observatory networks that will allow scientists to install monitoring equipment on the seafloor to provide long-term, continuous data collection. Planned experiments include the monitoring of hydrothermal systems, installation of a seismograph network, geophysical imaging of gas hydrates, and investigation of the role of disturbance in benthic ecosystems. These networks will provide great opportunities for many researchers and will further drive demand for underwater scientific equipment, including video cameras.

All of these projects involve work in areas too deep for human divers. ROVS and Autonomous Underwater Vehicles (AUVS) are becoming increasingly common as a means to explore these otherwise unreachable areas. As the name suggests, ROVS are controlled by human operators who often rely on visual feedback from cameras mounted on the vehicle. Figure 1.1 is an example of a small ROV used mainly to inspect rather than to manipulate underwater objects. At the top of the image is the cable or tether that provides the vehicle with power and communication. Some ROVS are capable of simple automatic tasks like basic station-keeping or docking,

but these capabilities remain open areas of research. AUVS operate without human intervention and typically carry out pre-programmed tasks such as swimming a grid pattern at a fixed depth. There are currently few, if any, commercially available vehicles that can reliably make complex real-time decisions, largely because of the poor quality of available data about the operating environment.

There are many constraints when working underwater, particularly at significant depths. These include high pressures, the absence of ambient light, extreme temperatures, and high attenuation of light and communication signals. Underwater environments tend to be low-contrast with few distinctive features. In many cases, poor visibility is due to high levels of turbidity and "marine snow" from particulate and planktonic organisms. For these reasons, many computer vision techniques that were developed for surface-based applications cannot easily be adapted for underwater use. Acoustic techniques such as sonar can provide additional data, but these systems are more costly and complex than video and usually give lower resolution. The availability of improved underwater vision systems would help to drive innovation in these challenging and poorly understood environments.

1.2 Problem Description

This project addresses the issue of automatic detection and tracking of man-made objects in underwater environments with poor visibility. This is demonstrated using video captured by an ROV in Saanich Inlet, BC.

The ROV is an engineering research vehicle similar to that shown in Figure 1.1 that is used to locate, inspect and maintain man-made objects resting on the seafloor. The objects in question include power and data cables, platforms and scientific instruments. Although the precise shapes of the objects are not known a priori, they can be distinguished from their surroundings by their straight edges, relative rigidity and lack of motion.

The ability to automatically track edges would help the ROV operator work more easily and with less eye strain. Future extensions of such a system could allow the ROV to perform automated tasks such as station keeping by maintaining an object centered in the camera, or swimming along cables laid on the seafloor.

1.3 System Description

The system consists of an estimate of optical flow, an edge detector, and multiple particle filters to track a varying set of long, continuous straight lines that persist over time. A prototype has been implemented as proof of concept; the system is not currently optimized for computational speed but the design choices allow for the possibility of a real-time implementation.

This section describes the major challenges addressed by this system: the system must be robust to marine snow and poor visibility, account for camera motion, extract low-intensity lines, and track edges.

1.3.1 Marine Snow

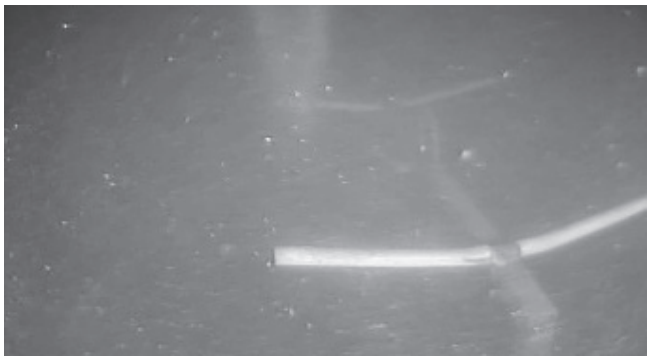
The term marine snow refers to underwater debris including sediment and planktonic organisms (e.g., algae and krill). Marine snow creates high-contrast visual clutter that interferes with the detection of lower-contrast objects of interest. Since all lighting is provided by the vehicle itself, there is significant light reflection off the particles, causing them to appear brighter than their surroundings.

As shown in Figure 1.2, there is variation in the shape, size, and severity of marine snow. The motion of the snow is unpredictable and cannot be reliably used to predict the camera motion. For example, in some cases, individual "snowflakes" are freely swimming organisms moving independently, in others they are relatively stationary in the environment and move only relative to the camera, and in others they are caught in turbulent water from the vehicle's thrusters.

The system minimizes the effects of snow by pre-processing each video frame and by relying on techniques that are robust to noise. The system trades off camera speed and severity of marine snow: in low-snow conditions, the camera may move relatively quickly, but in heavy-snow conditions, the camera must move more slowly in order to obtain good results.

1.3.2 Low Visibility

Light attenuates quickly under water. As a result, only bright objects very near the camera can create large intensity gradients which quickly fade as the camera moves away from the object, making it difficult to distinguish the object from the

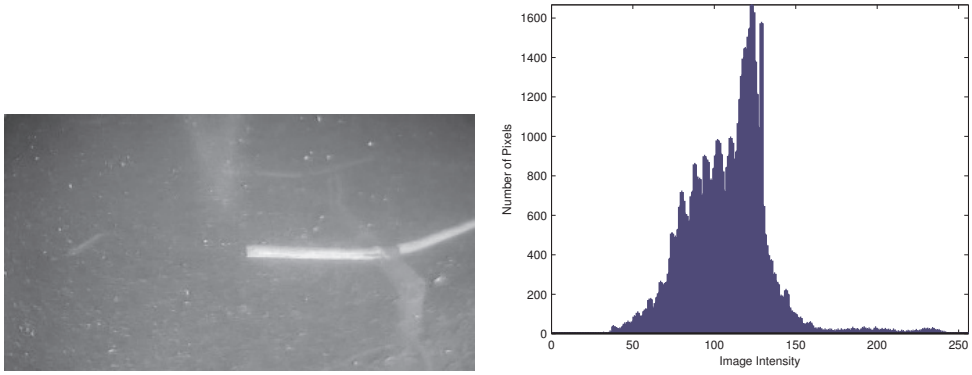


(a)

(b)

(c)

Figure 1.2: Object of interest under different levels of marine snow. (a) Moderate marine snow. (b) The faint lines in the centre of the image are the crossbars of the man-made object. The three other faint lines are fish. The bright line at the right of the image is the tether of the ROV. (c) Heavy marine snow with part of the vehicle visible along the bottom edge of the image.



(a) Raw Image **(b)** Histogram of pixel intensities

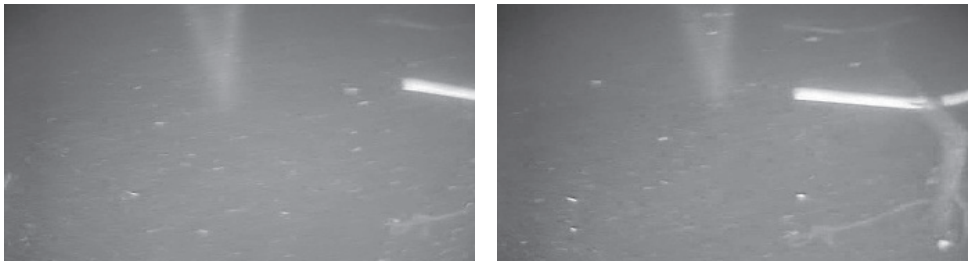
Figure 1.3: Low Visibility (a) Rear bars appear faint due to high light attenuation. (b) Histogram of the low contrast image shows that pixel intensities fall mostly within one-fifth of the available dynamic range.

background. For example, the object in Figure 1.3 has bright, horizontal bars at each end, separated by a distance of approximately 1.5 m. The bars close to the camera are easily visible while those at the other end are difficult to see. The histogram of intensities in Figure 1.3 shows the narrow range of pixel intensities present in a sample image.

1.3.3 Motion Estimation

The ROV and camera move in a dynamic and unknown environment. In order to accumulate evidence over time for the locations of edges, it is critical to describe the optical flow between subsequent frames. Due to the marine snow and lack of texture or reliable image features, flow estimation by feature tracking isn't feasible. As Figure 1.4 demonstrates, the ROV is capable of maneuvering quickly. The objects of interest are rigid, the motion of the marine snow is not necessarily related to the camera motion, and lighting or lens artifacts remain stationary relative to the camera.

The system estimates optical flow by assuming affine motion over short timespans. It treats visible portions of the vehicle and lighting and lens artifacts as background and disregards them for flow calculations. If background conditions change



(a) Frame 65 **(b)** Frame 75

(c) Frame 85 **(d)** Frame 95

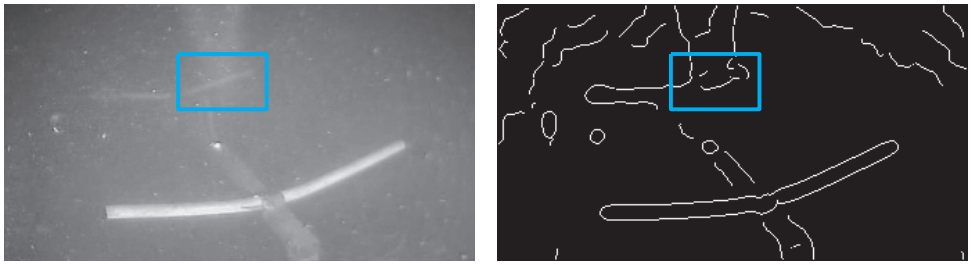
Figure 1.4: Camera motion in one second of video

significantly during an ROV mission, the system will falsely identify background artifacts and target edges unless the ROV operator is able to reset the background by recording two to three seconds of video with no target objects in view.

1.3.4 Edge Detection

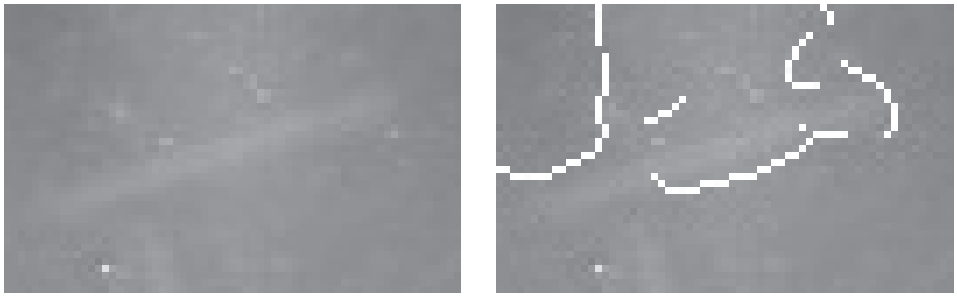
Faint edges are difficult to distinguish from the background using only image intensity in an individual frame. Often, the snow is of higher contrast than the edges of interest. Figure 1.5 illustrates the difficulties of edge detection under such noisy conditions. Better results can be obtained for a particular image by fine-tuning the edge detector parameters, but a more robust solution is required for video with changing conditions.

After minimizing marine snow and lighting artifacts, the system runs edge detection and line extraction and further post-processes those results. Despite this, the system is not able to extract all lines in all image frames. For this reason, the system is not solely an edge detector, but also relies on object tracking.



(a) Raw image

(b) Edges



(c) Close-up of raw image

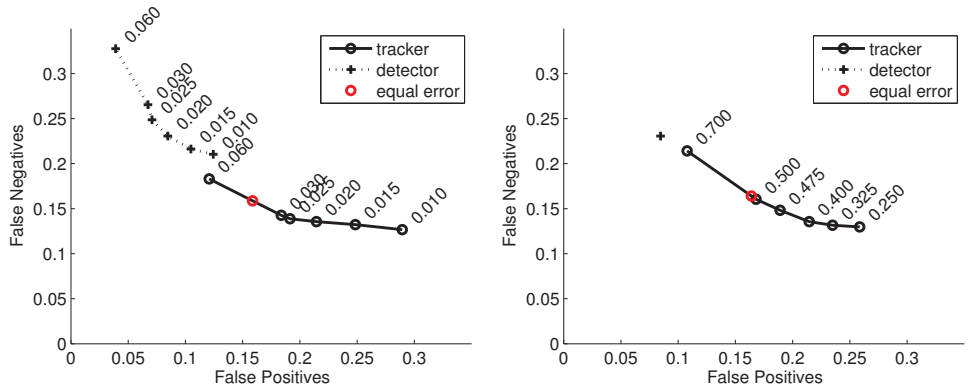
(d) Close-Up of Edges superimposed on raw image

Figure 1.5: Edge Detection (a) Noisy, low contrast image with region of interest in blue box. (b) Close-up of the region of interest. The small intensity gradients make it difficult to detect and localize the rear bars. (c) Image after edge detection. (d) The edges of the rear bar are not properly detected or localized.

1.3.5 Target Identification and Tracking

The edges are rigid and non-deformable but their appearance varies as the camera moves. The apparent pose of the object can change, affecting the location, orientation and length of the edge. Further, due to the poor visibility, edges quickly become faint as they move farther from the camera, causing changes in intensity gradient. This is demonstrated in Figure 1.2, where the same object is shown in three different configurations.

The number and pose of target edges is unknown and varying. This results in overlap and ambiguities. In order to track edges over multiple time steps, it is necessary to match lines between consecutive frames, identify new edges, and



(a) Performance curve for different values of γ_{detect}

(b) Performance curve for different values of γ_{track}

Figure 1.6: Performance curves generated by varying the thresholds for detection and tracking. For this dataset, the equal error rate is approximately 16% for both cases.

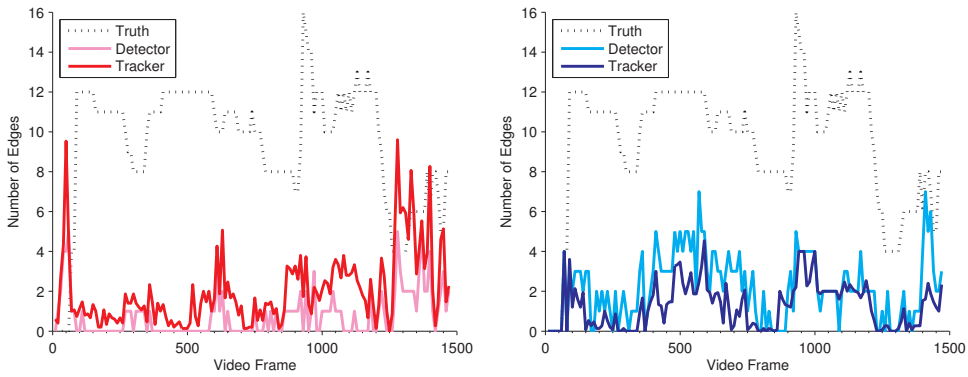
detect incorrect edges. To do this, we establish a method to compare two line segments quantitatively use a Sequential Monte Carlo (SMC) method as a probabilistic framework to bring together evidence from motion estimation and edge detection.

1.4 Testing and Results

The system performance was evaluated by comparing the edges identified by the system to ground truth. The test video is 50 seconds long and includes many different test conditions. There is variation in the level of marine snow, the velocity of the camera, the distance between the camera and the targets, and the lighting conditions.

Overall, the system performs well under the conditions in the test video. Figure 1.6 shows the performance curves that result from varying the thresholds for detection and tracking, γ_{detect} and γ_{track}, respectively. For this dataset the equal error rate is approximately 16% and the performance can be shifted along the curves to reach a desired balance between false negatives and false positives. Lower false negative rates are achieved by decreasing one or both of the thresholds.

Figure 1.7 shows how the results change for different conditions in the dataset.



(a) False positives **(b)** False negatives

Figure 1.7: Performance over time for $\gamma_{detect} = 0.02$ $\gamma_{track} = 0.4$. (a) The false positive rate varies throughout the test sequence and has spikes at the beginning and end where there is heavy marine snow and quick camera motion. The tracker has more false positives than the detector. (b) The false negative rate depends on the conditions in the video and is lowest when the target edges are high-contrast. The tracker has fewer false negatives than the detector.

This figure also shows the contribution of the tracker compared to detection alone. The tracker decreases false negatives at the expense of increasing false positives. The system performance is affected by many factors and the poorest performance occurs under conditions of heavy marine show, low-contrast targets, and fast camera motion. Performance also suffers if the background conditions in the image change.

1.5 Contributions

The first contribution of this thesis is an overview of existing methods related to this work and an evaluation of their suitability for underwater applications with significant levels of marine snow. Techniques such as de-hazing and rain removal often rely on prior knowledge that is not available in this application or assumptions that do not hold in an underwater environment. Much work on underwater image processing requires better visibility and less marine snow. On the other hand, SMC

methods create a framework to incorporate uncertain and noisy observations.

The second contribution is the overall system design. Existing techniques are evaluated and adapted for use in the larger system. Where gaps exist between system components, new methods are developed. This includes determining how to assign uncertainty to edge location, and how to compare results from the particle filter to results from an edge detector. Testing and verification of the system includes an explanation of how parameters can be tuned by an ROV operator, and limitations on the conditions under which this system is effective.

1.6 Thesis Outline

Chapter two provides a survey of related work and their suitability for this application. Chapter three describes the proposed system. Chapter four discusses testing methodology and provides results. Chapter five states conclusions and suggests future work. Code and raw data are available online [Gamroth, 2010].

Chapter 2

Background

One of the main challenges of this project is that many computer vision techniques rely on assumptions that do not hold in an underwater environment with significant marine snow. To address this challenge, a wide range of techniques were considered and their limitations were explored. This chapter summarizes these techniques and provides an evaluation of their suitability for this task. Pseudocode is provided wherever a particular implementation of a general method is discussed.

2.1 Atmospheric Conditions

Atmospheric conditions such as snow, rain, and haze are common sources of poor image quality. Much work has been done to remove their effects using appropriate models of the scene or the atmosphere. Many of the assumptions required for these models do not hold in an underwater environment with no ambient lighting.

One of the difficulties of removing the effects of bad weather is that the effects increase exponentially with the distance from the camera and can therefore not be removed by simple linear or space-invariant techniques [Narasimhan and Nayar, 2003]. However, this also means that bad weather can serve as a means for capturing scene structure. Two or more images of the same scene under different weather conditions can be used to estimate depth and recover scene contrast [Narasimhan and Nayar, 2002]. While this approach could prove valuable for a static camera underwater, it is not suitable for a camera on a moving vehicle. As well, the pos-

sibility of scene structure being encoded by bad weather relies on the atmosphere being homogeneous haze rather than marine snow.

A related technique is the use of physics-based methods for improving contrast in colour images [Tan and Oakley, 2001, Oakley and Satherley, 1998]. These methods require prior knowledge of the distance from the camera to the objects, and an estimate of atmospheric conditions such as scattering coefficient and sky radiance.

Another interesting approach to minimizing atmospheric effects is haze removal from a single image without any depth information by relying on the "dark channel" prior [He et al., 2009]. This approach is based on the observation that in most haze-free outdoor scenes, a majority regions have some pixels (called dark pixels) that have very low intensity in at least one colour (rgb) channel. This trend occurs because many natural scenes contain shadows or saturated colours. In hazy images, then, the dark pixels have the intensity of air light.

The effects of rain and snow are globally predictable in the frequency domain [Barnum and Narasimhan, 2009]. When falling rain or snow appears as bright streaks in an image, it can be modelled as blurred Gaussians and a frequency model can be created [Barnum et al., 2007]. That model can be used as a snow detector in still images and videos. This techniques assumes that the streaks caused by snow or rain occur in predictable directions and the approximate shape of the streak is constant; neither of these assumptions hold in the underwater environment.

There are several other methods for rain detection and removal, including [Starik and Werman, 2003] and [Zhang et al., 2006]. One interesting approach to rain detection in videos is based on a model of the photometric and dynamic properties of rain [Garg and Nayar, 2003]. Candidate rain pixels are identified using photometric constraints: rain causes a temporary and linear increase in pixel intensity. It is assumed that the raindrops are imaged as streaks in an unknown but consistent direction. Dynamic constraints require that rain pixels have spatiotemporal correlations along the direction of the streaks. Rain is detected by selecting pixels in each frame that meet these constraints. This approach could be used in underwater environments where the motion of the snow is somehow constrained, such as with a stationary camera and a steady current that dictates the flow of marine snow.

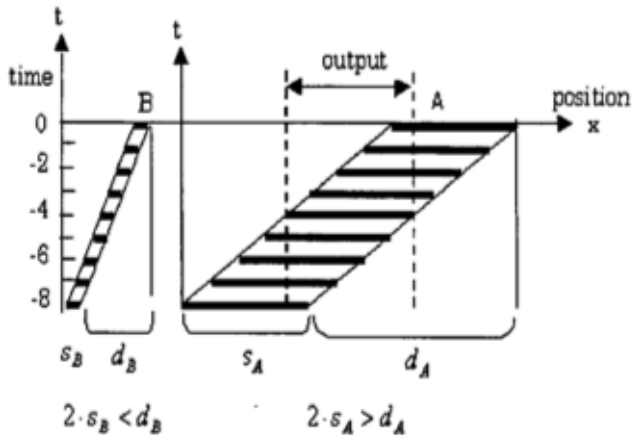


Figure 2.1: Relationship between the size, s_I and displacement d_I of an object I and the effect of temporal median filtering. Small objects are filtered out and larger objects are retained. (image from [Hase et al., 1999])

Finally, a temporal median filter can be used to remove snow from video [Hase et al., 1999]. This assumes that the snow is relatively small and fast-moving compared to the objects of interest in the video. Any objects that occupy a given pixel for less than half the duration of the median filter will be removed. Consider an object A of size s_A that moves d_A pixels in t seconds. The object will be removed by a filter of size t unless the condition $2s_A \geq d_A$ is met.

Figure 2.1 illustrates the effect of temporal median filtering. Object B is small and fast-moving relative to the filter width t and will be removed. Object A is relatively large and slow-moving and will not be removed; it will be shown at the correct size at an average location. An object with size and velocity between that of A and B would be partially removed and would appear smaller than the true object. This technique is well-suited for removing marine snow.

2.1.1 Optical Snow

Optical snow is "a generalization of optical flow in which the assumption of local spatial continuity of the motion field is abandoned" [Langer and Mann, 2003]. It does not assume a continuous motion field and, unlike layered motion, it does not require smoothness in layers; optical snow allows for discontinuities in motion at

any point in the image. Optical snow describes many natural situations, including the fall of snowflakes and camera motion in highly cluttered scenes.

Given prior knowledge of an appropriate motion model, optical snow can be used to measure properties of the scene or camera motion. The case of parallel optical snow describes the lateral motion of a camera or snow falling. Methods are available for measuring the direction and range of speed of the snow [Langer and Mann, 2003]. Similarly, the translation and roll of a camera can be estimated by modelling motion parallax [Mann and Langer, 2004]. However, the problem of model selection for a particular application is poorly understood.

The marine snow in this application is well described as optical snow since it is possible for each "snowflake" to move independently of the background and other snowflakes. However, a reasonable model is not available to describe the motion of the optical snow.

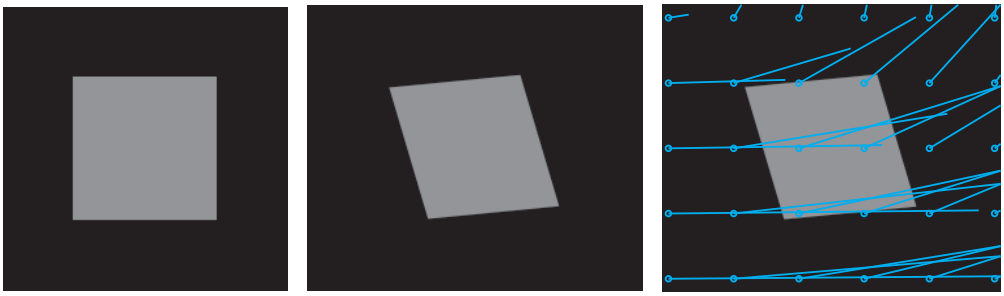
2.2 Motion Estimation

Lucas and Kanade proposed a technique for image registration that considers the spatial intensity gradients of two images and uses Newton-Raphson iteration to find the disparity vector (transform), h, that minimizes the difference between the images [Lucas and Kanade, 1981].

This algorithm was implemented and extended to calculate a global first-order flow model relating two images [Young, 2010]. The flow model has six parameters which describe an affine transformation: x translation v_{x_0}, y translation v_{y_0}, dilation d, rotation r, skew along primary axis s_1, and skew along secondary axis s_2. If a pixel at image location (x,y) is represented by the vector (v_x,v_y), then the first-order flow model is given by Equation 2.1.

$$\begin{bmatrix} v_x & v_y \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} d+s_1 & s_2+r \\ s_2-r & d-s_1 \\ v_{x_0} & v_{y_0} \end{bmatrix} \tag{2.1}$$

This equation describes a spatially continuous transformation from one image to another and allows features in one image to be mapped to the corresponding location in the other image. Since it is a global parameterization, it provides flow



(a) Initial image **(b)** Transformed image **(c)** Flow field

Figure 2.2: Affine Flow. (a) Square (b) Rhombus created by transforming the square using the following parameters: $d = 0, r = \pi/16, s_1 = 0, s_2 = 0.1, v_{x_0} = 0, v_{y_0} = 0$. (c) Affine flow field: circles denote the pixels for which the flow is shown and lines denote the magnitude and direction of the flow. The advantage of global parameterization is that flow is calculated for featureless regions.

estimates for textureless regions of the image; this is an advantage for underwater scenes with sparse features. Optical flow estimation is illustrated in Figure 2.2. Algorithm 1 describes the particular implementation.

Algorithm 1: Affine Flow [Young, 2010]

input : intensity images I_1, I_2
output: flow parameters, $f = [v_{x_0}, v_{y_0}, d, r, s_1, s_2]$

1 smooth images I_1 and I_2 by convolving with a Gaussian;
2 calculate spatial intensity gradients G_x and G_y by 2.2;
3 calculate temporal intensity gradient G_t by 2.3;
4 solve for the flow, f, in 2.4 by Gaussian elimination;

$$\begin{aligned} G_x &= (I_1 + I_2) * [0.25, 0, -0.25] \\ G_y &= (I_1 + I_2) * [0.25, 0, -0.25]' \end{aligned} \tag{2.2}$$

$$G_t = I_1 - I_2 \tag{2.3}$$

$$Af = -G_t \tag{2.4}$$

Where A is defined as follows:

$$\begin{aligned} x_{G_x} &= x * G_x \\ y_{G_x} &= y * G_x \\ y_{G_y} &= y * G_y \\ x_{G_y} &= x * G_y \\ A &= [G_x, G_y, x_{G_x} + y_{G_y}, x_{G_y} - y_{G_x}, x_{G_x} - y_{G_y}, y_{G_x} + x_{G_y}] \end{aligned}$$

2.3 Underwater Video

Many techniques have been developed specifically for underwater images or videos but they often require the input images be in a high-contrast, high-visibility environment with little marine snow [Rzhanov et al., 2000], [Salvi et al., 2008]. In these cases, pre-processing is usually limited to contrast enhancement.

A commercially available system called Lyyn enhances images by performing histogram equalization on each colour channel individually and then recombining the colour channels [Lyyn, 2010, Holm and Holm, 2008]. Another enhancement strategy uses a multi-step process to increase contrast in local image patches [Salvi et al., 2008]. In this strategy, the first step is to normalize the brightness using a homomorphic filter; this also reduces intensity gradients due to non-uniform lighting. Then, Contrast Limited Adaptive Histogram Equalization (CLAHE) followed by bilinear interpolation is used to enhance local contrast [Pizer et al., 1987]. Finally, adaptive noise-removal filtering is performed to minimize noise, particularly in areas with small intensity gradients. Both of the above methods perform poorly in the presence of marine snow.

One system was developed specifically for detection and tracking of low-contrast translucent targets in a marine environment, despite the presence of marine snow. Background subtraction is performed to minimize lighting and lens effects. This

is done for the red, green, and blue channels separately by subtracting the time-average pixel intensity [Walther et al., 2004]. Once the background has been subtracted, every fifth frame is analyzed using oriented filter responses in a pyramid using steerable filters at four orientations [Simoncelli and Freeman, 1995]. A saliency detector tuned to respond to oriented edges then extracts the areas of interest.

We implemented this system but it was not successful for the videos under consideration. The poor performance is probably because the saliency detectors were not discriminative enough for our application. Also, the system was developed for use in shallow water with the sky acting as a diffuse light source; these lighting conditions could have resulted in lower-contrast marine snow and more easily visible targets relative to the conditions in our videos. This could not be confirmed since the raw video from the original work was not available.

2.4 Background Subtraction

Marine snow can be considered "background" in the sense that it is clutter that prevents the easy identification of "foreground" targets. Many background subtraction algorithms require a stationary camera or the ability to train on the background.

One approach that allows a moving camera in an unknown environment relies instead on building a real-time model of the background by tracking the trajectories of salient features. Then, other objects with similar trajectories are labelled as background [Sheikh et al., 2009]. This assumes that the foreground can be differentiated from the background by a difference in motion.

Dynamic textures are spatiotemporal image patches that have regular but non-repeating characteristics, such as smoke, foliage, and waves [Doretto et al., 2003]. Instead of physics-based modelling to classify or generate these video segments, dynamic textures use an appearance-based model. The model parameters are learned from training video sequences. Once a dynamic texture model has been learned, it can be used for motion segmentation [Fazekas et al., 2009] or background subtraction [Mahadevan and Vasconcelos, 2008]. This model has been successfully applied to dynamic scenes with a moving camera. However, the model does rely on the ability to learn an appropriate model of the background dynamic texture. This might be possible for some underwater scenarios but not for the videos in

question where the marine snow lacks regularity.

2.5 Edge Detection

Edge detection and line extraction are standard techniques in computer vision. In this context, edges are defined as a binary property of each pixel whereas lines (or line segments) are vectors in the image plane. Often, an edge detector is used to create a binary image which then becomes input to a line extraction algorithm. Well-known methods for edge detection and line extraction are the Canny edge detector and Hough transform, respectively.

The Canny edge detector uses the following three criteria: good detection, good localization, and one response to a single edge [Canny, 1986]. Additionally, the detector should work well in noisy images. This method is described by Algorithm 2.

Algorithm 2: Canny Edge Detection

input : intensity image I
output: binary edge image

1 smooth image I by convolving with a Gaussian;
2 calculate the strength and orientation of the intensity gradient at each pixel by convolving with the first derivative of a 1D Gaussian in the x and y directions;
3 perform non-maximum suppression;
4 perform hysteresis thresholding;

The Hough Transform is a generic method for feature extraction and there are many variations and extensions including its use for extracting lines in images [Hough, 1962, Duda and Hart, 1972]. A line of infinite length is described by the shortest vector that connects the origin to any point on the line. The parameters of the line are the magnitude, ρ, and angle, θ of the vector. Algorithm 3 shows how a histogram with (ρ, θ) bins can be created to identify line segments in the image. Figure 2.3 shows the process of extracting line segments from an image.

The Hough transform is sensitive to noise and often returns false positives. The implementation in Algorithm 3 is prone to generating multiple overlapping line segments at slightly different orientations rather than one longer line. While

Algorithm 3: Hough Transform for Line Extraction

input : binary edge image I,
output: lines

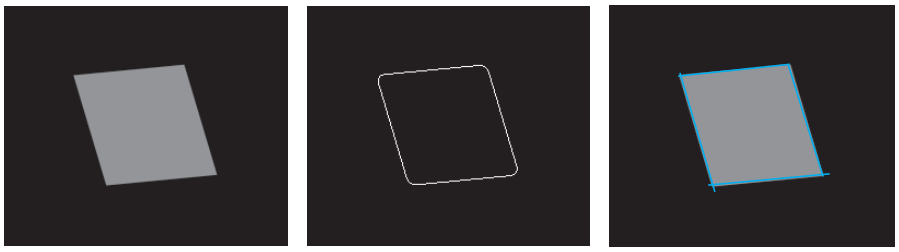
```
Perform Hough Transform;
create a 2D histogram with bins for (ρ, θ) to describe all possible lines;
for each edge pixel in I do
    for θ ∈ Θ do
        calculate the value of ρ;
        increment the value of the corresponding histogram bin;
    end
end
Identify the peaks in the parameter space;
select all histogram bins above a threshold;
iteratively select local maxima and suppress the neighbouring bins;
Find the endpoints of the line segments;
extract line segments in I associated with the peaks of the Hough Transform;
merge line segments that correspond to the same (ρ, θ) if they are separated
  by a gap smaller than a specified length;
retain lines longer than a minimum threshold;
```

parameter tuning can improve this, post-processing to merge overlapping line segments is a more robust solution.

An alternative to identifying lines in *x*-*y* images and tracking those lines over time is to study the *x*-*y*-*t* image volume. In this way, the energy (temporal edges) can be identified directly [Adelson and Bergen, 1985]. Recursive filtering and edge closing can be used to iteratively improve initial edge detections [Monga et al., 1989]. Medical imaging techniques search for edges in *x*-*y*-*z* in a similar fashion. However, for all of these methods, best results are obtained for high-contrast images when much is known a priori about the structure of the objects being imaged.

2.6 Sequential Monte Carlo Methods

This section reviews the basic particle filter described by [Doucet et al., 2001, Bengio, 1999]. A particle filter is a Sequential Monte Carlo (SMC) method that



(a) Original image **(b)** Edges **(c)** Extracted Lines overlaid on original image

Figure 2.3: Line extraction. (a) Object with strong gradients at edges. (b) Result of Canny edge detection. (c) Line segments extracted from edge detection using Hough transform and post-processing

provides a probabilistic framework for capturing the uncertainty inherent in many practical settings. As the name suggests, it is a simulation-based technique that uses a weighted set of particles (samples from a target distribution) to estimate a model of the system. The basic concept is that, at every iteration, the particles are evaluated against the most recent observations; particles that are a good match for the data are highly weighted and are propagated while those that are a poor match are given low weights and are extinguished. This process of re-weighting and re-sampling may continue indefinitely.

Particle filters can be used in many practical settings that require estimating unknown quantities (unobserved states) given incomplete observations. For example, a particle filter is appropriate when the exact location and pose of an object is uncertain due to atmospheric conditions, clutter, measurement noise, incomplete modelling, etc. A particle filter can be used to leverage prior knowledge about the object and gather evidence over time to create a probabilistic description of the object's true location and pose.

Several alternatives to the particle filter have historically been used. A Kalman filter can be used for data that are modelled by a linear Gaussian state-space model, where an exact analytical expression is available to calculate the posterior distribution. A Hidden Markov Model (HMM) filter can be used for partially observed data that are modelled by a finite state-space. In order to maintain mathematical tractability, these filters impose certain constraints on the data. However, prac-

tical problems often violate these constraints as they are often high-dimensional, nonlinear or non-Gaussian. In these cases, particle filters provide an alternative technique.

The following notation will be used throughout this thesis:

unobserved state	$\{\mathbf{x}_t; t \in \mathbb{N}\}, \mathbf{x}_t \in \mathscr{X}$
unobserved state up to time t	$\{\mathbf{x}_{0:t}\} \triangleq \{\mathbf{x}_0, ..., \mathbf{x}_t\}$
initial distribution	$p(\mathbf{x}_0)$
transition distribution	$p(\mathbf{x}_t \vert \mathbf{x}_{t-1})$ for $t \geq 1$
observations	$\{\mathbf{y}_t; t \in \mathbb{N}^*\}, t \in \mathscr{Y}$
observations up to time t	$\{\mathbf{y}_{0:t}\} \triangleq \{\mathbf{y}_0, ..., \mathbf{y}_t\}$
marginal distribution	$p(\mathbf{y}_t \vert \mathbf{x}_t)$ for $t \geq 1$
posterior distribution	$p(\mathbf{x}_{0:t} \vert \mathbf{y}_{1:t})$

The unobserved states are modelled as a Markov process. The observations are conditionally independent given the unobserved states and the marginal distribution. The goal of the particle filter is to recursively estimate the posterior distribution at each timestep. The marginal distribution or other function of interest can then be computed. Often, a Bayesian model is used to relate the prior distribution to the likelihood of the observations to generate a posterior belief. Bayes' Theorem in given in Equation 2.5.

$$p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_{1:t}|\mathbf{x}_{0:t})p(\mathbf{x}_{0:t})}{\int p(\mathbf{y}_{1:t}|\mathbf{x}_{0:t})(\mathbf{x}_{0:t})d\mathbf{x}_{0:t}} = p(\mathbf{x}_t|\mathbf{y}_{1:t})\frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t)}{\int p(\mathbf{y}_t|\mathbf{x}_t)(\mathbf{x}_t)d\mathbf{x}_t} \tag{2.5}$$

Figure 2.4 illustrates one iteration of a particle filter used to estimate a 1D signal. This example begins at time $t-1$ with N unweighted particles providing an estimate, $\hat{\mathbf{x}}_{\mathbf{t-1}}^{(\mathbf{i})}$, of the signal. The particles have only one parameter: location along the horizontal axis. Each particle is assigned an importance weight, $\tilde{\mathbf{w}}_{\mathbf{t-1}}^{(\mathbf{i})}$, proportional to the signal strength at that location. New particles are created in two steps. First, the resampling step generates a set of unweighted particles whose locations are determined by the importance weights. Next, the prediction step introduces variation into the set of unweighted particles. One iteration is complete and the N particles now provide the updated estimate $\hat{\mathbf{x}}_{\mathbf{t}}^{(\mathbf{i})}$. When the signal is observed again at time t, the particles are again re-weighted and the cycle continues. The details

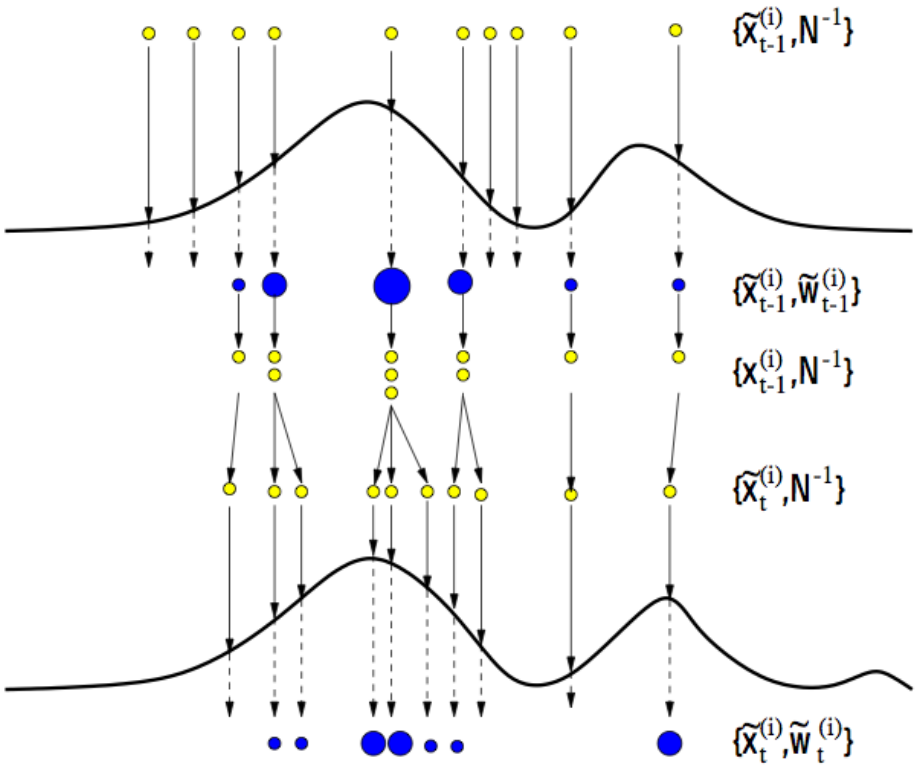


Figure 2.4: Iterative resampling of particle filter (image from [Doucet et al., 2001]). Unweighted (yellow) particles are an estimate of the 1D signal. Weighted (blue) particle reflect the signal strength at the particle location. Resampling iteratively moves particles towards peaks in the signal.

of this process are given in Algorithm 4.

The resampling step eliminates particles with low weights and multiplies those with high weights. This step requires as input the particle weights and returns as output a selection of particles such that weighted average of the initial particles is equal to the unweighted average of the selected particles. In this way, resampling is generic to all applications and does not require any domain knowledge. Several resampling schemes have been proposed, including residual resampling, minimum variance resampling, and multinomial resampling. The choice between these three options does not significantly affect performance.

The particle filter has many advantages, including its simplicity and ease of implementation. It is attractive for real-time applications as it is recursive without the need to recompute past samples or store prior observations, and it does not degenerate as t increases. Further, since each step in the process is modular and each particle can be computed independently of other particles (except for normalization), this technique is well-suited to a parallel implementation [Lee et al., 2009]. Many extensions and improvements have been proposed, including the unscented particle filter [van der Merwe et al., 2000], and the boosted particle filter [Okuma et al., 2004], [Lu et al., 2009].

The challenge in implementing a particle filter for a particular application is calculating particle weights, and in particular, determining how to compute the marginal distribution, $p(\mathbf{y}_t|\mathbf{x}_t)$.

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\hat{\mathbf{x}}_t^{(i)}) p(\hat{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\hat{\mathbf{x}}_t^{(i)}|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{y}_{1:t})} \tag{2.6}$$

$$\tilde{w}_t^{(i)} = w_t^{(i)} [\sum_{j=1}^{N} w_t^{(j)}]^{-1} \tag{2.7}$$

$$p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) \approx \hat{p}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{(\mathbf{x}_{0:t}^{(i)})}(d\mathbf{x}_{0:t}) \tag{2.8}$$

In general, particle filters perform poorly for the case of multi-modal data. Returning to the example in Figure 2.4, the particles would gradually migrate to the global maximum on the left and away from the local maximum on the right. Once the particles are concentrated in one location, they will fail to respond to a peak arising in another location, unless a particle happens to be sampled in the neighbourhood of that peak when variation was introduced in the prediction step. There are several approaches to dealing with multi-modal data, including multiple independent particle filters and mixtures of filters [Vermaak et al., 2003].

2.7 Summary

Many strategies were investigated, including the effects of atmospheric conditions, optical snow, underwater video techniques, background subtraction, optical flow

Algorithm 4: Particle Filter

input : observations $\mathbf{y}_{0:t}$
output: unobserved state $\mathbf{x}_{0:t}$

Initialization;
$t = 0$;
for $i \leftarrow 1$ **to** N **do**
    sample the states $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$;
end
for $t \leftarrow 1$ **to** T **do**
    Importance Sampling;
    for $i \leftarrow 1$ **to** N **do**
        sample $\hat{\mathbf{x}}_{\mathbf{t}}^{(\mathbf{i})} \sim q(\mathbf{x}_t^{(i)} | \mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{1:t})$;
        set $\hat{\mathbf{x}}_{\mathbf{0:t}}^{(\mathbf{i})} = (\hat{\mathbf{x}}_{\mathbf{0:t-1}}^{(\mathbf{i})}, \hat{\mathbf{x}}_{\mathbf{t}}^{(\mathbf{i})})$;
    end
    for $i \leftarrow 1$ **to** N **do**
        evaluate the importance weights $w_t^{(i)}$ by (2.6);
        normalize the importance weights $\tilde{w}_t^{(i)}$ by (2.7);
    end
    Resampling;
    multiply/suppress samples $\mathbf{x}_{0:t}$ with high-low importance weights $\tilde{w}_t^{(i)}$, respectively, to obtain N random samples $(\mathbf{x})_{0:t}^{i}$ approximately distributed according to $p(\mathbf{x}_{0:t}^{(i)} | \mathbf{y}_{1:t})$;
    for $i \leftarrow 1$ **to** N **do**
        $w_t^{(i)} = \tilde{w}_t^{(i)} = \frac{1}{N}$;
    end
    Output;
    approximate posterior distribution $p(\mathbf{x}_{0:t} | \mathbf{y}_{1:t})$ using the samples $(\mathbf{x})_{0:t}^{i}$ by 2.8;
end

estimation, edge detection, line extraction, and SMC methods. In some cases, the required assumptions are not appropriate for the problem under consideration. However, many techniques are suitable, including temporal median filtering, Lucas-Kanade first-order optical flow, Canny edge detection, Hough transforms and SMC methods. Implementation details such as equations and pseudocode are provided for these suitable techniques.

Chapter 3

System

3.1 Problem Description

The goal of this project is the automatic detection and tracking of man-made objects in underwater environments with poor visibility. The system is tested on video from an ROV.

The video was recorded in July, 2009, in Saanich Inlet, BC, by a Seaeye Falcon ROV at depths of approximately 100m below the surface. The camera was a Seaeye CAM04N with a field of view of 90° in water and a sensitivity of 0.35 Lux. The iris, gain control and exposure adjustment were all automatic. Vehicle-mounted cameras provided all lighting as no light from the surface penetrates that deeply. The video was initially used for navigation purposes during an inspection of an underwater structure.

The video has a frame rate of 30 Hz and resolution of 720x480 in RGB. The resolution is 340x185 after downsampling to remove interlacing, converting to a single-channel (intensity) image and cropping to remove video overlay artifacts. The video frames suffer from poor visibility due to turbidity and marine snow.

The marine snow in the video is visual clutter caused by planktonic organisms and particulate matter suspended in the water. The appearance of the snow varies spatiotemporally and ranges from large, bright fish to barely visible particles that create a hazy environment. There are many possible causes for the motion of marine snow, including currents, moving water from the ROV's thrusters, and self-

(a) Seaeye Falcon ROV

(b) Seaeye CAM04N video camera

Figure 3.1: ROV and camera used for data collection

propelled organisms. The motion of the the ROV also creates apparent motion of the marine snow. With so many dynamic factors contributing to the apparent motion, it is unlikely that an a priori model could be used to fully describe the optical flow.

3.2 System Description

The proposed system consists of pre-processing, optical flow calculation, and detection and tracking of a set of long, continuous straight line targets that persist over time. The full system is summarized in Algorithm 7. Code is available online [Gamroth, 2010].

Though it is not optimized for real-time performance, design choices allow for this possibility. The system does not rely on having knowledge of the future (i.e. it is causal), nor does it require any prior knowledge of the environment that an ROV operator could not obtain in a short initialization step.

3.2.1 Pre-processing

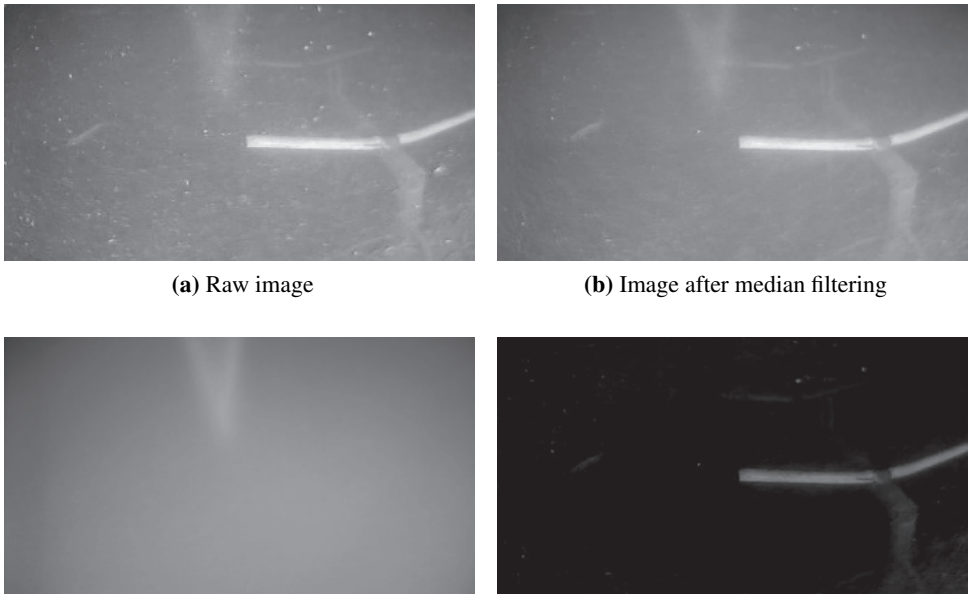
Each image is pre-processed to minimize marine snow, remove background artifacts and convert to a single channel. The first step is conversion from a three-channel (RGB) image to a single-channel (gray) intensity image using Equation 3.1. This conversion is used in Matlab and reflects humans' perception of the illuminance of the three colours. However, it is not optimized for an underwater environment. A line of future investigation would be to adapt the full system to operate on an RGB image or consider using alternative colour spaces such as TCbCr which may offer greater robustness to underwater photometric noise [Qi and Cooperstock, 2007].

$$gray = 0.2989R + 0.5870G + 0.1140B \tag{3.1}$$

The second pre-processing step is reducing marine snow through temporal median filtering. The value of each pixel is set to the temporal median of that pixel over the past t images. Since most of the optical snow moves quickly relative to the targets, the proper selection of t causes the snow to be filtered out while the targets are retained. Too low a value of t results in the optical snow not being filtered out. Too high a value results in slower-moving objects being filtered out, or the location of the edges being mis-identified (in particular, the pixels along the leading edge being replaced by background pixels). Therefore, if a higher value of t is required due to higher levels of snow, the camera or ROV must move more slowly. Figure 3.2 gives an example of the results of temporal median filtering.

Finally, background subtraction is performed to remove lighting and lens artifacts, and parts of the ROV that are visible. This requires a few seconds of video in which there are no objects of interest. For each pixel, the temporal median is calculated and the resulting image is called the "background". In the final pre-processing step, the background is subtracted from the image. Figure 3.2 gives examples of the background image and the results of background subtraction.

During real-world operation of the vehicle, the background may change when, for example, the camera tilts so that the ROV's manipulator is visible. In some cases, the operator could reset the background image but this may not always be feasible. Adaptive background subtraction is an area for future work.



(a) Raw image

(b) Image after median filtering

(c) Background image

(d) Image after median filtering and background subtraction

Figure 3.2: Image pre-processing. (a) Original image. (b) Temporal median filtering over 5 video frames. Marine snow is reduced at the expense of losing information about small or fast-moving targets. (c) Background image identifies light artifacts including a beam of light in top centre and gradual darkening at perimeter of image. (d) Pre-processing reduces noise and lighting artifacts.

3.2.2 Motion Estimation

The scene motion can be accurately described as "optical snow" as defined in Section 2.1.1 because it is characterized by spatial and temporal motion discontinuities. Although the background and targets are stationary, rigid and relatively textureless, the marine snow is ubiquitous, high-contrast and unpredictable.

Scenes that can be readily analyzed using optical snow filtering techniques tend to involve globally consistent motion that can be modelled a priori, such as rain falling in straight lines or forward motion through a cluttered scene. In this problem, no such model is available. In some special cases, the motion of the marine snow is simple and could be modelled, such as when the snow itself is

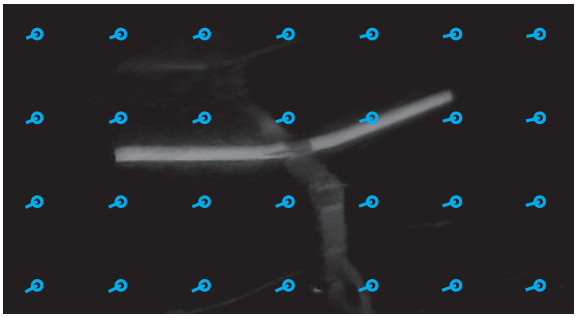


Figure 3.3: Optical flow between two subsequent video frames. The circles identify the pixels for which flow is displayed; the direction and magnitude of the lines represents the flow. The assumption of (global) affine flow provides flow estimates in featureless regions.

stationary and the only motion is relative to a translating camera. However, in most cases, the motion of the marine snow is far more complex, such as when the snow is caught in turbulent water from the ROV's thrusters or the camera is moving and rotating in an unpredictable manner. Therefore, the motion is not estimated using optical snow methods.

Since the scene is relatively textureless and sparse in features, any attempts to find local matches between images yield few, if any, unique matches. The snow is largely homogeneous and any "flake" of optical snow is a relatively good match to any other. As a result, methods for estimating motion that allow for local discontinuities were found to be prone to error and often assume zero motion at all locations except edges of bright objects where the intensity gradient can be reliably matched. Therefore, global assumptions about motion are required.

It is assumed that the optical flow between two subsequent frames can be approximated by an affine transform. Over short enough timescales, factors such as motion parallax and occlusion have negligible effects. With this assumption, the Lucas-Kanade method can be applied. The sparsity of features and textures means that intensity-matching in the Lucas-Kanade algorithm is dominated by the object of interest rather than the snow. A visualization of the results optical flow calculations is shown in Figure 3.3.

There are conditions under which the affine approximation will fail. For exam-

ple, large non-stationary objects like fish cannot be described by affine motion. For our dataset, this does not appear to be a problem. Firstly because these cases are rare, and secondly because the motion estimate is used as an input to a system that is robust to error or inaccuracy. The affine approximation may also fail if background subtraction is not done properly. Bright, stationary artifacts will force the flow to zero at those locations.

The accuracy of the flow estimate depends on the geometry of the scene. Faint edges have less influence on the results than high-contrast edges and better results are obtained when there is more evidence. As a result, the best estimates are obtained when the scene contains several strong intensity gradients that are well-separated in space.

It is also important to note that this does not provide a true estimate of camera motion, only of the visual difference between two video frames. For example, if the camera is moving in an empty environment, the optical flow will be zero regardless of the actual ROV motion.

An avenue of future work would be to incorporate additional data into the motion estimation, such as the control signals sent to the ROV thrusters, or measurements from depth sensors, accelerometers or flow meters.

3.2.3 Edge Detection and Line Extraction

Edge detection is performed using the Canny edge detector and line extraction is performed using a Hough transform. Parameters for these methods were chosen by trial and error. The standard deviation *sigma* of the Gaussian for the Canny edge detector is set to 3; the spacing of the Hough histogram bins is 2° along the θ axis and 3 along the ρ axis.

Although the pre-processing step reduces noise, the resulting image is not clear enough for perfect edge and line detection. The result is many missed detections and false positives as well as overlapping true positives. A post-processing step is needed to improve results.

Figure 3.4 shows the results of edge detection and the extracted line segments are overlaid on the image. To reduce the number of false positives, the lines returned by the Hough transform are evaluated using a Gabor filter and any lines

with a filter response below a threshold are discarded. The Gabor filter is also used to determine the true orientation of the line, since the Hough lines are insensitive to the sign of the gradient.

The overlapping true positives are undesirable since they add complexity and uncertainty to the system. Many of the overlapping lines are segments of the same true line and are therefore merged to generate one longer line. Figure 3.4 shows the results after post-processing to merge overlapping lines.

3.2.4 Gabor Filter

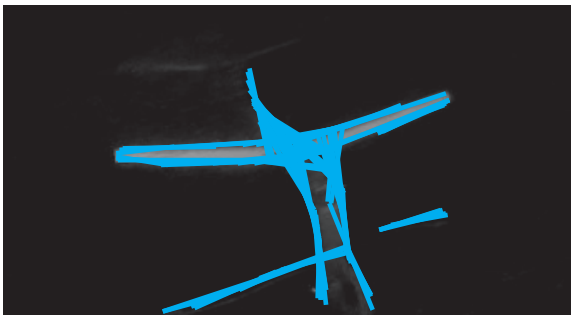
The edge strength, or fit of a line segment, is evaluated using a Gabor filter [Gabor, 1946]. This is done by constructing a filter based on the parameters of the line. The Gabor filter is centred on the (x,y) centre of the line segment, oriented at θ degrees, and has a length proportional to the length of the line segment. The width of the filter is also proportional to the length to allow greater uncertainty of the exact location or orientation of longer line segments and to prevent line segments from degenerating to points. The angle, θ, of the filter ranges from 0 to 360 to capture the direction of the edge as well as the direction of the intensity gradient.

Equation 3.2 describes a Gabor filter. The filter is a function of location (x,y), frequency λ, phase ψ and orientation θ of a sinusoid, standard deviation σ of a Gaussian, and aspect ratio γ.

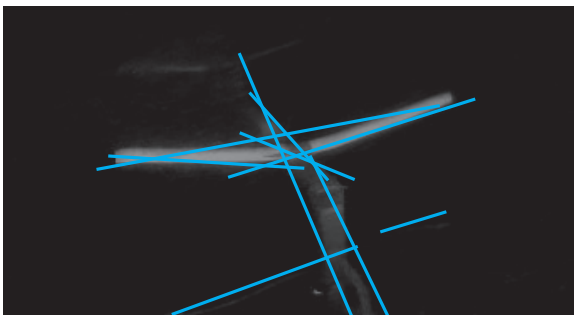
$$\begin{aligned} g(x,y,\lambda,\theta,\psi,\sigma,\gamma) &= \exp\left(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda}+\psi\right) \qquad (3.2)\\ x' &= x\cos\theta + y\sin\theta \\ y' &= -x\sin\theta + y\cos\theta \end{aligned}$$

A disadvantage of Gabor filters for applications with marine snow is that the filter responds equally strongly to one long, low-contrast edge in the direction of the filter and one small, high-contrast dot. Additionally, it is difficult to select a threshold for a Gabor filter that will eliminate false positives without also eliminating the low-contrast true positives. An improvement on this approach is proposed whereby we replace the Gabor filter with several smaller Gabor filters along the

(a) Binary edge image



(b) Line segment extraction



(c) Line segments after post-processing

Figure 3.4: Line segment extraction. (a) The edge detector successfully detects and localizes the higher contrast edges but is less successful with weaker edges. (b) A post-processed Hough transform returns many overlapping line segments that fit the binary edge image. (c) Additional post-processing merges line segments.

same axis. We compare the results of the smaller filters: similar values across all segments indicates a long line and varying results among segments indicates marine snow.

Figure 3.5 shows a standard Gabor filter and the corresponding smaller filters. Note that the filter in subfigure (b) is for visualization purposes only and is not convolved with the image.

In practice, we found many cases where we wish to allow a small portion of the smaller filters to have a higher or lower response than the rest. For example, some true positives have a break due to an occlusion, and sometimes the extracted line is improperly located on the true line, causing low filter responses at one or both ends. To allow for these cases, we considered the 70th percentile response and divide that by the response of the standard Gabor filter. This process is described in Algorithm 5 and was found to improve results when evaluating which tracked edges should be propagated to future time steps. The standard Gabor filter is used for initial edge detection.

Algorithm 5: Modified Gabor Filter

input : intensity image I, line (x, y, θ, l)
output: edge strength

1 generate a Gabor filter using the line parameters (x, y, θ, l) and convolve with I to calculate the filter response R;
2 generate a set of smaller Gabor filters equally spaced along the line (x, y, θ, l) and convolve each with I to calculate the responses $r^{(i)}$;
3 calculate $g^{(i)} = \frac{r^{(i)}}{R}$ for each smaller filter;
4 return a given percentile of $g^{(i)}$

3.2.5 Line Similarity

In several cases, it is necessary to quantify the similarity between two lines. The Hough transform often returns several line segments that form parts of one long edge on a target object. Targets that are initialized as two separate line segments will sometimes converge over time and it is desirable to identify this situation and merge the two targets. Also, detected line segments need to be evaluated to determine if they provide support for an existing tracked line or if they should be used

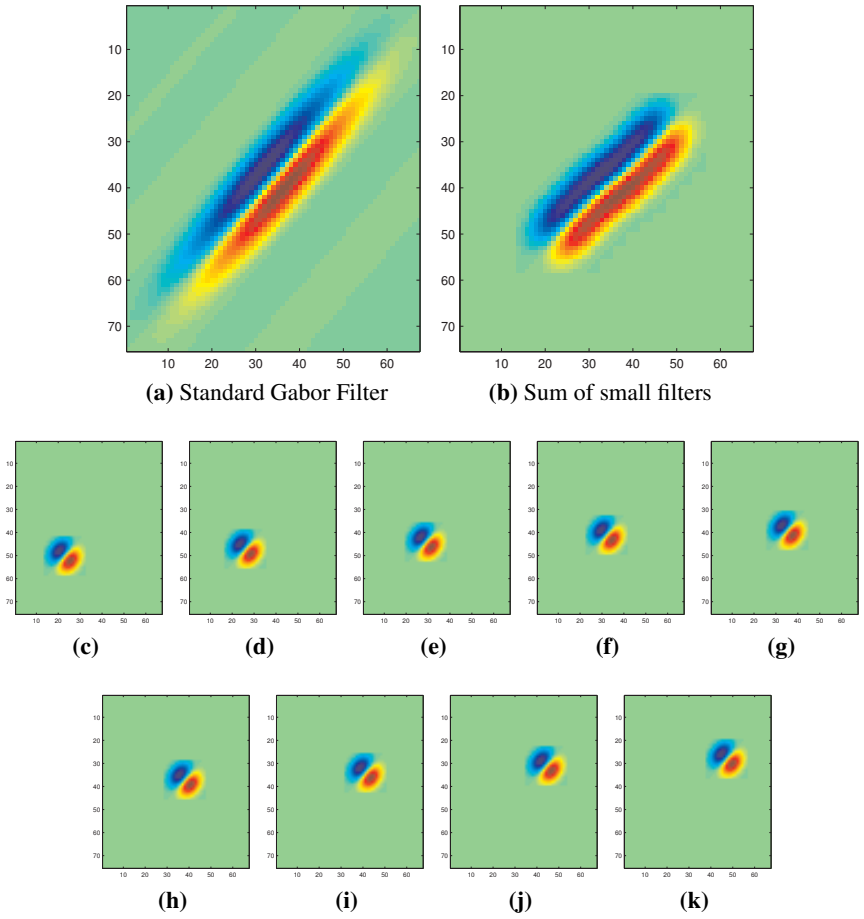
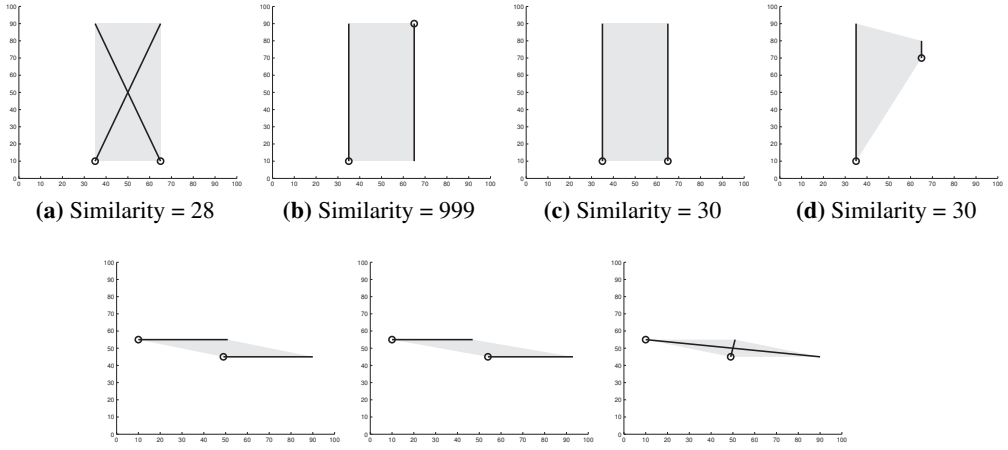


Figure 3.5: Gabor Filter. (a) The Gabor filter cannot differentiate between a long, low-contrast edge and a short high-contrast edge. (b) The overall shape of a Gabor filter can be recreated by adding together a bank of smaller Gabor filters. (c-k) Filter bank of multiple Gabor filters. When an image is convolved with the filter bank, the relative response strengths differentiate between a long low-contrast edge and a short, high-contrast edge.



(a) Similarity = 28 **(b)** Similarity = 999 **(c)** Similarity = 30 **(d)** Similarity = 30

(e) Similarity = 10 **(f)** Similarity = 999 **(g)** Similarity = 999

Figure 3.6: Line similarity. Circle denotes start of each line and thereby the positive/negative orientation. Dissimilar lines filtered out in the first step are shown with a similarity of 999.

to initialize a new target. In all cases, lines should be compared and those that meet some similarity criterion can be paired.

Intuitively, the location and angle of the lines are important and interdependent. For example, collinear line segments could be considered to be similar even with a large distance between their centre points, while orthogonal lines should not be considered similar even if their centre points coincide. The angle of the line must represent the direction of the intensity gradient in the image so that, for example, the top and bottom edges of a thin white bar are not considered matches for each other.

A similarity measure that meets these criteria was developed. First, a filtering step identifies dissimilar lines whose orientations differ by more than 45° or that are too far apart to overlap at all (in other words, the centre points are separated by a distance greater than half the sum of the lengths of the lines). Second, lines that were filtered out in the first step are compared by calculating the area of the quadrilateral described by the four line end points and normalizing by the average length of the lines. This calculates the area per unit length. Lines with a correspondence value of zero are collinear, and less-similar lines have larger values. For

convenience, the lines filtered out in the first step are assigned an artificially high value.

As illustrated in Figure 3.6, this approach produces reasonable results. However, the discontinuity introduced in the filtering step can lead to instability in cases like Figure 3.6 (e) and (f). The line segments in (e) overlap while those in (f) do not. This behaviour can be advantageous when it prevents several short streaks of marine snow from being interpreted as one longer line segment, but disadvantageous when it prevents two halves of a long, low-intensity edge from being connected. Most tracked targets are not affected by this property of the similarity measure, but it is possible that a different approach to measuring similarity could improve system performance in some challenging cases.

3.2.6 Sequential Monte Carlo Tracker

Edge detection and line extraction are not adequate on their own as a means of identifying target locations over time. The sparseness of image features (such as SIFT features [Lowe, 2004]) precludes tracking by matching across images. Instead, evidence from edge detections should be propagated forward in time, allowing evidence from past detections to increase the probability of target identification at the current time step. Particle filtering, a Sequential Monte Carlo (SMC) method, provides a framework for capturing this uncertainty.

As described in Section 2.6, there are four main steps to a particle filter: initialization, importance sampling, weighting, and resampling. This process is summarized in Algorithm 6 and is discussed in this section. Section 3.2.7 addresses the details of how detected edges are matched to a particle filter and how multiple edges are handled.

We rely on the terminology introduced in Section 2.6. The unobserved state (true edge location), $\mathbf{x}_t$, is modelled as a Markov process with initial distribution $p(\mathbf{x}_0)$ and transition equation $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The observations (images) $\mathbf{y}_t$, are assumed to be conditionally independent given the process $\mathbf{x}_t$ and marginal distribution $p(\mathbf{y}_t|\mathbf{x}_t)$.

A new particle filter with $N = 30$ is initialized by sampling the initial distribution $p(\mathbf{x}_0)$ from a four-dimensional Gaussian centred on an edge in the image using

Equation 3.4. Each particle represents an edge centred at (x, y) with length $2l$ and orientation of θ degrees. This parameterization was chosen because the line orientation is typically much closer to ground truth than the line length and the system is simpler and more intuitive if these variables can be sampled with zero covariance while allowing greater variation in length than orientation. Although it might seem desirable to allow, for example, the centre point of a line to have greater variance along the direction of the line than tangent to the line, it was found that the simpler approximation was acceptable. The assumption of zero covariance would not have been appropriate had the lines been parameterized using the (x, y) coordinates of the two endpoints.

Once the particles have been initialized, they are evaluated at each subsequent time step. The first step is to transform the particles using the optical flow from the previous to the current image, as described in Section 3.2.2. Each particle is transformed by finding the two (x, y) endpoints, applying the transformation in Equation 2.1, and converting back to the (x, y, θ, l) line parameterization.

Then, particles are resampled. A mixture proposal is used whereby a fraction, α, of the particles is sampled from the corresponding edge extracted at time t and the remaining $(1 - \alpha)$ are sampled assuming a random walk diffusion from the previous time step. This is given by Equation 3.5 and serves to combine evidence from both sources.The particle filter becomes unstable when α is too high because of the variation in the edge detections from image to image. When the edge detector does not return a matching edge, α is set to 0.

The next step, assigning particle weights, determines which particles best match the observations. First, the edge strength, G, is calculated by convolving the image with a Gabor filter centred on the particle. The observation likelihoods are calculated using Equation 3.6 which measures how similar the edge is to a black-white step edge. Finally, the importance weights, $w_t^{(i)}$ are calculated using Bayes' Theorem in Equation 3.7 and then normalized.

The final step, resampling, is done using residual resampling [de Freitas, 1998]. It is expected that results would not be significantly affected by the choice of resampling scheme and therefore minimum variance sampling or multinomial sampling could also have been used.

Algorithm 6: Particle Filter

input : observations $\mathbf{y}_t$
output: unobserved state $\mathbf{x}_t$

Initialization;
$t = 0$;
extract target edge with parameters (x, y, θ, l) from $\mathbf{y}_0$;
set $\mu_0 = (x, y, \theta, l)$;
for $i \leftarrow 1$ **to** N **do**
    sample the states $\mathbf{x}_0^{(i)}$ by (3.4);
end
for $t \leftarrow 1$ **to** T **do**
    estimate optical flow and apply transform to particles;
    Importance Sampling;
    for $i \leftarrow 1$ **to** N **do**
        sample $\hat{\mathbf{x}}_t^{(i)}$ by (3.5);
    end
    for $i \leftarrow 1$ **to** N **do**
        compute the observation likelihood for $\mathbf{x}_t^{(i)}$ by (3.6);
        evaluate the importance weights $w_t^{(i)}$ by (3.7);
        normalize the importance weights $\tilde{w}_t^{(i)}$;
    end
    Resampling;
    generate unweighted samples $\tilde{\mathbf{x}}_t^{(i)}$ by resampling $\mathbf{x}_t^{(i)}$ according to the importance weights $w_t^{(i)}$;
    for $i \leftarrow 1$ **to** N **do**
        $w_t^{(i)} = \tilde{w}_t^{(i)} = \frac{1}{N}$;
    end
    Output;
    approximate edge location μ_t by posterior distribution $p(\mathbf{x}_t|\mathbf{y}_t)$ in (3.8);
end

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = e^{\frac{-(\mathbf{x}_t^{(i)})-\mathbf{x}_{t-1}^{(i)})^2}{2\sigma^2}} \tag{3.3}$$

$$q_e(\mathbf{x}_t|\mathbf{y}_t) = e^{\frac{-(x_t^{(i)}-\mu_t)^2}{2\sigma^2}} \tag{3.4}$$

where μ_t is the (x, y, l, θ) of the edge detection at time step t.

$$q(\mathbf{x}_t^{(i)}|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{y}_{1:t}) = \alpha q_e(\mathbf{x}_t^{(i)}|\mathbf{y}_t) + (1-\alpha)p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}) \tag{3.5}$$

where q_e is a Gaussian distribution centered at the edge detection and p uses a random walk diffusion model

$$p(\mathbf{y}_t|\mathbf{x}_t) = e^{\frac{-(1-G_t^{(i)})^2}{2\sigma^2}} \tag{3.6}$$

where $G_t^{(i)}$ is the correlation of $\mathbf{y}_t$ with the Gabor filter (3.2) described by particle $\mathbf{x}_t^{(i)}$.

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{1:t-1}^{(i)}, \mathbf{y}_{1:t})} \tag{3.7}$$

$$p(\mathbf{x}_t|\mathbf{y}_t) \approx \hat{p}(\mathbf{x}_t|\mathbf{y}_t) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_t \tag{3.8}$$

3.2.7 Multiple Monte Carlo Trackers

Particle filters are not well-suited for multi-modal data as they tend to converge to the global maximum while losing support for other maxima. Since our goal is to track an unlimited number of target edges, we require a way to extend particle filters to track multiple objects. This also requires a means to match detections in the image to existing targets, to initialize new targets, to merge overlapping targets, and to eliminate low-confidence targets.

The approach to tracking multiple targets is to create multiple, independent particle filters. This simple method is appropriate since the targets are well-separated

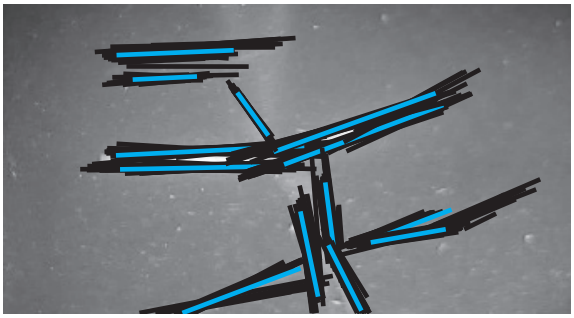


Figure 3.7: Multiple targets tracked by particle filters. The blue lines represent the mean particle for each tracker. The black lines are the individual particles.

in the parameter space and targets are not required to maintain separate identities (if two targets cross, it isn't important which tracker follows which target). The disadvantage of this approach is that the computational time increases with each new target. However, the increase is linear with the number of targets and particle filters can be computed in parallel.

The line similarity measure in Section 3.2.5 is used to match detected line segments to existing particle filters. In general, there is no more than one match per particle filter, but in the case of multiple matches, the best match is used. It is possible for a single detection to be matched to two different targets, but likely that those two targets will be merged in a subsequent step. If no match is identified, the particle filter simply proceeds without sampling from a new edge detection. Detected edges that are not matched to an existing target are used to initialize new particle filters.

Figure 3.7 shows multiple particle filters, each tracking a different target edge. The black lines represent individual particles. The trackers with the greatest variation in particles are those with the largest difference between tracked edge and matching detection.

To avoid maintaining redundant targets, the means of all particle filters are compared using the line similarity measure. If two targets are deemed to be similar, only the particle filter with the strongest mean edge strength is retained. An avenue for future work is to investigate other means of merging two particle filters.

The edge strength of the mean particle is evaluated at each time step. Targets are removed if the mean strength falls below a threshold. There is a hysteresis loop in that the edge strength threshold for initial detection is higher than the threshold for target propagation. This successfully reduces the number of false targets but the system is sensitive to these parameters. In a real-time system, it is recommended that the operator have direct, real-time control over these settings.

Although the particle filters are treated independently, they are affected by the same flow transform at each time step. This has the effect of constraining all targets to move in a cohesive fashion, consistent with the motion of a rigid object.

3.2.8 System Performance

The full system is described in Algorithm 7. This system can run indefinitely, iterates at every time step, and requires only the previous and current states as input.

In order to run indefinitely in real-time, a system must have memory and Central Processing Unit (CPU) requirements that are non-increasing with time and must not require knowledge of the future. This is true of our system. Temporal median filtering requires the past n images. Optical flow estimation requires the images at time t and $t-1$. Line extraction uses the image at time t. Tracking requires the image at time t and M particle filters, where M_t is determined by the number of targets at time t. Therefore, storage requirements do not increase with time and are only affected by the content of the video (number of targets). Similarly, the number of compute cycles per iteration is proportional only to the number of active targets.

This implementation is a proof-of-concept and is not optimized for computational speed. The current system, implemented in MATLAB, takes in the order of two seconds to process one video frame on a 2.53 GHz Intel core 2 Duo processor. However, a real-time system is possible, and the SMC framework lends itself to a GPU implementation [Lee et al., 2009].

3.3 Summary

The system described in this section detects and tracks target objects in an underwater environment, despite poor visibility and the presence of marine snow. The

Algorithm 7: System

input : observations (images) $\mathbf{y}_t$
output: unobserved state (true edge locations) $\mathbf{x}_{m,t}$ for each of m targets

$M = 0$;
for $t \leftarrow 1$ **to** T **do**
    preprocess image $\mathbf{y}_t$;
    estimate optical flow f_t using Lucas-Kanade method for first-order affine optic flow;
    extract targets using edge detector and Hough transform;
    if *there are* M_{new} *new targets* **then**
        for $m \leftarrow 1$ **to** M_{new} **do**
            generate N particles by sampling from a Gaussian distribution centered on (x, y, l, θ) of detection;
            $f_{t,m} = 0$;
        end
        $M = M + M_{new}$;
    end
    for $m \leftarrow 1$ **to** M **do**
        transform unweighted samples $\tilde{\mathbf{x}}_{m,t-1}^{(i)}$ using f_t;
        propose N new particles $\mathbf{x}_{m,t}^{(i)}$ by (3.5) ;
        compute the observation likelihood for $\mathbf{x}_{m,t}^{(i)}$ by (3.6);
        update the importance weights $w_{m,t}^{(i)}$ by (3.7);
        generate unweighted samples $\tilde{\mathbf{x}}_{m,t}^{(i)}$ by resampling $\mathbf{x}_{m,t}^{(i)}$ according to the importance weights $w_{m,t}^{(i)}$;
    end
    remove M_1 targets whose edge detection confidence is below a threshold;
    merge M_2 pairs of targets who overlap with each other;
    $M = M - M_1 - M_2$;
end

main system components are a method of estimating optical flow, an edge detector, and multiple particle filters.

This system is designed to be of practical value to an ROV operator. The design choices lead to several recommendations for how the system should be configured and what the operator can to do improve performance. Under heavy snow conditions, the operator should increase the time over which median filtering is performed. This, in turn, means the vehicle should be maneuvered more slowly.

The system is sensitive to the edge strength thresholds and the operator should have direct, real-time control over these settings in order to adapt as conditions change.

If the background conditions change, the background subtraction will become less effective and this will affect both the motion estimation and edge detection. Depending on operating conditions, the operator may be able to reset the background by capturing 2 to 3 seconds of video with no targets in sight. However, in many cases, the operator risks losing his or her bearings by maneuvering the camera to capture background footage.

There are several avenues of future research, including alternative colour spaces, feedback from vehicle sensors or controllers as input to motion estimation, automatic parameter tuning (including edge strength thresholds) and dynamic background subtraction. Another possibility that has not been discussed is to build a 3D object model based on edge locations and assumptions of rigid motion, and use that model to provide additional evidence for the location of faint edges. Finally, a GPU-based real-time implementation would allow the system to be used in practice.

Chapter 4

Results and Discussion

This chapter presents the results of running the proposed system on a video sequence captured by a ROV operating in an environment with marine snow. The goal of these tests is to evaluate the success of the system and identify overall trends in performance. In addition to reporting the total number of errors over the full dataset, we will examine the errors in individual video frames under different conditions in the video and different system settings.

Similar to many other computer vision systems, there is a trade-off between types of error: false positives can often be reduced at the expense of increasing false negatives. A common goal, and one that holds for this application, is to reduce the number of false negatives as far as possible while keeping the number of false positives at a "reasonable" level. A less ambiguous measure for a given dataset is the equal error rate, or point at which the numbers of false negatives and false positives are equal.

4.1 Dataset

The dataset consists of 50 seconds (1500 frames) of test video plus 3 seconds (90 frames) of background video [Gamroth, 2010]. The dataset includes several different test conditions and the marine snow varies from moderate to heavy. These conditions are representative of poor visibility conditions in this area. There are lighting artifacts including a gradual reduction in intensity towards the perimeter

of the image and a beam of light in the top centre.

The video segment begins with no visible objects of interest, followed by quick camera motion as targets come into view. There are several seconds of relatively slow motion followed by several seconds of faster motion, all while keeping the target object in the frame. Next, the vehicle's tether becomes visible; it is very high contrast and is located between the camera and the targets. The tether is so much brighter than the rest of the image that the camera settings were automatically adjusted, causing everything but the tether to become darker. While the tether is in view, the camera moves farther from the targets until they are barely visible in the background. Finally, the tether leaves the frame and the camera approaches the object of interest. In the final several seconds of video, the marine snow becomes very heavy. This dataset will allow the proposed system to be tested under a range of realistic conditions.

The proposed system relies on having several seconds of background video under the same lighting and camera conditions as the test video. In practical settings, this could be acquired by an ROV operator by turning the vehicle to point away from the targets. The purpose of the background video is to identify lens and camera artifacts so they can be removed during processing. In this case, the background video was recorded several seconds before the test sequence.

4.2 Ground Truth

A measure of ground truth is required in order to quantitatively evaluate the success of the proposed system. Ground truth is defined as the edges that an ROV operator would like to see highlighted. Some of the ground truth edges are difficult to see, and some narrow bars (with top and bottom edges) are marked only with a single edge. Although some of the ground truth edges are unlikely to be detected by this or other systems, they represent the benchmark of an ideal system against which the proposed system can be measured.

An annotation tool was developed to allow ground truth to be easily generated. The tool consists of a Graphical User Interface (GUI) that automatically loads the next video frame and allows the user to identify each edge by clicking the two endpoints. An annotation file is created with the pixel locations of all the ground

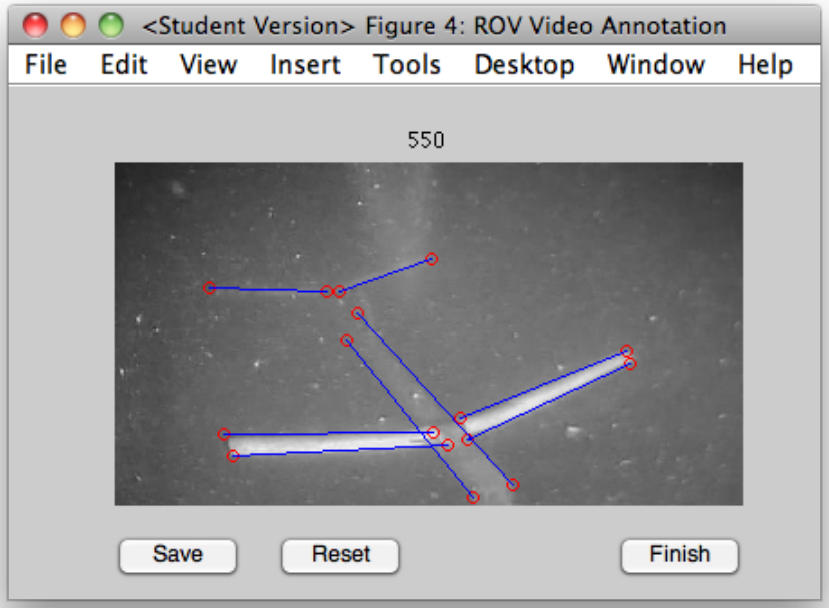


Figure 4.1: Screenshot of annotation tool. Ground truth edges (blue line segments) are created for every tenth video frame by selecting the end points (red circles).

truth edges for each image.

The user interface for this tool is shown in Figure 4.1. Each tenth frame was annotated (three annotated frames per second). There are 1383 ground truth edges over the full sequence. A subset of the ground truth annotations can be seen in Figure 4.4 and Figure 4.5. The black and blue lines are ground truth. The annotations were reviewed by the ROV operator who provided the video sequences.

4.3 Test Methodology

For each annotated image, the line segments identified by the system were compared to ground truth. Matching line segments are considered true positives, line segments that do not match ground truth are considered false positives, and ground

truth lines with no match are considered false negatives. The two numbers, false positives and false negatives, are reported for each ground truth image as a means of quantitatively evaluating the system.

The scoring for matches is binary; the result is either true or false without assigning a value to the quality of the match. The line similarity measure described in Section 3.2.5 is used to identify matches between ground truth and system results. The direction of the gradient is not considered and as a result the bottom and top of a thin white bar are considered matches for each other.

The results of several tests are presented. The first test is the performance of the proposed system excluding the SMCtracker (detector only). This characterizes how successful the system is at distinguishing the targets from the background by processing each frame independently. The performance of the detector is important, since edges that are never identified by the detector cannot be tracked by the full system. The second test is the performance of the full system (detector plus tracker). This test demonstrates the effects of gathering evidence over time; it also confirms that our method for motion estimation allows tracked edges to be transformed from one time step to the next with sufficient accuracy. By studying how these results change over the course of the dataset, we see how the system performs in the face of varying levels of marine snow, camera speeds, and contrast of target edges.

Finally, the effects of three parameters are evaluated. The parameters are: the size of the time window, t, over which temporal median is performed, the edge strength threshold for detection, γ_{detect}, and the threshold for retaining a tracked target, γ_{track}. The purpose of these tests is not to identify the single, optimal value of each parameter, but rather to demonstrate how system performance is affected and to identify a range of parameter values that produce reasonable results.

For the first set of tests, parameters were set to $t = 1$, $\gamma_{detect} = 0.02$ and $\gamma_{track} = 0.4$ and held constant throughout the test sequence. Depending on the operational goals of the ROV operator and the desired trade-off between false negatives and false positives, these may not be the optimal parameter settings but they do result in reasonable conditions for benchmarking system performance. When testing the effects of varying parameters, the parameter under consideration was adjusted while maintaining the other two at the values given above. It is possible that ad-

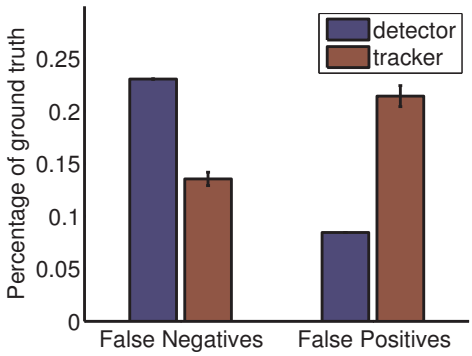


Figure 4.2: System performance as percentage of number of ground truth edges.

justing these parameter values during the video would improve results; this was not tested.

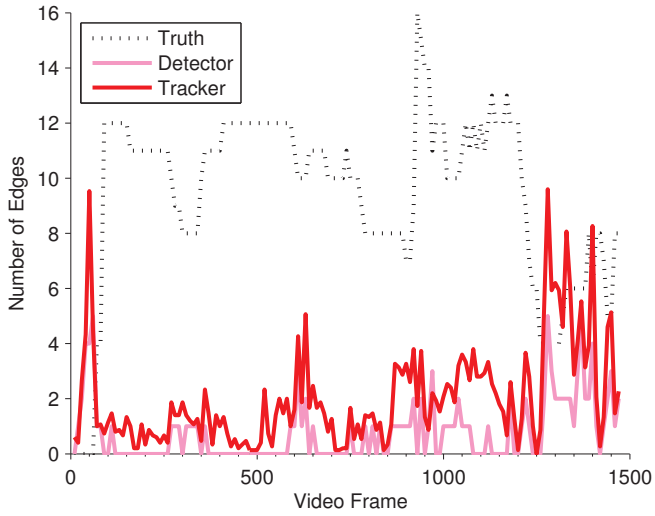
Each test was run 15 times. The system detector is deterministic and therefore gives the same results each time. The tracker is probabilistic and mean results are reported. The mean error rates over the full dataset are reported to $\pm$ 2% with $\geq$95% confidence.

4.4 Test Results

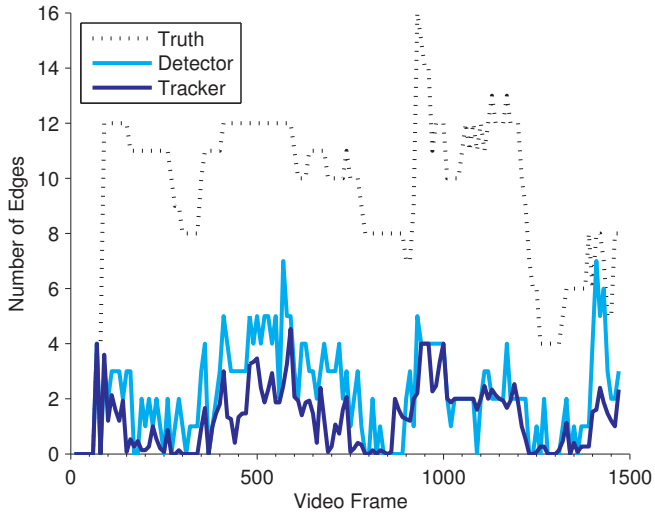
4.4.1 Target Detection

The detector alone returns approximately 23% false negatives and 9% false positives over the entire dataset. These results are given in Figure 4.2; they demonstrate that the system can successfully distinguish the targets from the background. Further understanding of system performance can be gained by studying the performance over time.

Figure 4.3 shows the number of false negatives and false positives for each ground truth image over the test dataset. The number of line segments is shown rather than percentages since the number of ground truth edges varies from frame to frame. From this graph we see that the system performs better for some video segments than others. For example, in the first 350 frames there is no more than

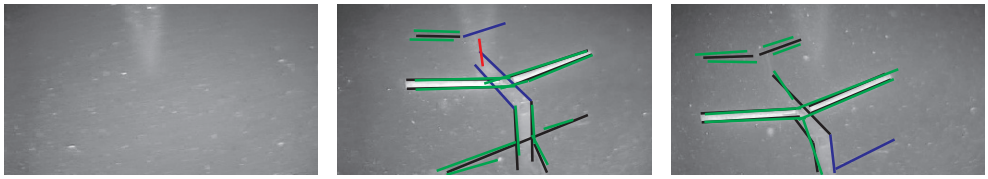


(a) False positives

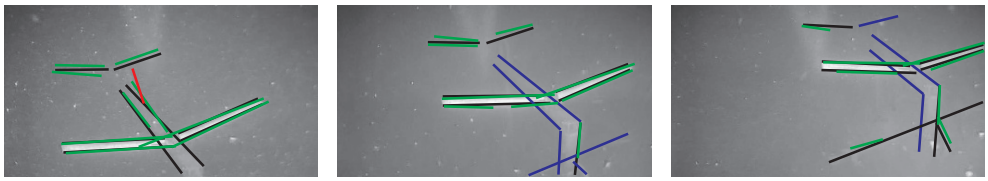


(b) False negatives

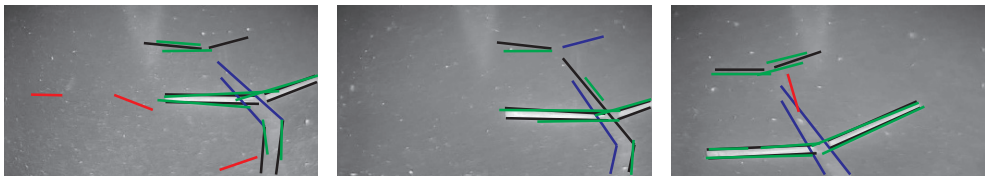
Figure 4.3: System performance every tenth frame. For comparison purposes, the number of ground truth edge in each frame is shown. Tracker error rates are reported to $\pm$ 0.4 with $\geq$95% confidence. (a) The detector has fewer false positives than the tracker and both have maxima at the start and end of the sequence. (b) The detector has more false negatives than the tracker. The spikes in false negatives are smaller than the spikes in false positives.



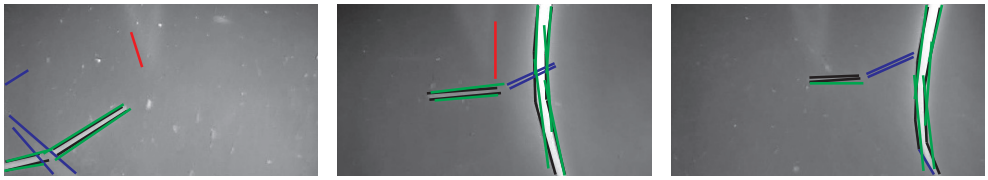
(a) Frame 0 **(b)** Frame 100 **(c)** Frame 200



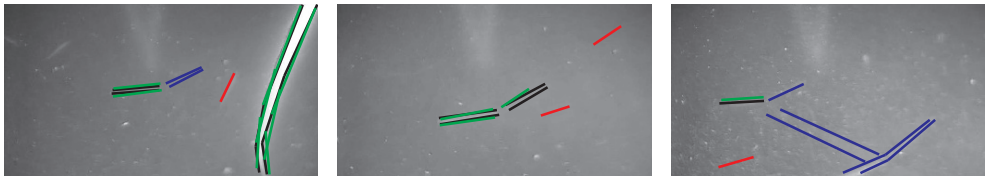
(d) Frame 300 **(e)** Frame 400 **(f)** Frame 500



(g) Frame 600 **(h)** Frame 700 **(i)** Frame 800



(j) Frame 900 **(k)** Frame 1000 **(l)** Frame 1100



(m) Frame 1200 **(n)** Frame 1300 **(o)** Frame 1400

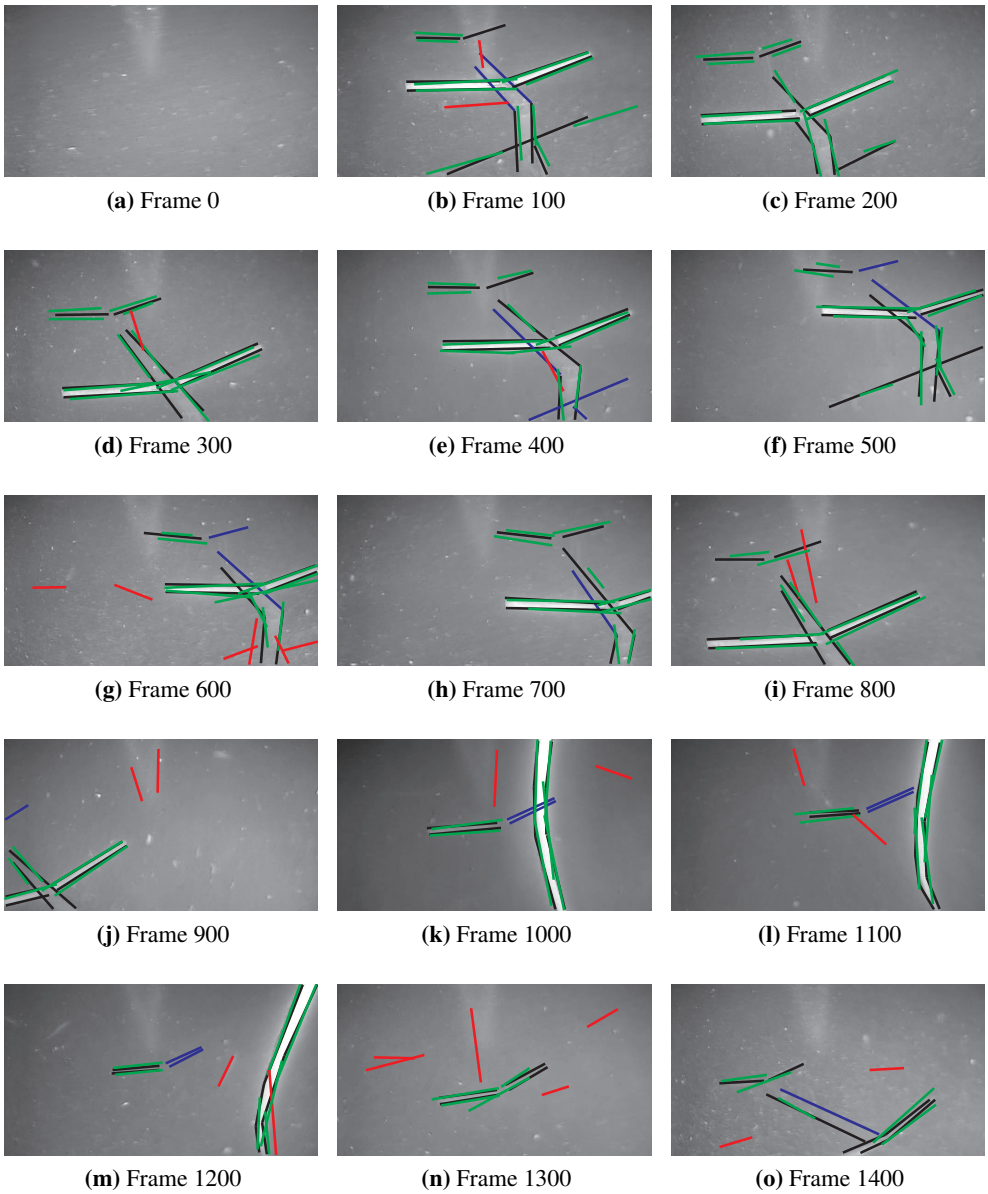
Figure 4.4: Results for detection only. Black and blue lines are ground truth annotations. Green and red lines are system results. Blue lines are false negatives. Red lines are false positives.

one false positive per frame and a low false negative rate. The false negatives increase after frame 350 while the number of false positives stays relatively low. There is a spike in both false positives and false negatives around frame 1400. These differences in performance reflect changing conditions in the video and can be better understood by examining individual video frames.

Figure 4.4 shows results for every hundredth frame. The detector results and ground truth are superimposed on the raw images, with blue lines representing false negatives and red lines representing false positives. As would be expected, the higher-contrast edges are detected more often. Lower-contrast edges that are detected are more likely to be inexact matches, with only a portion of the line segment identified.

We can make several observations about these results. In frames 100 through 500, the camera moves slowly, there is moderate marine snow and the targets are close to the camera. There are few false positives and the false negatives are on the lower-contrast targets. Some false positives, including those in frames 300 and 800 and the rightmost one in frame 600, appear to be aligned with portions of the real object that were not included in ground truth. The tether of the ROV becomes visible sometime between frame 900 and 1000 and moves quickly. Due to the automatic camera settings, the appearance of the bright tether reduces the contrast of the rest of the image. Frame 1000 shows a case where the background subtraction was not able to remove the lighting pattern at the top of the image. This is probably due to the change of contrast caused by the tether. Despite the heavy snow in frame 1400, there is only one false positive, although many true edges were also missed.

Overall, the detector is successful. As would be expected, the number of errors is directly affected by the magnitudes of the intensity gradients of the target edges and by the presence of high-contrast noise. The effects of these factors were not quantized because the dataset does not include video segments in which the factors are independently varied.



(a) Frame 0 **(b)** Frame 100 **(c)** Frame 200

(d) Frame 300 **(e)** Frame 400 **(f)** Frame 500

(g) Frame 600 **(h)** Frame 700 **(i)** Frame 800

(j) Frame 900 **(k)** Frame 1000 **(l)** Frame 1100

(m) Frame 1200 **(n)** Frame 1300 **(o)** Frame 1400

Figure 4.5: Results for full system with tracker. Black and blue lines are ground truth annotations. Green and red lines are system results. Blue lines are missed detections. Red lines are false positives. Figure 4.3 gives mean results over 15 tests and so will not exactly match the number of errors in these images.

4.4.2 Target Tracking

The addition of the tracker to the system decreased false negatives from 23% to 14% at the cost of increasing false positives to approximately 22%. This trend is expected since edges from previous time steps are carried forward by the tracker, resulting in better target identification but also allowing errors to propagate.

The overall percentages are given in Figure 4.2. Figure 4.3 shows the mean results over time. They follow the same general trends as the detector alone but have more severe spikes in numbers of false positives. The variance is not plotted in Figure 4.3 but is relatively constant and each point is reported to $\pm$ 0.4 with $\geq$95% confidence.

Figure 4.5 shows results for every 100th frame. Again, the higher contrast targets are more likely to be identified. Frames 900, 1100 and 1300 in Figure 4.5 give examples where the background subtraction failed to remove the lighting artifact in the top centre of the image and caused false positives.

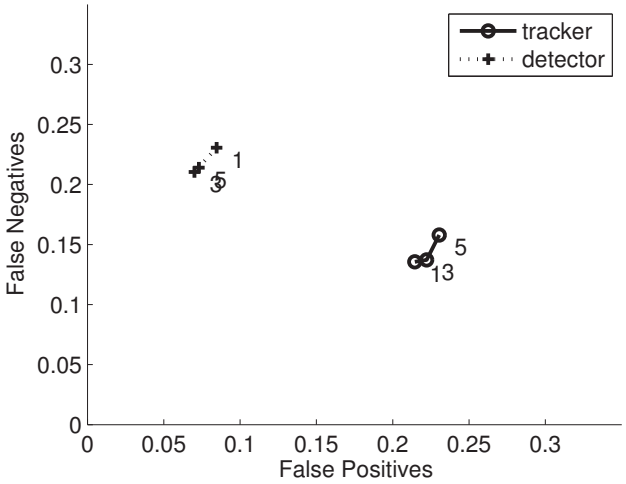
Many of the false positives in frame 1300 have similar orientations. This is also observed at other times in the sequence (e.g. Figure 4.9 (e) and Figure 4.11 (c)), although the dominant orientation changes over time. Although it is difficult to tell from the still images, the orientation of the false positives is aligned along the direction of camera motion. The faint edges causing the false positives are most likely caused by motion blur of marine snow.

4.4.3 Parameter Tuning

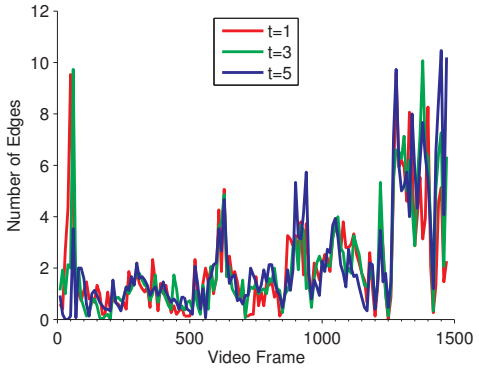
Three parameters are considered: the size of the time window t over which temporal median is performed, the detection threshold γ_{detect}, and the tracking threshold γ_{track}. Table 4.1 shows the parameter values that were tested. The values were selected to focus on a performance range that would be of practical interest to a human operator.

Time

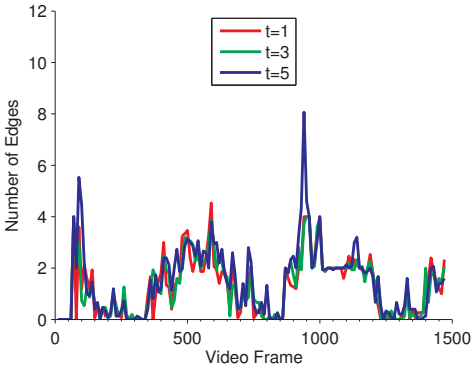
The parameter t was tested for values ranging from 1 (no temporal median filtering) to 5. The results, shown in Figure 4.6, indicate that the system is not very sensitive to this parameter over the range tested. The best overall system performance is



(a) Performance Curves

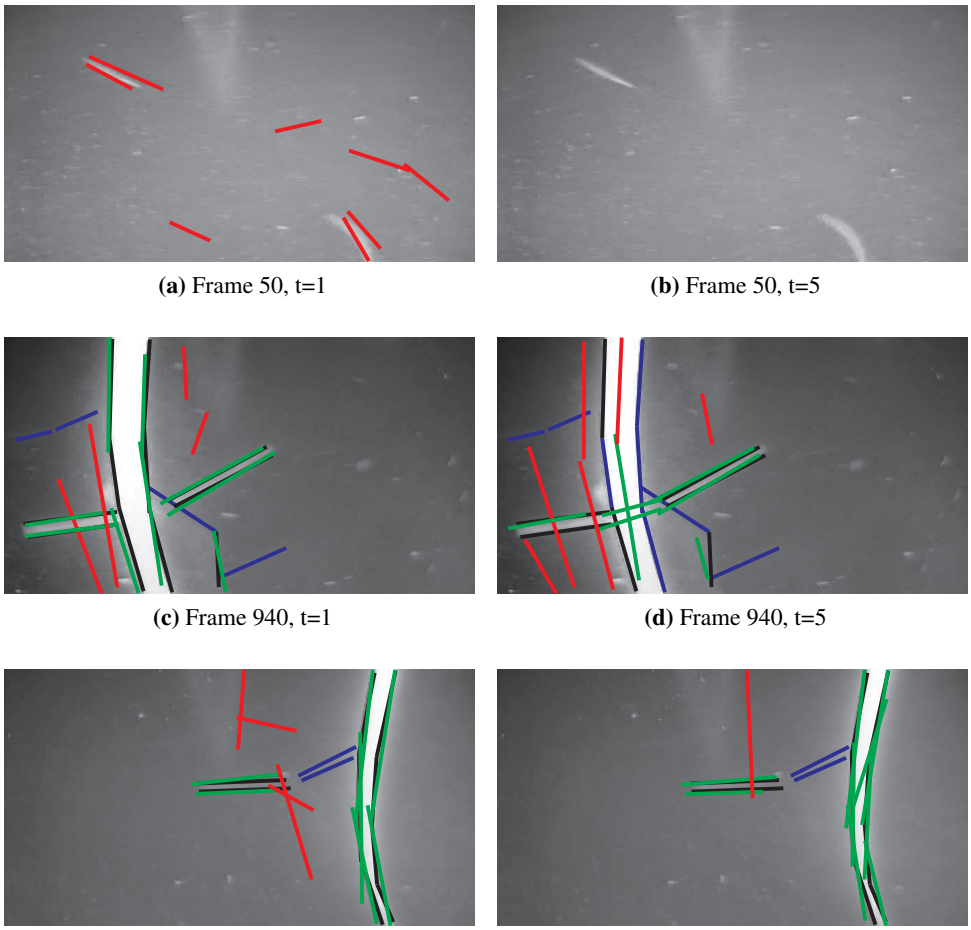


(b) False positives over time



(c) False negatives over time

Figure 4.6: System performance for different values of the parameter t. (a) Performance curves for tracker and detector. Tracker error rates are reported to $\pm$ 2% with $\geq$95% confidence. The corresponding value of t is shown for each datapoint. Overall performance is not greatly affected by varying t within this range. (b) False positives for three values of t. Although $t = 1$ shows the best overall performance, there are segments of the video during which a higher value of t has fewer false positives. (c) False negatives for three values of t. The difference in mean performance between parameters is less than one edge per frame.



(a) Frame 50, t=1

(b) Frame 50, t=5

(c) Frame 940, t=1

(d) Frame 940, t=5

(e) Frame 1090, t=1

(f) Frame 1090, t=5

Figure 4.7: Sample video frames for different values of the parameter t. Black and blue lines are ground truth annotations. Green and red lines are system results. Blue lines are missed detections. Red lines are false positives. (a-b) With heavey marine snow, $t = 5$ has no false positives while $t = 1$ has many. (c-d) The edges of the fast-moving tether are properly localized for $t = 1$ but not $t = 5$. (e-f) The tether is stationary and the edges are localized equally well by $t = 1$ and $t = 5$.

Table 4.1: Parameter values for testing

	Parameters		
Test	t	γ_{track}	γ_{detect}
Time	1	0.40	0.02
	3	0.40	0.02
	5	0.40	0.02
Detection	1	0.40	0.01
	1	0.40	0.015
	1	0.40	0.02
	1	0.40	0.025
	1	0.40	0.03
	1	0.40	0.06
Tracking	1	0.25	0.02
	1	0.325	0.02
	1	0.40	0.02
	1	0.475	0.02
	1	0.55	0.02
	1	0.70	0.02

achieved for $t = 1$ although the difference in results is not significant. Both the false positives and false negatives increase slightly with increasing t.

Unlike the tracker, performance of the detector is best when $t = 3$ and worst when $t = 1$. The most likely explanation is that the detector operates best when the temporal median filter is large enough to reduce noise but not so large as to lose information.

Figure 4.6 shows the results over time. This shows that no value of t is always best; a low value of t is beneficial at certain points in the video sequence and detrimental at others. Figure 4.7 gives a few examples of this. There is heavy marine snow and quick camera motion in frame 50 which results in more false positives for lower values of t. At frame 940 there is quick motion of the bright white tether. The edges of the tether are properly identified and localized for $t = 1$ but not for $t = 5$ because the leading edges of the tether are filtered out. Frame 1090 gives another example of a higher value of t outperforming a lower value.

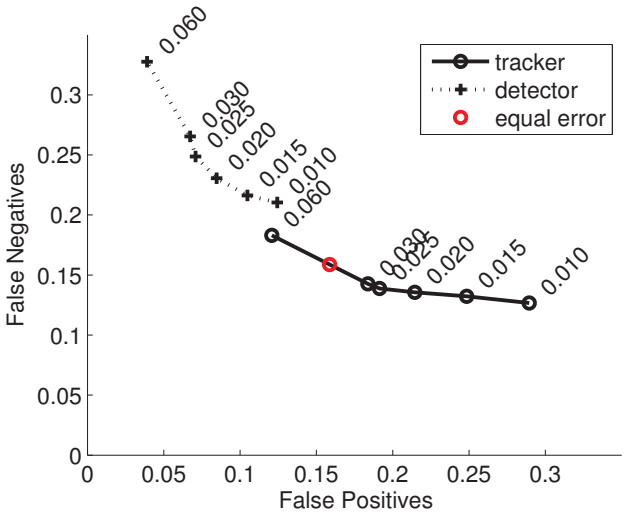
As suggested by the shape of the curve in Figure 4.6, adjusting the value of t is not necessarily a trade-off between false negatives and false positives. The equal error rate does not lie on the curve produced by these tests. Instead, the value of t can be chosen to minimize all errors for a given set of environmental conditions. A low value of t is suitable for fast motion and low snow while a higher value of t is best for slower motion and heavier snow.

Detection Threshold

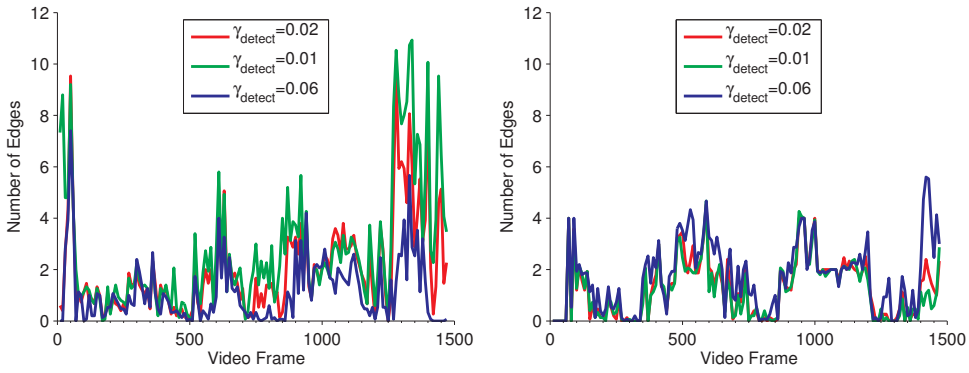
The parameter γ_{detect} was evaluated for values of between 0.01 and 0.06. The equal error rate for this dataset is approximately 0.159 and occurs when γ_{detect} is between 0.03 and 0.06. Decreasing γ_{detect} results in fewer false negatives and more false positives. As shown in Figure 4.8, this trend is observed under all conditions in the test video. This is to be expected since a lower threshold admits smaller intensity gradients, whether they are true edges or not. The results for the detector alone follow the same trend as the tracker results.

Figure 4.9 gives examples of system performance at several points in the video. In frame 630, the lower value of γ_{detect} results in more false positives but one less false negative. Frames 1420 and 1440 show system performance when the marine snow is heavy and the target object is faintly visible. These images clearly illustrate that the lower parameter setting results in fewer false negatives and more false positives. Frame 1440 gives another example of the improved quality of the matches for the lower parameter value. This is not because the lower parameter causes better detections, but because it allows the target to be detected sooner, giving the tracker more time to iteratively improve the quality of the fit.

If the goal is to reduce false negatives, γ_{detect} should be low, regardless of conditions in the video. The system's sensitivity to the parameter depends, of course, on the intensity gradients in the image. For many parts of the segment, changing γ_{detect} does not have a strong effect, indicating that the strengths of the true edges are outside the range tested. However, in cases like the spike around frame 1450, the edge strengths fall within the range of the tested thresholds and therefore the results are noticeably affected by the parameter setting.



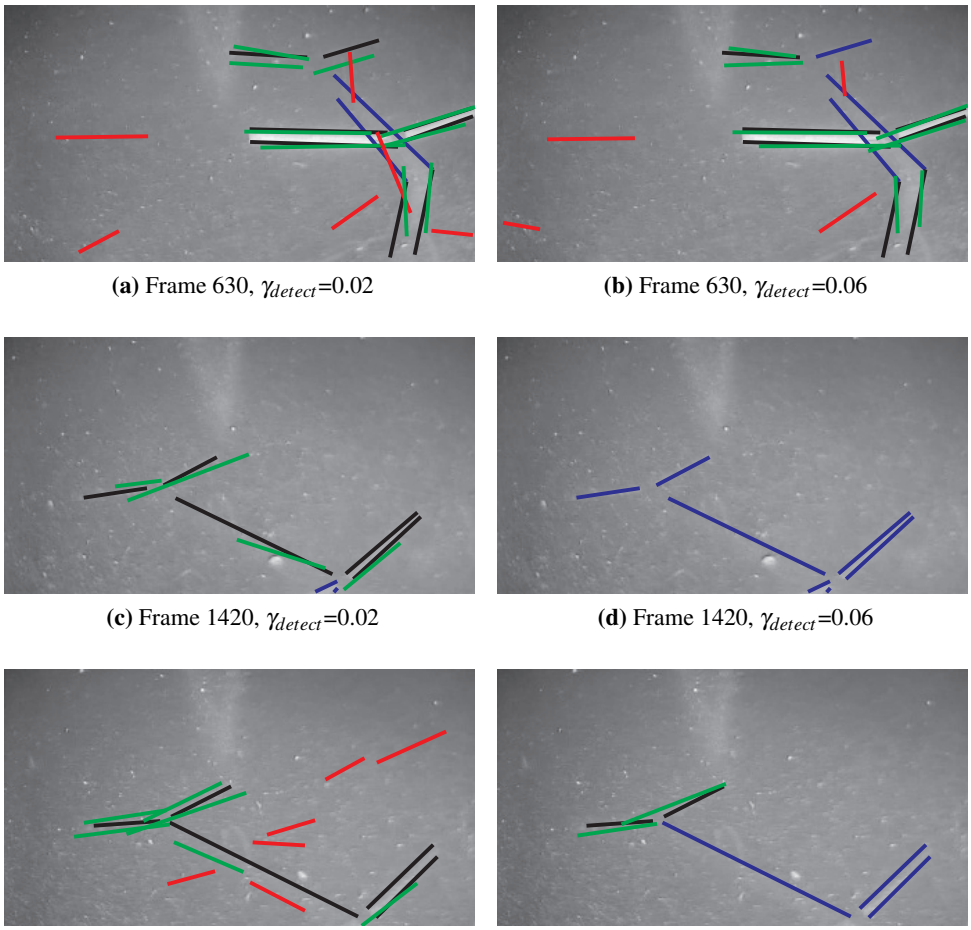
(a) Performance Curves



(b) False positives over time

(c) False negatives over time

Figure 4.8: System performance for different values of γ_{detect}. (a) Performance curves for tracker and detector, including the equal error point at 0.159. Error rates are reported to $\pm$ 2% with $\geq$95% confidence.. The corresponding value of γ_{detect} is shown for each datapoint. (b) False positives for three values of γ_{detect}. The lowest value of γ_{detect} results in the most false positives and vice versa. (c) False negatives for three values of γ_{detect}. The lowest value of γ_{detect} results in the fewest false positives and vice versa. Changing γ_{detect} has a smaller effect on false negatives than false positives.



(a) Frame 630, γ_{detect}=0.02

(b) Frame 630, γ_{detect}=0.06

(c) Frame 1420, γ_{detect}=0.02

(d) Frame 1420, γ_{detect}=0.06

(e) Frame 1440, γ_{detect}=0.02

(f) Frame 1440, γ_{detect}=0.06

Figure 4.9: Sample video frames for different values of γ_{detect}. Black and blue lines are ground truth annotations. Green and red lines are system results. Blue lines are false negatives. Red lines are false positives. (a-b) A local maximum in the number of false positives. (c-d) The target edges fall below the threshold of $\gamma_{detect} = 0.06$ and above $\gamma_{detect} = 0.02$. (e-d) Only two edges are above $\gamma_{detect} = 0.06$. All other target edges, as well as six noisy edges, are above $\gamma_{detect} = 0.02$

Tracking Threshold

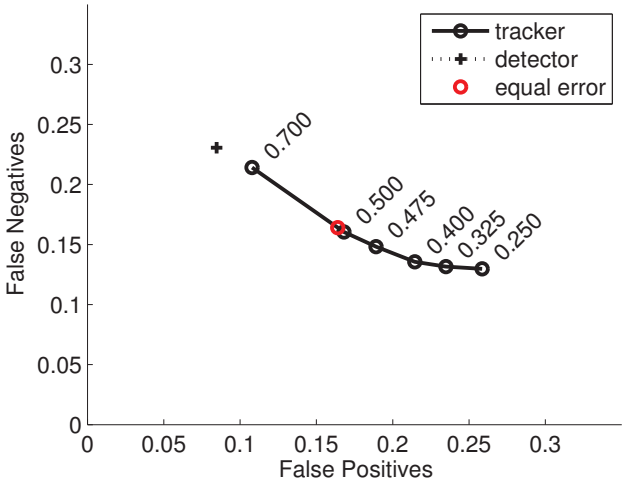
The parameter γ_{track} was evaluated for values between 0.25 and 0.7. The equal error rate for this dataset is 0.164 and occurs when γ_{detect} is approximately 0.5. Similar to the results for γ_{detect}, the smaller the value of γ_{track}, the lower the rate of false positives and the higher the rate of false negatives. This can be seen in Figure 4.10. In this case, the performance of the detector is not affected.

Figure 4.11 gives examples of system performance at several points in the video. In frame 240, the system has no false positives or false negatives for $\gamma_{track} = 0.4$ and two of the target line segments are too faint to be detected when γ_{track} is increased. By contrast, there are significant numbers of false positives in frame 1280 when $\gamma_{track} = 0.4$. As observed in the previous section, many of the false positives are aligned with the camera motion. In frame 1410, only one target edge is correctly tracked when $\gamma_{track} = 0.7$. Overall, the examples in Figure 4.11 show that an operator would likely wish to adjust the value of γ_{track} for different conditions to trade off false positives and false negatives.

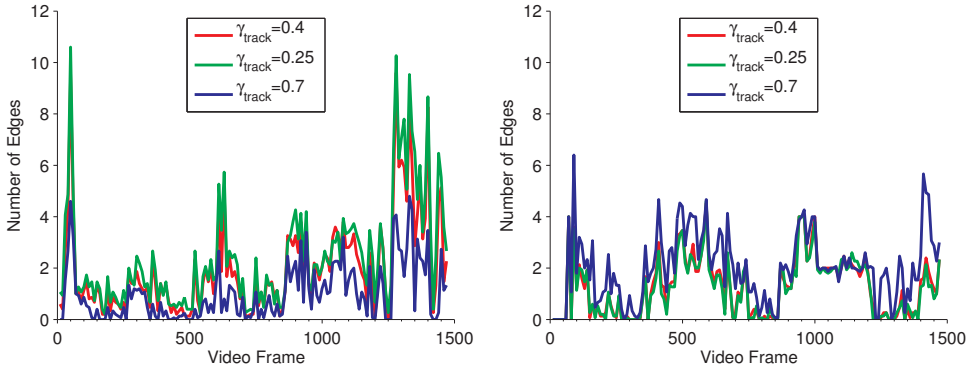
Discussion

This set of tests shows the effects of varying the three parameters t, γ_{detect} and γ_{track}. The optimum value of t is dependent on the conditions in the video and tests suggest that $t = 1$ is suitable for moderate snow conditions and faster motion while a higher value of t is appropriate for heavier snow and slower motion. There is a gradual and predictable change in the system performance as γ_{detect} and γ_{track} are varied. Increasing either of these parameters results in more false negatives and fewer false positives. The values can be chosen to create the appropriate trade-off between false negatives and false positives for a given task.

Note that the overall percentage of false positives could have been reduced in all cases by reporting the numbers for frames 100 - 1200 only. This indicates that the overall number can be misleading because it does not tell the whole story. However, the shape of the performance curves indicate trends that would hold even if the scales on the axes were shifted by the selection of a different video segment.



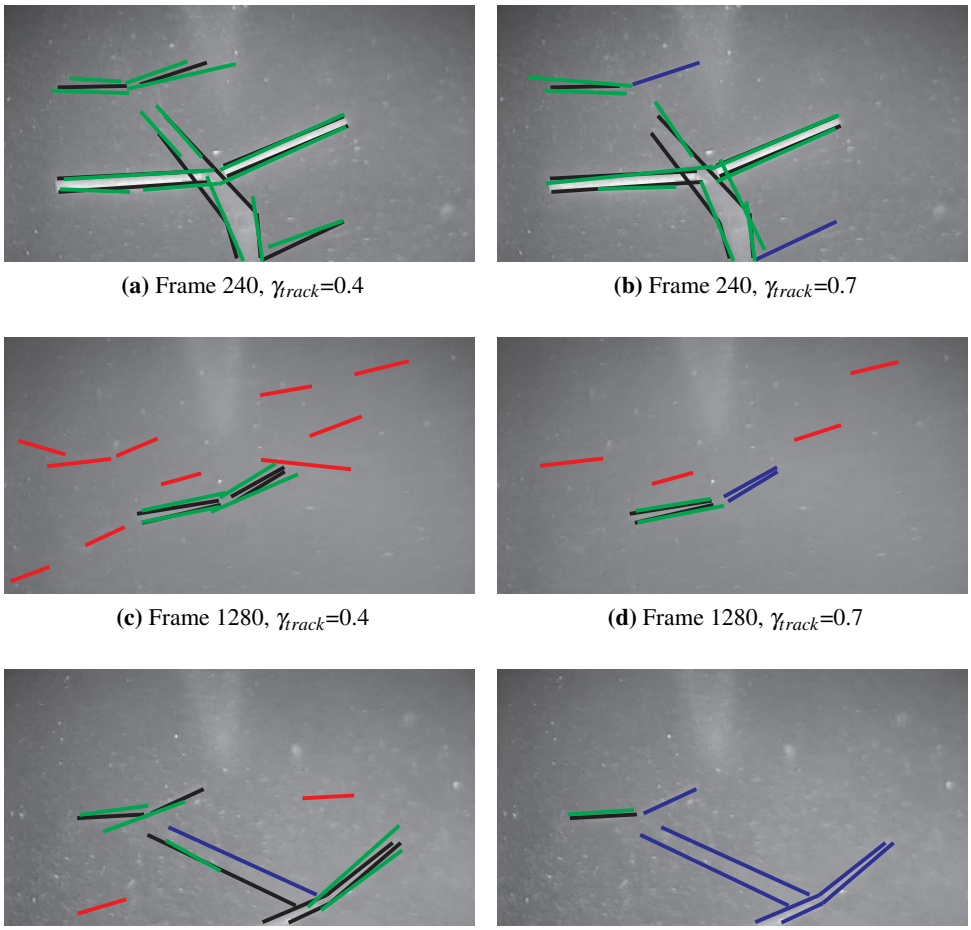
(a) Performance Curves



(b) False positives over time

(c) False negatives over time

Figure 4.10: System performance for different values of γ_{track}. (a) Performance curves for tracker and detector, including the equal error point at 0.164. Tracker error rates are reported to $\pm$ 2% with $\geq$95% confidence. The corresponding value of γ_{track} is shown for each datapoint. The detector is not affected by the value of γ_{track} and therefore appears as a single point. (b) False positives for three values of γ_{track}. The lowest value of γ_{track} results in the most false positives and vice versa. (c) False negatives for three values of γ_{track}. The lowest value of γ_{track} results in the fewest false negatives and vice versa.



(a) Frame 240, γ_{track}=0.4

(b) Frame 240, γ_{track}=0.7

(c) Frame 1280, γ_{track}=0.4

(d) Frame 1280, γ_{track}=0.7

(e) Frame 1410, γ_{track}=0.4

(f) Frame 1410, γ_{track}=0.7

Figure 4.11: Sample video frames for different values of γ_{track}. Black and blue lines are ground truth annotations. Green and red lines are system results. Blue lines are false negatives. Red lines are false positives. (a-b) A value of $\gamma_{track} = 0.4$ results in zero errors and increasing the threshold to $\gamma_{track} = 0.7$ creates only two false positives. (c-d) Motion blur causes false positives. (e-f) Most edges are strong enough to be detected but fall below the threshold of $\gamma_{track} = 0.7$.

4.5 Summary

The system was evaluated against a video recorded during an ROV mission in water with moderate to heavy marine snow. Overall, the system performs well under the test conditions.

The detector alone (without tracking) returns more false negatives than false positives under nearly all test conditions. The tracker carries the detector results forward over time and reduces the number of false negatives while increasing the false positives. For this application, the tracker results are preferable to the detector alone since fewer of the target edges are missed.

The equal error rate of the system with tracker is approximately 16% and the balance between false negatives and false positives can be adjusted by parameters. Lower false negative rates are achieved by decreasing one or both of the detection and tracking thresholds. For example, a false negative rate of 14% occurs with a false positive rate of 22%.

The time parameter can be increased for better performance in heavy marine snow but the vehicle must be maneuvered more slowly.

This testing was limited in the sense that the dataset was recorded for other purposes and did not include several interesting test cases. Future research would benefit from being able to set up such cases, such as recording the same scene with the camera moving at different speeds.

Chapter 5

Conclusions and Future Work

This project addresses the issue of automatic detection and tracking of man-made objects in underwater environments with poor visibility and marine snow. This is demonstrated using video captured by an ROV in Saanich Inlet, BC. The challenges of this project are to account for marine snow and poor visibility, estimate camera motion, detect low-contrast edges and track targets.

We review many related techniques and asses their suitability for this application. We demonstrate a prototype system that consists of consists of a temporal median filter, Canny edge detection, the Hough transform, Lucas-Kanade optical flow estimation, and SMC filters.

Overall, the system performs well under the conditions in the test video. The equal error rate is approximately 16%. Parameter tuning allows the false negative rate to be reduced at the expense of increasing the false positive rate and vice versa. Over the range of tested values, the false negatives were reduced to 13% at the expense of increasing false positives to 26%. Conversely, the false positives can be reduced to 17% at the expense of increasing false negatives to 16%.

The detector alone (without tracking) returns more false negatives than false positives under nearly all test conditions. The tracker carries the detector results forward over time and reduces the number of false negatives while increasing the false positives. For this application, the tracker results are preferable to the detector alone since fewer of the targets edges are missed.

Parameter tuning allows the system to be adjusted based on the conditions in

the video. The time parameter, t, should be reduced for low-snow conditions or fast motion and should be increased in heavy snow. When t is high, the ROV operator must move the camera more slowly to prevent targets from being missed or mis-localized. The two thresholding parameters, γ_{track} and γ_{detect}, allow the ROV operator to trade off false negatives and false positives. Low thresholds result in fewer false negatives and more false positives.

The system performance is affected by many factors. Poorest performance occurs under conditions of heavy marine show, low-contrast targets, and fast camera motion. Performance also suffers if the background conditions in the image change.

There are several avenues of future research, including alternative colour spaces, feedback from vehicle sensors or controllers as input to motion estimation, automatic parameter tuning (including edge strength thresholds), dynamic background subtraction, and 3D object modelling. Testing under controlled conditions would allow the effects of environmental conditions to be studied. Finally, a GPU-based real-time implementation would then allow the system to be used in practice and the system could allow real-time parameter tuning by a human operator.

Bibliography

E. H. Adelson and J. R. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America*, 2, 1985. → pages 20

P. Barnum, T. Kanade, and S. G. Narasimhan. Spatio-temporal frequency analysis for removing rain and snow from videos. In *Proceedings of the First International Workshop on Photometric Analysis for Computer Vision*, 2007. → pages 13

P. C. Barnum and S. Narasimhan. Analysis of rain and snow in frequency space. *International Journal of Computer Vision*, 86(2-3), 2009. → pages 13

Y. Bengio. Markovian models for sequential data. *Neural Computing Surveys*, 2: 129–162, 1999. → pages 20

J. F. Canny. A computational approach to edge detection. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume PAMI-8, pages 679–698, 1986. → pages 19

N. de Freitas. Particle filter resampling code. Academic website, 1998. URL http://www.cs.ubc.ca/~nando/software.php. Matlab Code. → pages 39

G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51:91–109, 2003. ISSN 0920-5691. 10.1023/A:1021669406132. → pages 18

A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice / Arnaud Doucet, Nando de Freitas, Neil Gordon, editors ; foreword by Adrian Smith*. Springer, New York ; London :, 2001. ISBN 0387951466 0387951466. → pages vii, 20, 23

R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communicatins of the ACM*, 15:11–15, January 1972. ISSN 0001-0782. doi:10.1145/361237.361242. → pages 19

S. Fazekas, T. Amiaz, D. Chetverikov, and N. Kiryati. Dynamic texture detection based on motion analysis. *International Journal of Computer Vision*, 82:48–63, 2009. ISSN 0920-5691. 10.1007/s11263-008-0184-y. → pages 18

D. Gabor. Theory of communication. part 1: The analysis of information. *Electrical Engineers - Part III: Radio and Communication Engineering, Journal of the Institution of*, 93(26):429 –441, november 1946. doi:10.1049/ji-3-2.1946.0074. → pages 33

C. Gamroth. Supporting material for master's thesis. Academic website, 2010. URL http://www.cs.ubc.ca/nest/lci/thesis/cgamroth. → pages 11, 28, 46

K. Garg and S. Nayar. Detection and removal of rain from videos. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2003. → pages 13

H. Hase, K. Miyake, and M. Yoneda. Real-time snowfall noise elimination. In *International Conference on image processing*, volume 2, 1999. → pages vii, 14

K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1956–1963. IEEE Computer Society, June 2009. doi:10.1109/CVPRW.2009.5206515. → pages 13

A. Holm and O. Holm. Method that improves human interpretation of color images with limited spectral content, 2008. United States Patent Application 20,080,304,695. → pages 17

P. Hough. Method and means for recognizing complex patterns, 1962. United States Patent 3,069,654. → pages 19

M. S. Langer and R. Mann. Optical snow. *International Journal of Computer Vision*, 55(1), 2003. → pages 14, 15

A. Lee, C. Yau, M. B. Giles, A. Doucet, and C. C. Holmes. On the utility of graphics cards to perform massively parallel simulation of advanced monte carlo methods. Technical report, Oxford University Department of Statistics, 2009. → pages 24, 43

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004. ISSN 0920-5691. doi:10.1023/B:VISI.0000029664.99615.94. 10.1023/B:VISI.0000029664.99615.94. → pages 38

W.-L. Lu, K. Okuma, and J. J. Little. Tracking and recognizing actions of multiple hockey players using the boosted particle filter. *Image and Vision Computing*, 27(1-2):189–205, 2009. ISSN 0262-8856. doi:10.1016/j.imavis.2008.02.008. → pages 24

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, pages 121 –130, 1981. → pages 15

Lyyn. Lyyn website, 2010. URL http://www.lyyn.com. corporate website. → pages 17

V. Mahadevan and N. Vasconcelos. Background subtraction in highly dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6. IEEE Computer Society, 2008. → pages 18

R. Mann and M. S. Langer. Estimating camera motion through a 3d cluttered scene. *Canadian Conference on Computer and Robot Vision*, 0:472–479, 2004. doi:10.1109/CCCRV.2004.1301486. → pages 15

O. Monga, R. Deriche, G. Malandain, and J.-P. Cocquerez. Recursive filtering and edge closing: two primary tools for 3d edge detection. In *European Conference on Computer Vision*. Springer Berlin / Heidelberg, 1989. → pages 20

S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. *International Journal of Computer Vision*, 48(3), 2002. → pages 12

S. G. Narasimhan and S. K. Nayar. Contrast restoration of weather degraded images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25: 713–724, 2003. ISSN 0162-8828. doi:10.1109/TPAMI.2003.1201821. → pages 12

J. Oakley and B. Satherley. Improving image quality in poor visibility conditions using a physical model for contrast degradation. *IEEE Transactions on Image Processing*, 7(2):167–179, Feb 1998. ISSN 1057-7149. doi:10.1109/83.660994. → pages 13

K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *European Conference on Computer Vision*, pages 28–39, 2004. → pages 24

S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, and K. Zuiderveld. Adaptive histogram

equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39(3):355 – 368, 1987. ISSN 0734-189X. doi:10.1016/S0734-189X(87)80186-X. → pages 17

Z. Qi and J. Cooperstock. Automated change detection in an undersea environment using a statistical background model. In *OCEANS 2007*, pages 1 –6, sept. 2007. doi:10.1109/OCEANS.2007.4449201. → pages 29

Y. Rzhanov, L. Linnett, and R. Forbes. Underwater video mosaicing for seabed mapping. In *International Conference on Image Processing*, volume 1, pages 224–227 vol.1, 2000. doi:10.1109/ICIP.2000.900935. → pages 17

J. Salvi, Y. Petillot, S. Thomas, and J. Aulinas. Visual SLAM for underwater vehicles using video velocity log and natural landmarks. In *OCEANS 2008*, pages 1 –6, 15-18 2008. doi:10.1109/OCEANS.2008.5151887. → pages 17

Y. Sheikh, O. Javed, and T. Kanade. Background subtraction for freely moving cameras. In *IEEE International Conference on Computer Vision*, pages 1219 –1225. IEEE Computer Society, September 2009. doi:10.1109/ICCV.2009.5459334. → pages 18

E. P. Simoncelli and W. T. Freeman. The steerable pyramid: a flexible architecture for multi-scale derivative computation. In *IEEE International conference on Image Processing*. IEEE Computer Society, 1995. → pages 18

S. Starik and M. Werman. Simulation of rain in videos. In *Texture Workshop, International Conference on Computer Vision*, 2003. → pages 13

S. Tan and J. Oakley. Physics-based approach to color image enhancement in poor visibility conditions. *Journal of the Optical Society of America*, 2001. → pages 13

Triton. Triton logging company website, 2010. URL http://www.tritonlogging.com/. corporate website. → pages 1

R. van der Merwe, A. Doucet, N. de Freitas, and E. Wan. The unscented particle filter. In *Advanced Neural Information Processing Systems*, 2000. → pages 24

J. Vermaak, A. Doucet, and P. Pérez. Maintaining multi-modality through mixture tracking. In *IEEE International Conference on Computer Vision*, page 1110, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1950-4. → pages 24

D. Walther, D. R. Edgington, and C. Koch. Detection and tracking of objects in underwater video. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 544–549, Los Alamitos, CA, USA, 2004. IEEE Computer Society. doi:10.1109/CVPR.2004.80. → pages 18

D. Young. Affine optic flow. Matlab Central, 2010. URL http://www.mathworks.com/matlabcentral/fileexchange/27093-affine-optic-flow. Matlab Code. → pages 15, 16

X. Zhang, H. Li, Y. Qi, W. Leow, and T. Ng. Rain removal in video by combining temporal and chromatic properties. In *IEEE International Conference on Multimedia and Expo*, 2006. → pages 13