University of Alberta

SHAPE BASED JOINT DETECTION AND TRACKING WITH ADAPTIVE MULTI-MOTION MODEL AND ITS APPLICATION IN LARGE LUMP DETECTION

by

Zhijie Wang

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

Spring 2012
Edmonton, Alberta

Dedicated to My Parents
Wang, Jiansheng and Xiao, Shouyan

Abstract

This thesis is motivated by a practical real application, Large Lump Detection (LLD), for which we provide a complete automatic system to detect large lumps in the oil sands mining surveillance videos. To this end, we propose a solution built around three main research components, each of which raises a specific issue, is formulated in a general way, and is tested on both the LLD problem and other similar applications.

The first issue is related to the detection of objects that undergo sudden changes in motion. We formulate this problem in a joint detection and tracking (JDT) framework using multiple motion models, where these models are predicted adaptively. The prediction exploits the correlation between motion models and object kinematic state. As a result, objects are detected more accurately when they change their motion.

The second issue concerns defining an appearance model which differentiates objects from background in an effective manner. We propose a novel shape based appearance model for kernel based trackers which typically model an object with a primitive geometric shape. As a result, by employing the proposed shape based appearance model, the kernel based trackers can improve their accuracy significantly.

The last issue aims to ensure an object detection which handles the steam occlusion. We propose a new steam detection method which directly feeds a discrete wavelet transformed image to an Adaboost classifier. In this way, the proposed method is not only accurate because a proper classifier is learned by Adaboost, but also computationally efficient because the feature extraction step is omitted.

The complete object detection solution for the LLD problem is obtained by combining the above three techniques. The proposed steam detection method en-

sures that objects of interest are not occluded, and then, the improved JDT method with the shape based appearance model performs the detection. Extensive experiments and encouraging results which demonstrate the effectiveness of the proposed solution to the large lump detection problem are provided.

Acknowledgements

I am deeply indebted to my supervisor Dr Hong Zhang for all his guidance and support during the course of this thesis. Dr Zhang has been a source of constant motivation through these more than five years. His unique style of supervision would bring out the best in me without making me feel overly stressed, and his tremendous insight on the subject always ensured that my research progressed in a fruitful direction. I will always cherish the time spent working with him.

I would like to thank Dr Nilanjan Ray, and Dr Martin Jagersand who were part of my supervisory committee and gave useful suggestions during my candidacy and final oral examination. Thanks are due to Dr Biao Huang, and Dr Zenian Li who were part of my examining committee. Their feedback was very useful and has signicantly contributed towards the improvement of my thesis.

I would like to acknowledge the support of Abhishek Srivastava who started off being my office mate and over the years evolved into a very close friend. We started working on our Ph.Ds together and he has been a constant support through the years especially when circumstances were tough. I would like to thank Dr Mohamed Ben Salah who helped improving my thesis. I would also like to thank Zhengwei Zhang, Hui Wang, Rajarshi Maiti, Bo Li, Weitao Li, Jichuan Shi, Li He and Jiemin Wang. I will always cherish their friendship and support. Finally, I dedicate this thesis to my parents. Without their love and support, this thesis would not have been possible.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Large Lump Detection

Oil sands mining is an important industry in Canada. Canada has oil sand deposits containing about 1.7 trillion barrels of bitumen in-place, comparable in magnitude to the world's total reserves of conventional crude oil [106]. To separate oil from sand, the very first step in the current methods of production is to reduce the size of oilsand fragments that are excavated in open-pit mines. This is done by feeding oilsand fragments into a crusher as shown in Fig. 1.1. Since the mining operation takes place throughout the year including the cold winter in northern Canada, frozen large lumps, the size of several refrigerators (as shown in Fig. 1.2), can form and jam the crusher. The production downtime caused by these jamming events can results in huge economic losses. Therefore, an automatic solution is extremely desirable to detect the presence of such large lumps in the feed to crushers so that precautionary operational procedures could be taken to minimize the risk of jamming. In this thesis, we propose a computer vision based solution to this problem. This solution has two practical advantages. First, there is no need for additional facilities to be mounted because the real time surveillance video monitoring the worksite is readily available. Second, the current production process is not interfered by the computer vision based solution.

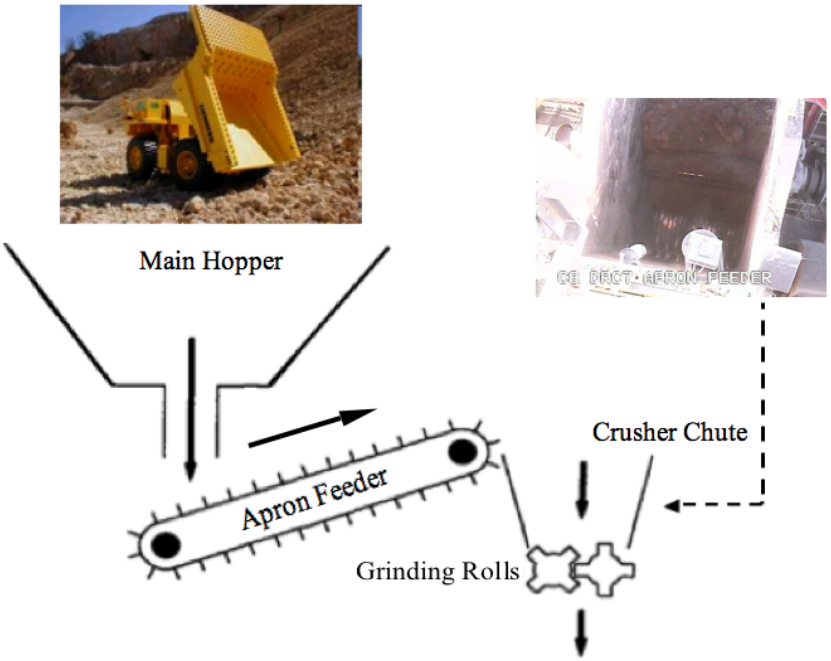


Figure 1.1: Illustration of the first processing step in the oil sands mining production.

1.1.1 JDT Based Object Detection

To achieve the objective announced before, this LLD problem is formulated as an object detection problem in this thesis. In the computer vision framework, an object detection method determines whether the object of interest is present in the input image or not and, if so, estimates the gross location of the object or the exact object boundaries. In this thesis, joint detection and tracking (JDT) is adopted as the framework within which we are going to propose our object detection method for

Figure 1.2: A large lump on an apron feeder before it falls into a crusher.

the large lump detection problem. JDT is a Bayesian recursive estimator with a hybrid object state,

$$p(X_t|Z^t) \propto p(Z_t|X_t)p(X_t|Z^{t-1}). \tag{1.1}$$

X_t contains both discrete and continuous variables, describing the current number of present objects and the objects' kinematic states (such as location) respectively. JDT recursively estimates X_t given all the observations up to time t, Z^t, so that the current number of object can be determined by this estimated hybrid object state. In other words, the object detection and tracking are conducted jointly in the same framework. In this way, the detection of new targets entering the scene, i.e., tracking initialization is embedded in the JDT framework without relying on an external object detection algorithm. As a result, JDT can handle new objects appearing and old objects dying from time to time, i.e., fulfill the detection mechanism as a robust object detection method.

Indeed, most of the object detection methods introduced in the literature are not suitable for the application at hand. For instance, background subtraction [47, 99, 36, 88, 68, 107], one of the most popular object detection techniques, builds a representation of a scene or a *background model* and then computes the deviation between each subsequent frame and this model. Any significant deviation corresponds necessarily to the moving objects. However, one basic assumption within background substraction methods is that the background should remain static during the whole tracking process. Therefore, this does not stand in problems with a constantly moving or varying background such as the application at hand. Another popular class of object detection methods utilizes supervised learning methods to learn different object views from a set of examples, and then generates a function that maps object features to "object" or "non-object" labels. This class of methods includes, among others, Adaptive Boosting (Adaboost) and support vector machine (SVM). They have been successfully employed in many applications such as pedestrian and face detection [96, 72]. Although the supervised learning methods can be directly applied to the large lump detection problem, however a much better performance can be obtained by combining them with joint detection and tracking methods. They are more beneficial in computing, as a first step, a soft detection

decision map of how likely an object is present. After that, this likelihood function can be incorporated into the JDT method as an appearance model. By doing so, we do not limit detection to the frame at hand as supervised learning methods do. Instead, decisions are taken based on the information gathered over time, i.e., during all the past tracking process. These claims defending the choice of the JDT framework are illustrated with experiments shown in Fig. 1.3 with an example of the LLD problem. Figs 1.3(b) and 1.3(e) show two images of the likelihood of having a lump centered at a certain location. These likelihood images are computed from the two original images shown in Figs 1.3(a) and 1.3(d) using a supervised learning method. Based on the likelihood maps, it is more likely that a lump is present in Fig. 1.3(a) than in Fig. 1.3(d) which is incorrect. However, by combining the evidence computed by the same supervised learning method from all previous frames, a JDT method provides the correct decision. Specifically, based on the combined evidence maps (the posterior probability images estimated by JDT) in Figs 1.3(c) and 1.3(f), it is more likely that the candidate object in Fig. 1.3(d) is a lump than the one in Fig. 1.3(a). This single example shows that JDT based methods can make reliable decisions based on the accumulated evidence during the tracking process.

1.1.2 LLD System

Once the JDT framework is chosen as the basic object detection method for the LLD problem, we build the complete LLD solution around it as shown in Fig. 1.4. This figure shows the flowchart of reaching a detection decision given an input image. A region of interest (ROI) is first specified in the input image. Then a preprocessing step is conducted to detect whether the input image is occluded by steam or not. In the case of steam occlusion, the input image will be labeled as a steam image and discarded without further processing. Otherwise, the input image will be fed to the JDT method for large lump detection. If any large lump is detected, an alarm will be triggered.

The complete LLD solution described above contains three major important components, a prediction model (our first contribution described in Chapter 2) and an appearance model (our second contribution described in Chapter 3) in the JDT

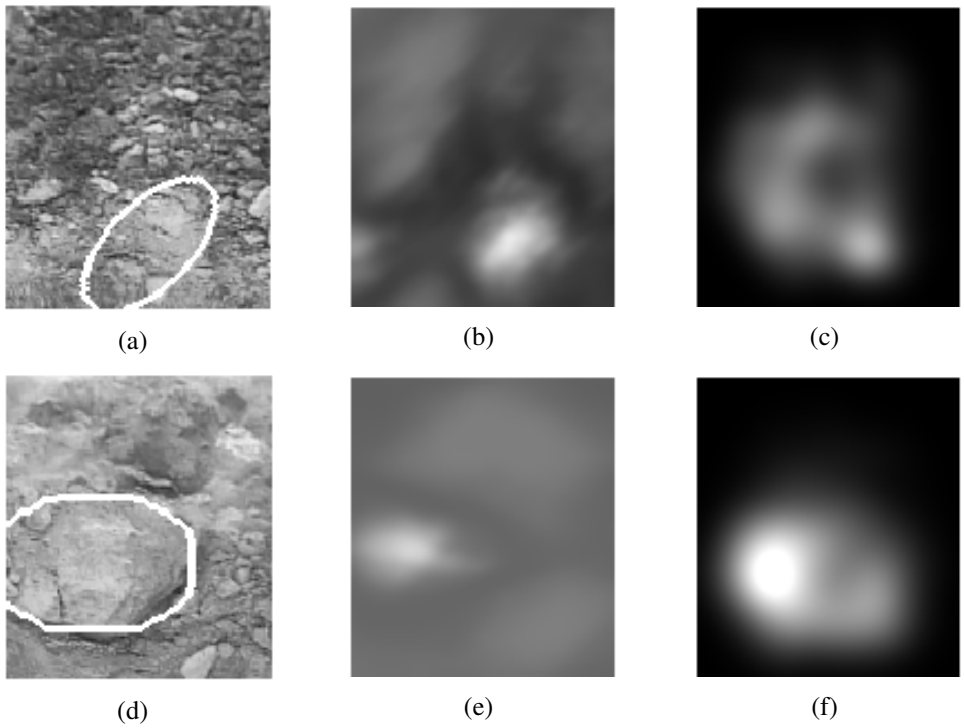


Figure 1.3: (a) An image without any lump, (b) likelihood image of the single frame in (a), (c) posterior probability density estimated by a JDT method until the frame in (a), (d) an image with a lump, (e) likelihood image of the single frame in (d), (f) posterior probability density estimated by a JDT method until the frame in (d).

framework, and a steam detection component (our third contribution described in Chapter 4). The prediction model and appearance model components correspond to the JDT block shown in Fig. 1.4. This JDT block is the core of the LLD solution, since it is directly responsible for detecting the large lumps. In this block, a Bayesian recursive estimator recursively estimates the object state X_t given all the previous observations $Z^t = \{Z_1, ...Z_t\}$ following

$$p(X_t|Z^t) \propto p(Z_t|X_t) \int p(X_t|X_{t-1}) p(X_{t-1}|Z^{t-1}) dX_{t-1}. \tag{1.2}$$

The prediction model and appearance model components correspond respectively to the two terms $p(X_t|X_{t-1})$ and $p(Z_t|X_t)$ in this framework. The prediction model $p(X_t|X_{t-1})$ predicts the object state based on the object motion models. We formulate it novelly with adaptive multiple motion models in Chapter 2 to detect objects that may undergo sudden motion changes. With the predicted object state, the appearance model $p(Z_t|X_t)$ then updates it based on the observation. We thus

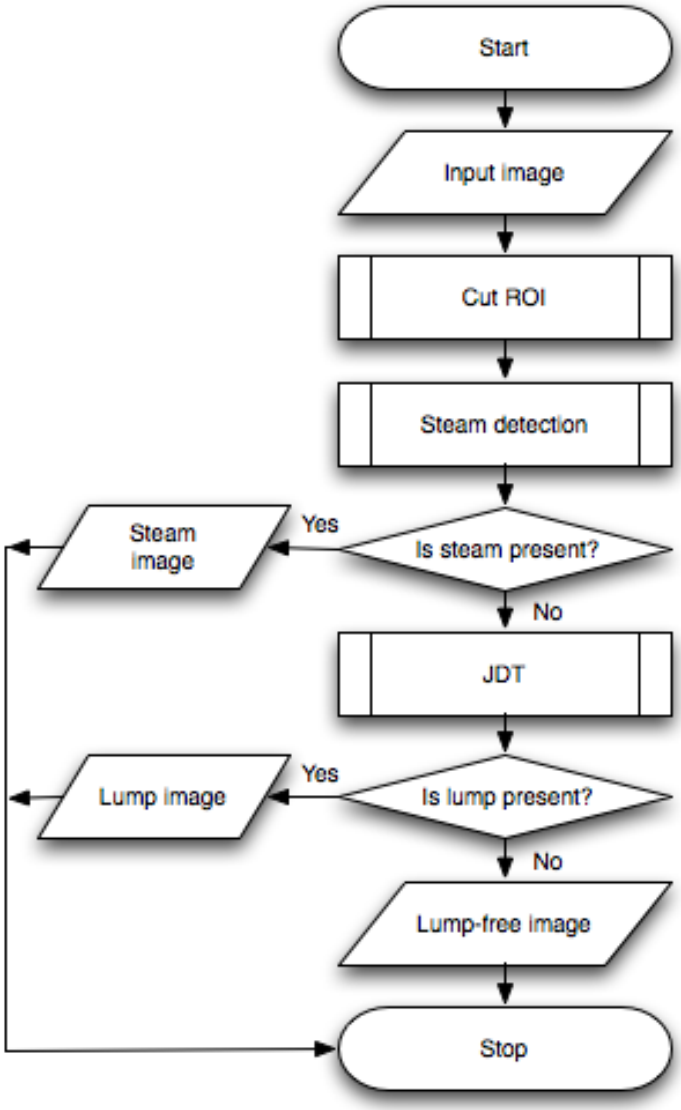


Figure 1.4: The proposed solution for large lump detection.

propose a novel shape based appearance model in Chapter 3 to incorporate shape information into tracking/detection. These two components along with each other dominantly determine the performance of the JDT framework, i.e., the JDT block in the LLD solution, and in turn the large lump detection performance. Besides these two components, another important component in this LLD solution corresponds to the steam detection block in Fig. 1.4. This steam detection component is important and necessary, because steam happens frequently in the LLD application during the wintertime. Ignoring this problem may directly result in the failure of the large

lump detection. Therefore, we propose an accurate and efficient steam detection method in Chapter 4 to prevent steam from interfering the large lump detection. To summarize, this thesis focuses on investigating the three critical problems briefly mentioned above, and solving these problems completes the previously discussed three components in the LLD solution. Although these three problems are raised in the LLD application, they are not limited to it. In the following, we will explain each one of the components in detail.

1.2 Motivated Research Problems

As previously mentioned, three general questions related to the three components in the system have to be investigated in order to provide a complete LLD solution.

The first issue is related to the multiple motion models that objects to be detected may undergo. In other words, objects in the real world may experience different motion models at different times and may switch arbitrarily from a motion model to another. However, the existing JDT methods assume a single motion model for the objects of interest. Therefore, they generally fail to keep track of the objects when they change their motion models. This question is relevant in our LLD application because the lumps move in more than one motion pattern on the feed to the crusher. For example, a lump moves on the belt in a slow acceleration model due to the perspective distortion (as shown by the two consecutive frames in Figs 1.5(a) and 1.5(b)), then it moves in a fast acceleration model due to gravity when it reaches the end of the belt and falls down (as shown by the two consecutive frames in Figs 1.5(b) and 1.5(c)). This phenomenon happens not only in the LLD problem but also in many other applications. For instance, soccer players may experience models of acceleration, constant velocity, and so on. In order to make a JDT method detect these kind of objects accurately, the ability of handling motion model changes has to be incorporated into the JDT methods. More precisely, an adaptive way of predicting changes of the motion model should be implemented in order to allow the tracker switch between motion models smoothly and accurately.

The second issue is related to the object appearance model embedded in the JDT

(a) (b) (c)

Figure 1.5: Three consecutive frames from the LLD application.

tracker which generally plays a critical role in object detection performance. The research around this specific question is mainly motivated by the difficulty encountered in defining a proper appearance model for the LLD problem which allows differentiating lumps from dirt because of their appearance similarity (refer to Fig. 1.6). Defining an appearance model is more closely related to object representation. For the sake of computational efficiency, an object can be represented by a primitive geometric kernel rather than its actual contour. The tracking methods employing this kind of object representation are typically called *kernel-based trackers* [102]. The JDT based tracking method adopted in this thesis for the LLD problem belongs to the class of kernel based trackers as we track the lump by a primitive elliptical kernel in order to reach a real time performance. Owing to such concise object representation, most of the existing appearance models in kernel based trackers utilize the textural information within the kernel only. Interestingly, shape information of a general form has never been fully exploited in kernel tracking. Therefore, a method which utilizes shape information in the appearance model of kernel based trackers is obviously needed to improve the ability to distinguish objects of interest from background.

(a) (b)

Figure 1.6: Two example images showing the similarity between the appearance of lumps and dirt.

The last issue is related to the occlusion that may intervene in some applications where detection is the main purpose. Occlusion happens in various forms. For example, it may be caused by the object itself or by other objects in the scene. In this specific research, we specifically investigate the occlusion resulting from the presence of steam, or similar objects in terms of appearance. This happens quite frequently in the LLD problem especially in winter and occludes the monitoring cameras used in the worksite (refer to Fig. 1.7). In fact, detecting whether the objects are occluded by steam -or similar objects- is an open and interesting problem which needs to be dealt with in many applications. For instance, various methods have been proposed to detect steam and smoke based on chromaticity, shape, motion, and energy information. Among these, signal processing methods based on energy information are the most suitable to the LLD problem although they do not really ensure a good compromise between accuracy and computational efficiency. Therefore, an accurate steam detection method with a real time performance is highly needed for the application at hand, the LLD problem.

(a) (b)

Figure 1.7: Two example images occluded by steam.

1.3 Contributions

In this thesis, we propose a complete automatic object detection system for LLD problem by addressing the three issues listed above. This system is based on a preprocessing step, the steam detection component, which ensures that the input images are steam-free in order to activate the following detection process. The JDT method, then, carries out the object detection task aided by two proposed components, the adaptive multi-motion model and the shape aided appearance model.

These major components summarize the thesis contributions and are listed in the following.

- **Adaptive multi-motion model**: We propose an adaptive multi-motion model formulated within the JDT framework to address the question of detecting objects which undergo multiple motion models mentioned earlier as the first issue. We formulate, in the same framework, both detection of varying number of objects and handling their sudden motion changes, extending what has been generally done in the literature where each problem was tackled separately. Prediction of motion model changes is supported by the introduction of a new prediction function which embeds the correlation information between the motion models and the object kinematic state. The advantages of this method are tested using, in addition to the LLD problem which motivated this thesis, other general applications in order to demonstrate the flexibility of the method. In all the considered experiments, the objects of interest change their motion models during the tracking process.

- **Shape based appearance model**: We propose a new way to incorporate shape knowledge into the appearance model of kernel based trackers in order to improve the object detection performance and address the second issue listed in the previous section. In the existing methods, the appearance models are based on textural information within the kernel boundary or the gradient information along the kernel boundary. Instead, the proposed appearance model provides a new way to utilize the prior shape information for the benefit of kernel based trackers. We formulated the appearance model in a general way which is flexible enough to bear with various shape contraints. Depending on the considered shape constraints, we addressed different real applications including the LLD problem. A series of comparative and extensive quantitative studies have shown the effectiveness of the method.

- **DWT & Adaboost based steam detection**: We propose also a steam detection component to address the third issue related to lumps occlusion resulting from steam. The proposed method directly feeds a discrete wavelet

transformed image to an Adaboost classifier which discards the need for the feature extraction step, always necessary in the existing methods. By doing so, the proposed method realizes the best compromise between accuracy and efficiency among the competing methods. Indeed, because a proper classifier is learned by Adaboost, the method reaches accurate results. In addition, it ensures a very low computational time because the cumbersome step, feature extraction, is discarded.

- **Large lump detection**: Besides the above three research contributions, a practical contribution is the whole proposed system for the large lump detection problem in oil sands mining. The proposed system automatically detects large lumps so that the risk of jamming crushers can be reduced and significant economic loss may be prevented.

1.4 Organization

In this thesis, three novel techniques are proposed to address some critical problems related to the LLD application: detect objects that undergo sudden changes in motion, integrate shape knowledge into the appearance model of kernel based trackers, and detect steam both accurately and efficiently. These three components are the basis of our solution for the large lump detection problem. For this reason, we organize the thesis as follows:

- In Chapter 2, we introduce the improved JDT method that detects objects experiencing multiple motion models. This chapter mainly improves the prediction model in the Bayesian recursive estimator in order to predict the object state accurately. Along with the novel appearance model proposed in the next chapter, accurate posterior object state can be finally estimated. In this way, the performance of the JDT step or block in the LLD system can be improved, and in turn the large lump detection performance can be improved. Specifically, in this chapter we first review related works to the JDT framework which investigate multiple motion models in tracking and models change predictors. Then, we detail the proposed formulation to bear with

multiple interacting model sets and embed the correlation information between motion models and object kinematic state. These contributions are supported with several general experiments including the LLD application.

- In Chapter 3, we present a novel technique to incorporate shape knowledge into the appearance model of kernel based trackers. The resultant shape based appearance model is a general appearance model for the Bayesian recursive estimator. It can update the object state predicted by the proposed prediction model in Chapter 2 based on the new observation, to obtain the posterior object state accurately. With the novel prediction model in the previous chapter and the novel appearance model in this chapter, the performance of the JDT method in the LLD system can be obviously enforced. Specifically in this chapter, after reviewing silhouette based trackers using shape knowledge and the competing knowledge based kernel trackers, we explain how the shape cue is defined. A general formulation embedding various shape constraints in the appearance model of the proposed kernel tracker is then stated. A large number of experiments illustrating the superiority of the proposed component in terms of accuracy and computational efficiency are carried out using various datasets.

- In Chapter 4, we provide a novel steam detection method based on DWT and Adaboost. This steam detection method mainly functions as a preprocessing step before the JDT step in the LLD system. It filters out those input images occluded by steam which happens frequently in the LLD application during the winter time. Without this step, the following JDT step will be severely interfered even though with the new techniques proposed in Chapters 2 and 3. As a result, the LLD performance will be affected due to the false positives resulting from steam. Specifically in this chapter, after an exhaustive description of the existing steam detection methods, the new steam detector is presented with focus on its theoretical advantages vis-a-vis the limitations of the competing techniques. A set of experiments first demonstrate that the proposed method outperforms the competing methods in terms of both accu-

racy and efficiency. Then, additional tests are also conducted on two smoke data sets in order to show that the proposed method also works in other similar contexts. In addition, an analysis is conducted to show the contribution of each one of the components of the new method.

- In Chapter 5, we integrate the three components proposed in the previous chapters, and provide the complete large lump detection system. This system is then applied to the online LLD videos which is the main motivation of this thesis. Evaluations on two types of data sets are conducted. One of them demonstrates the effectiveness and robustness of the provided system in terms of precision and recall. The other demonstrates the practical effect of the system in real scenarios which actually gave rise to crusher jamming and work stoppage in the past. Very encouraging results have been obtained.

- In Chapter 6, we conclude the thesis by summarizing its most important contributions. Some recommendations and promising future works are presented at the end.

Chapter 2

Joint Detection and Tracking Using Adaptively Multiple Motion Models

As stated in the introductory part of this thesis, the first component of the proposed object detection method for large lump detection deals with detection of objects having multiple motion models. Although motivated for this specific problem, the proposed component in this chapter is formulated in a general way. Indeed, the core idea behind the proposed solution is not limited to the targeted application and, as a result, can be easily applied to other contexts. We propose a JDT-based adaptive multi-motion model in order to accurately detect moving objects taking into account possible changing motion models. The proposed technique differs from the existing JDT-based methods mainly in two ways. First we express the solution in the JDT framework via a formulation in the *multiple* motion model setting. Second, we introduce a new motion model prediction function which exploits the correlation between the motion models and object kinematic state.

2.1 Introduction

In object tracking and detection, two problems that attract researchers' attention are (1) variable motion models of the moving objects, and (2) varying number of moving objects. In this work, we tackle both problems, i.e., we tackle the problem of detecting varying number of objects undergoing various motion models. Each of these two problems has been generally posed in the literature as a hybrid state estimation problem, that is, estimation of a partially observed stochastic process

with discrete- and continuous-valued states [63]. Among all the hybrid state estimation schemes, the interacting multiple model (IMM) estimator [8, 4] is the most cost-effective and has been widely adopted. IMM has the ability to estimate the state of a dynamic system with several behavior modes which can "switch" from one to another [63]. Such a system can be used to model a variety of problems with multiple behavior modes. In fact, IMM has been employed by many works to tackle separately each of the two previously mentioned problems: variable motion models and varying number of objects.

Concerning the first problem, a discrete state variable which indexes motion models along with a continuous state variable consisting of kinematic components have been employed by many IMM estimators to track objects that may change their motion models. For example, IMM is used with Kalman filters for describing maneuvering target dynamics, such as IMM-EKF, IMM-UKF, etc [1, 79]. To handle the problems with nonlinear dynamics and measurements, IMM is also combined with particle filters (PF) in [45, 20, 64] to track maneuvering targets. One potential problem with the basic particle filters is that the number of particles corresponding to a specific mode is proportional to the mode probability. If the mode probability is very low, only a very small fraction of the particles resides in that mode and this causes numerical problems [9]. Therefore, for tracking a manoeuvring target, authors in [9] combined IMM with a regularized particle filter so that a fixed number of particles are assigned to each mode. In addition to these traditional dynamical systems, IMM has been also combined with Gaussian Process Dynamic Models (GPDM) [18]. This mainly aims to effectively handle high dimensionality of the motion states and track the varied human body motion model. All these methods are intended to deal with the problem of variable motion models. However, they did not take into account the second problem -varying number of objects.

Regarding the problem of detecting a varying number of objects, many works have been proposed based on IMM estimators employing a discrete variable indicating the number of present objects. Such methods are commonly named *joint detection and tracking* (JDT) methods, because the detection decision is made by tracking the discrete object state corresponding to the number of present objects.

In other words, detection and tracking of objects are performed jointly in the same framework. Many works have been proposed on various issues of JDT since Ristic $et\,al.$ introduced the basic JDT framework and its applications in their book [80]. Rutten $et\,al.$ proposed two different particle filters for JDT [83]. The first one is an orthodox SIR filter augmenting the target state space with an *existence* variable. However, the second is derived such that the probability of existence is calculated using the weights of the particles so that the target existence does not need to be included as a part of the state vector. JDT methods were then generalized from single object detection to multiple object detection in [43] and [70]. They have also been applied in many applications for detecting different types of objects. For example, [69] jointly detected and tracked unresolved targets with monopulse radar. Additionally, in [28] and [29] Czyz $et\,al.$ applied the JDT method to detect objects in color videos by using a color observation model. A scale-invariant feature transform (SIFT)-based particle filter algorithm has been also presented for joint detection and tracking of independently moving objects in stereo sequences observed by uncalibrated moving cameras in [90]. Although all these methods deal with the varying number of objects of interest, they assume the objects undergo a single motion model. Hence, they are not suitable for problems where objects experience various motion model changes.

In this chapter, we propose a component which addresses both mentioned problems formulated in the JDT framework. The novelty of the proposed method can be summarized in two points. First, we use an IMM estimator formulated in the JDT framework which not only embeds multiple motion models, but also takes into account the varying number of objects. Second, a novel motion model prediction function is introduced. With this function, prediction of the current motion model is not based on the previous model alone, but also on the object kinematic state. Recall that in the existing IMM estimators, prediction of the interacting motion models -among others- follows the basic assumption of the Markov chain. In other words, the current active model is correlated with only the previous model, and the correlation is described by a fixed model transition matrix. For example, in [9, 77, 6] the IMM estimator is employed to switch between different motion models and a fixed

transition matrix is used to determine the probability of switching from one motion model to another. In [18, 46, 7], the IMM estimator is rather employed to switch between different gesture and expression states. A fixed transition matrix is also employed to define the probability of switching from one gesture/expression to another. Furthermore, IMM is also used to estimate the varying number of objects in [43, 70, 69, 28, 29, 90]. In these methods, the probability of transition from a given number of objects to another one is defined by a fixed transition matrix. In all the methods mentioned above, the prediction of the current interacting model is based only on the previous model. However, the interacting model is generally strongly correlated to the object kinematic state. Taking this into account, we define here a model prediction function where the current model is correlated also to the previous object kinematic state. To this end, any prior knowledge about model switching can be exploited to achieve an accurate model prediction, and eventually accurate object detection results. Notice that this new motion model prediction function is general enough to bear with any other interacting models. In this chapter and for sake of clarity, we limit ourselves to interacting motion models.

The remainder of this chapter is organized as follows. In Section 2.2, the general IMM solution for jointly detecting and tracking objects with a single motion model is first described, and then the new solution using adaptive multiple motion models is proposed. The implementation of the proposed solution with a particle filter is detailed in Section 2.3. Section 2.4 includes the experimental results that illustrate the performance of the proposed method, and finally a summary is drawn in Section 2.5.

2.2 Joint Detection and Tracking Using Adaptive Multiple Motion Models

In this section, we first introduce the existing IMM solution for jointly detecting and tracking objects using a single motion model. Then, we describe how this formulation is generalized to bear with the problem of detection with multiple motion models.

2.2.1 Object Detection with a Single Motion Model

In the JDT framework, the IMM estimator tackles the object detection problem by recursively computing the posterior probability density function (pdf) $p(X_t|Z^t)$. Z^t includes all the observations up to time t, $\{Z_1, \ldots, Z_t\}$. X_t is the hybrid object state defined by a variable length vector

$$X_t = [x_t \; E_t]. \tag{2.1}$$

x_t contains the objects' kinematic states at time t describing information including, for instance, location and velocity. $E_t \in \{0, 1, ..., M\}$ is a discrete existence variable associated with a set of interacting models indicating the number of objects that exist at time t. Here, we set the maximum number M to one for the simplicity of explanation, and for further details regarding extension to multiple objects please refer to [28]. Then, $E_t \in \{0, 1\}$ indicates whether the object is present at time t or not. The transition between different interacting models corresponding to E_t follows a Markov chain whose transition probabilities are specified by a transitional probability matrix (TPM)

$$\Pi = \begin{bmatrix} 1 - P_b & P_b \\ P_d & 1 - P_d \end{bmatrix} \tag{2.2}$$

where $P_b = P(E_t = 1|E_{t-1} = 0)$ denotes the probability of object birth and $P_d = P(E_t = 0|E_{t-1} = 1)$ denotes the probability of object death.

To make the detection decision, the probability of an object being present or absent, $P(E_t|Z^t)$, is estimated recursively in each frame. As we have only two states, $P(E_t|Z^t)$ is derived in the following by computing both $P(E_t = 1|Z^t)$ and $P(E_t = 0|Z^t)$. These two probabilities are derived differently because the object state is not defined when the object is not present. The difference will be explained in more details in the following derivations.

The probability of an object being present at the current frame is estimated as follows

$$P(E_t = 1|Z^t) = \int p(x_t, E_t = 1|Z^t) dx_t, \tag{2.3}$$

$$p(x_t, E_t = 1|Z^t) = \frac{p(Z_t|x_t, E_t = 1)\, p(x_t, E_t = 1|Z^{t-1})}{p(Z_t|Z^{t-1})}. \tag{2.4}$$

$p(Z_t|x_t, E_t = 1)$ is the object appearance model updating the predicted hybrid state according to the current observation, and it can be defined according to different applications. More details about the appearance model can be found in Chapter 3, where a new shape based appearance model is proposed. $p(x_t, E_t = 1|Z^{t-1})$ is the predicted hybrid state function which can be derived as follows,

$$\begin{aligned}
&p(x_t, E_t = 1|Z^{t-1}) \\
&= \int p(x_t, E_t = 1|x_{t-1}, E_{t-1} = 1)p(x_{t-1}, E_{t-1} = 1|Z^{t-1})dx_{t-1} \\
&+ p(x_t, E_t = 1, E_{t-1} = 0|Z^{t-1}) \\
&= \int p(x_t|x_{t-1}, E_t = 1, E_{t-1} = 1)(1 - P_d)p(x_{t-1}, E_{t-1} = 1|Z^{t-1})dx_{t-1} \\
&+ p_b(x_t)P_bP(E_{t-1} = 0|Z^{t-1}).
\end{aligned} \tag{2.5}$$

On the right hand side of the above equation, $p(x_t|x_{t-1}, E_t = 1, E_{t-1} = 1)$ is the object kinematic state transition function specified by the object's motion model. $p(x_{t-1}, E_{t-1} = 1|Z^{t-1})$ is the previous posterior pdf and $P(E_{t-1} = 0|Z^{t-1})$ is the previous object absence probability. $p_b(x_t)$ is the initial object pdf where subscript b stands for "birth". If no prior knowledge is available, it may be assumed to be uniformly distributed.

Similarly, the probability of an object being absent at the current frame is estimated according to

$$P(E_t = 0|Z^t) = \frac{p(E_t = 0, Z_t|Z^{t-1})}{p(Z_t|Z^{t-1})}, \tag{2.6}$$

$$\begin{aligned}
&p(E_t = 0, Z_t|Z^{t-1}) \\
&= p(Z_t|E_t = 0)\, P(E_t = 0|Z^{t-1}) \\
&= p(Z_t|E_t = 0)(P(E_t = 0, E_{t-1} = 0|Z^{t-1}) + P(E_t = 0, E_{t-1} = 1|Z^{t-1})) \\
&= p(Z_t|E_t = 0)((1 - P_b)P(E_{t-1} = 0|Z^{t-1}) + P_dP(E_{t-1} = 1|Z^{t-1})).
\end{aligned} \tag{2.7}$$

$p(Z_t|E_t = 0)$ is the likelihood function when the object is absent, and it is independent of the object state. This function is normally set as a constant.

2.2.2 Object Detection with Multiple Motion Models

The JDT method described above assumes that the object kinematic state transition function in Eq. (2.5), $p(x_t|x_{t-1}, E_t = 1, E_{t-1} = 1)$, is specified by a dynamic model $x_t = f(x_{t-1}, w(t-1))$. $f(\cdot)$ is a known function determined by an assumed motion model and $w(t-1)$ is the process noise. However in many problems, objects may undergo more than one motion model. For example, an aircraft moving in a constant velocity may accelerate suddenly. In this case if we use a constant velocity model to define the state transition probability function, then the method will lose track, as well as the detection, of the aircraft after it accelerates.

To improve the above JDT method so that it can detect objects that may undergo multiple motion models, two main improvements are introduced. In summary, the first improvement combines two object state formulations: $[x_t, E_t]$ which tackles the varying number of objects and $[x_t, \alpha_t]$ which tackles the varying object motion models. The combined object state $[x_t, \alpha_t, E_t]$ allows taking into account multiple motion models in the JDT framework. Intuitively, the prediction of the object kinematic state x_t is dependent not only on the previous object kinematic state x_{t-1}, but also the current motion model α_t as explained in Fig. 2.1. Furthermore, the second improvement formulates JDT with multiple motion models adaptively by employing a novel motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ which extends the original one $P(\alpha_t|\alpha_{t-1})$. Intuitively, the prediction of the object motion model α_t is dependent not only on the previous motion model α_{t-1}, but also the previous object kinematic state x_{t-1} as explained in Fig. 2.1. The details of the above two improvements are explained in the following.

Multiple motion model formulation: First, we formulate the JDT framework with multiple motion models. As mentioned before, there is normally one interacting model set in each IMM estimator. For example, a model set containing different motion models is used to track objects with variable motion models. For another example, a model set containing different present object numbers is used in JDT methods to estimate the number of objects in each frame. However when these two problems occur simultaneously, i.e., when objects that may change their motion models need to be detected, one model set is not enough. Therefore, as the first step

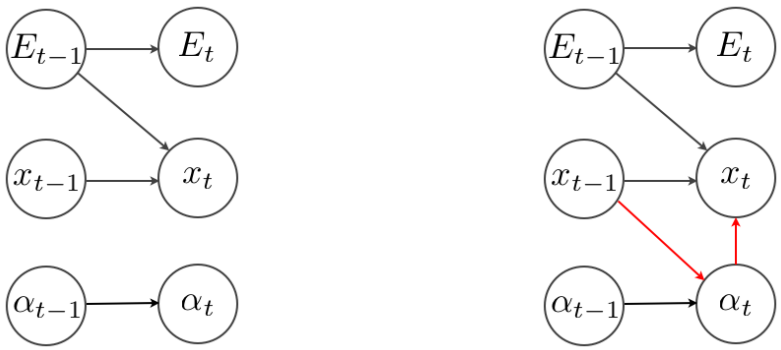


Figure 2.1: (a) The prediction strategies of the object kinematic state in the existing JDT methods, and of the motion model in the existing model prediction functions. (b) The prediction strategy of the object kinematic state and the motion model in the proposed JDT method.

of the modification, we formulate the IMM estimator in JDT methods to contain an additional set of motion models besides its original set of object presence models.

To this end, a discrete variable α_t that allows the filter to switch between different motion models is added to the object state $[x_t\ E_t]$. Hence, the original hybrid object state is modified to

$$X_t = [x_t\ \alpha_t\ E_t]. \tag{2.8}$$

$\alpha_t \in \{1, \ldots, M_m\}$ and M_m is the number of possible motion models. With the new variable α_t, the filter can switch between different motion models to select the appropriate one which best fits the object's actual model. If this is set properly, the object can be tracked accurately even when it changes its motion.

After the additional motion model variable is added to the hybrid object state, the original predicted hybrid state function in Eq. (2.5) has to be re-derived as

$$\begin{aligned}
&p(x_t, \alpha_t, E_t = 1|Z^{t-1}) \\
&= \sum_{\alpha_{t-1}} \int p(x_t, \alpha_t, E_t = 1|x_{t-1}, \alpha_{t-1}, E_{t-1} = 1) \\
&p(x_{t-1}, \alpha_{t-1}, E_{t-1} = 1|Z^{t-1})dx_{t-1} + p(x_t, \alpha_t, E_t = 1, E_{t-1} = 0|Z^{t-1}) \\
&= \sum_{\alpha_{t-1}} \int p(x_t|\alpha_t, x_{t-1}, E_t = 1, E_{t-1} = 1)P(\alpha_t|x_{t-1}, \alpha_{t-1})(1 - P_d) \\
&p(x_{t-1}, \alpha_{t-1}, E_{t-1} = 1|Z^{t-1})dx_{t-1} + p_b(x_t)P_b(\alpha_t)P_bP(E_{t-1} = 0|Z^{t-1}).
\end{aligned} \tag{2.9}$$

Similarly, $P_b(\alpha_t)$ is the probability of the initial object motion model where subscript b stands for "birth", and if no prior knowledge is available, it may be assumed uniformly distributed. Different from the original predicted hybrid state function in Eq. (2.5), the new function's object kinematic state transition function $p(x_t|\alpha_t, x_{t-1}, E_t = 1, E_{t-1} = 1)$ is characterized by its α_t^{th} motion model. Here α_t is one of the object's possible motion models, rather than a unique pre-fixed motion model as in the original formulation.

Motion model prediction function: Once the JDT framework is formulated with multiple motion models, we employ a novel adaptive motion model prediction function. In the existing methods, the transition of the motion model variable α_t is modeled by a Markov chain as discussed in Section 2.1. In other words, the motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ is assumed to be dependent only on the previous model.

$$P(\alpha_t|x_{t-1}, \alpha_{t-1}) \equiv P(\alpha_t|\alpha_{t-1}), \tag{2.10}$$

and the transitions are specified by an $M_m \times M_m$ transitional probability matrix, $\Pi = [\pi_{ij}]$, where

$$\pi_{ij} = P(\alpha_t = j|\alpha_{t-1} = i),\ (i, j \in \{1, \ldots, M_m\}). \tag{2.11}$$

In this thesis, we rewrite the above prediction function to take into account, in addition to the previous model, the object previous kinematic state. In this way, prior knowledge on when and/or where to switch to a certain motion model can be exploited to predict the model adaptively. Based on the Bayes rules, the new motion model prediction function is written as,

$$\begin{aligned} P(\alpha_t|x_{t-1}, \alpha_{t-1}) &= \frac{p(\alpha_t, x_{t-1}, \alpha_{t-1})}{\sum_{\alpha_t} p(\alpha_t, x_{t-1}, \alpha_{t-1})} \\ &= \frac{p(x_{t-1}, \alpha_{t-1}|\alpha_t)P(\alpha_t)}{\sum_{\alpha_t} p(x_{t-1}, \alpha_{t-1}|\alpha_t)P(\alpha_t)} \\ &= \frac{p(x_{t-1}|\alpha_t)P(\alpha_{t-1}|\alpha_t)P(\alpha_t)}{\sum_{\alpha_t} p(x_{t-1}|\alpha_t)P(\alpha_{t-1}|\alpha_t)P(\alpha_t)} \\ &= \frac{p(x_{t-1}|\alpha_t)P(\alpha_{t-1}, \alpha_t)}{\sum_{\alpha_t} p(x_{t-1}|\alpha_t)P(\alpha_{t-1}, \alpha_t)} \end{aligned} \tag{2.12}$$

Compared with the corresponding function in Eq. (2.10), the new function embeds an additional correlation between the objects' current motion model and kinematic state. This function indicates the likelihood of switching to a certain model α_t given both the previous model α_{t-1} and kinematic state x_{t-1}. It can be easily seen that when the second correlation is not taken into account, Eq. (2.12) simply degenerates to the equality in Eq. (2.10). To determine this prediction function, two terms in the right hand side of Eq. (2.12), $P(\alpha_{t-1}, \alpha_t)$ and $p(x_{t-1}|\alpha_t)$, need to be estimated from a training set.

A training set of N samples need to be collected from a sequence as follows,

$$\{(\alpha_t^1, x_{t-1}^1, \alpha_{t-1}^1), (\alpha_t^2, x_{t-1}^2, \alpha_{t-1}^2), ..., (\alpha_t^N, x_{t-1}^N, \alpha_{t-1}^N)\}.$$

Each sample pair contains the motion model at a certain time step, the object kinematic state and motion model at the previous time step. The first term that needs to be determined is the joint probability of α_{t-1} and α_t denoted by $P(\alpha_{t-1}, \alpha_t)$. It can be estimated as the relative frequency of the event (α_{t-1}, α_t) occurring in the training set by the standard method for maximum-likelihood parameter learning [82].

$$P(\alpha_{t-1} = i, \alpha_t = j) = \frac{\sum_s \delta((\alpha_t^s, x_{t-1}^s, \alpha_{t-1}^s), (j, x_{t-1}^s, i))}{N} \tag{2.13}$$

where δ is the Kroneker delta function: $\delta(a, b) = 1$, if $a = b$, and zero otherwise.

The second term that needs to be determined, $p(x_{t-1}|\alpha_t)$, is a probability density which can be estimated by various parametric and nonparametric methods [82]. Parametric methods assume a certain model for the pdf to be estimated, and then fit the parameters of that family of models to the observed data set. Examples of the parametric models include Gaussian, Poisson, and Beta distributions. The problem with the parametric methods is that they often oversimplify the underlying distribution complexity of the data from the real world. In contrast to parametric methods, nonparametric methods allow the underlying distribution complexity to grow with the data. Therefore they are preferred in estimating the probability density distribution $p(x_{t-1}|\alpha_t)$, because the correlation conveyed by this distribution is often complicated and varies with the applications. Popular nonparametric methods include histogram and kernel density estimation (KDE), and the latter is employed in

our work because it provides better performance than the former [25]. Hence, an individual probability density $p(x_{t-1}|\alpha_t = i)$ can be estimated for each model from the training subset $Y(i) = \{(\alpha_t^s, x_{t-1}^s, \alpha_{t-1}^s)|\alpha_t^s = i, s = 1 : N\}$.

$$p(x_{t-1}|\alpha_t = i) = \frac{1}{N} \sum_{(\alpha_t^s, x_{t-1}^s, \alpha_{t-1}^s) \in Y(i)} K(x_{t-1}, x_{t-1}^s), \tag{2.14}$$

where $K(x_{t-1}, x_{t-1}^s)$ is a kernel function, and the most popular one is the Gaussian [82]. Now, the adaptive motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ can be determined.

In the following, a large lump detection example is used to illustrate the proposed motion model prediction function more clearly. In this problem, lumps may be described roughly by two motion models: a slow acceleration model m_1 due to the perspective distortion when a lump moves on the belt (as shown by the two consecutive frames in Figs 2.2(a) and 2.2(b)), and a fast acceleration model m_2 due to gravity when the lump reaches the end of the belt and falls down (as shown by the two consecutive frames in Figs 2.2(b) and 2.2(c)). Therefore, an interacting model set containing two motion models $\{m_1, m_2\}$ is needed, and the possibility of switching to each motion model is not the same everywhere in the image. For example, it is much more probable that the lump switches to the fast acceleration model m_2 when it gets close to the end of the belt than in any other location. Therefore, the kinematic state, the lump location on the row axis in this case, is apparently correlated to the motion model, and this correlation can be learned to predict the motion model accurately. To achieve this, a training set of 60 samples is collected as described previously,

$$\{(m_1, 1, m_1), (m_1, 10, m_1), ..., (m_1, 260, m_2)\}.$$

Each sample pair contains the lump's motion model at a certain time step and its location on the row axis and motion model in the previous time step. In this example, the only kinematic state correlated to the motion model is the lump's location on the row axis. Based on the training set, the joint probability $P(\alpha_{t-1}, \alpha_t)$ can be estimated as in Table 2.1. Furthermore, the probability density $p(x_{t-1}|\alpha_t)$ can also be estimated by the kernel density estimation method. Fig. 2.3(a) shows the two individually estimated density distributions $p(x_{t-1}|\alpha_t = m_1)$ and $p(x_{t-1}|\alpha_t = m_2)$. Up

Table 2.1: The joint probability $P(\alpha_{t-1}, \alpha_t)$ in the large lump detection example.

$Pr(\alpha_{t-1}, \alpha_t)$	$\alpha_t = m_1$	$\alpha_t = m_2$
$\alpha_{t-1} = m_1$	0.97	0.03
$\alpha_{t-1} = m_2$	0	0

to this point, the motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ can be completely determined. Fig. 2.3(b) compares between this prediction function, taking into account the kinematic state, and the original prediction function which uses only the transition probability between different motion models. As shown in the figure, the new prediction function $P(\alpha_t = m_2|x_{t-1}, \alpha_{t-1} = m_1)$ varies depending on the object kinematic state x_{t-1} (the lump's location on the row axis). Contrarily, the original prediction function $P(\alpha_t = m_2|\alpha_{t-1} = m_1)$ does not embed such correlation information, and therefore, appears as a horizontal line corresponding to a constant transition probability.

(a) (b) (c)

Figure 2.2: Three consecutive frames from the LLD application

In Summary, the new JDT method takes advantage of two kinds of prior motion knowledge which are the eventual motion models that the object may undergo, and the probabilities of models switchings. Thus, the proposed JDT method should achieve better performances than the existing methods when dealing with objects with variable motions models.

2.3 Particle Filter Implementation

The JDT method derived in the previous section is implemented with a particle filter described with a pseudo-code depicted in Fig. 2.4. The posterior pdf at the previous

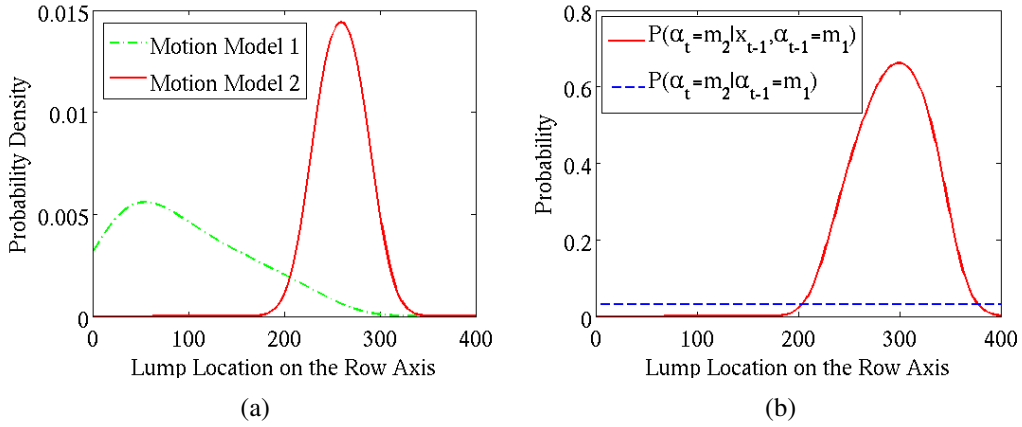


Figure 2.3: (a) The two estimated probability density distributions of the lump location on the row axis given the two potential motion models in the large lump example. The dashed line corresponds to $p(x_{t-1}|\alpha_t = m_1)$ and the solid line corresponds to $p(x_{t-1}|\alpha_t = m_2)$. (b) The comparison between the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ and the original prediction function $P(\alpha_t|\alpha_{t-1})$, using the probability of the transition between the previous motion model m_1 and the current motion model m_2 as an example.

time is approximated by a set of weighted particles $\{(x^i_{t-1}, \alpha^i_{t-1}, E^i_{t-1}, w^i_{t-1})\}^N_{i=1}$, where w^i_{t-1} is the i^{th} particle weight at time $t-1$. The input to the algorithm is the set of particles at a previous time and the current observed image, and the output is the set of particles at the current time. We will next explain the algorithm in more details.

- The first step predicts the current existence variable according to the previous existence variable and the TPM specified by Eq. (2.2) (refer to the pseudo-code in Fig. 2.4).

- Given the predicted existence variable, the second step predicts the object's current state based on the previous state and the predicted state function defined in Eq. (2.9).

- The third step weighs the particles representing the predicted state given the current observed image. To achieve the detection mechanism conveniently, the likelihood ratio is used as the particle weight here rather than the mea-

Input: $\{(x_{t-1}^i, \alpha_{t-1}^i, E_{t-1}^i, w_{t-1}^i)\}_{i=1}^N$, Z_t
Output: $\{(x_t^i, \alpha_t^i, E_t^i, w_t^i)\}_{i=1}^N$
1. Given $\{E_{t-1}^i\}_{i=1}^N$, generate $\{E_t^i\}_{i=1}^N$ according to the TPM specified by Eq. (2.2).
2. Based on $\{E_{t-1}^i\}_{i=1}^N$ and $\{E_t^i\}_{i=1}^N$, generate $\{(x_t^i, \alpha_t^i, E_t^i)\}_{i=1}^N$ from $\{(x_{t-1}^i, \alpha_{t-1}^i, E_{t-1}^i)\}_{i=1}^N$ according to Eq. (2.9).
3. Given Z_t, compute the weights $\{w_t^i\}_{i=1}^N$ for $\{(x_t^i, \alpha_t^i, E_t^i)\}_{i=1}^N$ according to Eq. (2.15).
4. Normalize $\{w_t^i\}_{i=1}^N$ to $\{\tilde{w}_t^i\}_{i=1}^N$.
5. Resample from $\{(x_t^i, \alpha_t^i, E_t^i, \tilde{w}_t^i)\}_{i=1}^N$ for N times to obtain a new set of particles $\{(x_t^i, \alpha_t^i, E_t^i, 1/N)\}_{i=1}^N$.

Figure 2.4: The particle filter implementation of the proposed JDT method.

surement model $p(Z_t|X_t)$. The likelihood ratio can be calculated as

$$L(Z_t|X_t) = \begin{cases} \frac{p(Z_t|x_t,\alpha_t,E_t=1)}{p(Z_t|E_t=0)} & \text{for} \quad E_t = 1, \\ 1 & \text{for} \quad E_t = 0. \end{cases} \tag{2.15}$$

- The fourth step performs normalization.
- The fifth and last step is the standard resampling step, which converts the set of weighted particles back to an equivalent set of unweighed particles approximating the current posterior pdf.

Once the posterior pdf $p(x_t, \alpha_t, E_t|Z^t)$ is approximated by the set of particles, the probability that the object exists is estimated based on Eq. (2.3) as

$$P(E_t = 1|Z^t) = \frac{1}{N} \cdot \sum_{i=1}^{N} \delta(E_t^i, 1). \tag{2.16}$$

2.4 Experimental Results

In this section, we test the proposed JDT method for detecting objects with multiple motion models using three types of experiments: (1) a synthetic example to illustrate the effectiveness of the formulation, (2) a real example from the large lump detection application which motivates this work , and (3) three general object detection examples to demonstrate the flexibility of the method. Further experiments

are carried out in order to study the effect of each level of the proposed technique through various comparisons. These comparisons are organized as follows:

- First, we compare the proposed method without the adaptive model prediction function, to which we refer by MMMJDT, against existing JDT techniques with a single motion model assumption, referred to by SMMJDTs. This experiment aims to study the effect of using a multiple motion model assumption rather than a single motion model. For this purpose, experiments include objects which undergo sudden motion changes.

- Second, we compare the MMMJDT with the proposed method with all its components including the new model prediction function. Recall that this function exploits the correlation between the motion models and object kinematic state. We refer to our complete method by Adaptive MMMJDT. To this end, we consider experiments where a prior knowledge about the correlation between motion changes and the object's kinematic state is available.

2.4.1 Synthetic Experiments

In this experiment, we follow the synthetic example in [28] and create a scenario where an object moves back and forth as shown in Fig. 2.5(a). The object of interest is a square with the pattern shown in Fig. 2.5(b). The object moves from one side to the other in a constant velocity for 10 frames before it changes its direction, and then, it repeats the same motion pattern. This motion pattern is assumed to be known a priori and we use two motion models to describe it, a constant velocity model

$$v_t = v_{t-1} + w_{t-1} \tag{2.17}$$

and a bouncing model, i.e., the same speed as the previous time instant but in the opposite direction

$$v_t = -v_{t-1} + w_{t-1}, \tag{2.18}$$

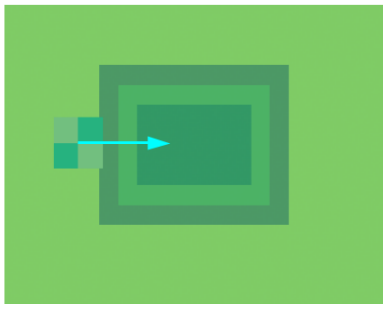
and w_{t-1} is the process noise. In all the competing methods used in the following, the adopted appearance model is defined by

$$p(Z_t|X_t) = \frac{1}{\sqrt{2\pi}\delta} exp(\frac{-dist(q_t, q^*)^2}{2\delta^2}). \tag{2.19}$$

q_t is the intensity histogram computed from image Z_t, specifically from the object region specified by X_t. $dist(q_t, q^*)$ is the distance between the object's histogram and the reference histogram q^*, and it is computed based on the Bhattacharyya similarity coefficient as follows,

$$dist(q_t, q^*) = \sqrt{1 - \sum_{u=1}^{U} \sqrt{q_t(u) \cdot q^*(u)}}. \tag{2.20}$$

U is the number of bins over which the histogram is calculated. Although very simple, this scenario illustrates very well the problem of interest.



(a)

(b)

Figure 2.5: (a) Experimental scenario. The background is represented by nested rectangles each of a different shape and an object by a small square with the texture shown in the right panel. (b) Object appearance used in the experiment. Note that in this experiment an object is modeled by its intensity histogram in a rectangular region, although with modification, other types of object models can be accommodated by the algorithm.

As mentioned before, we first compare the performances of SMMJDTs and MMMJDT. In this case we employ two single motion model JDT methods, one uses a random walk model (named SMMJDT1) and the other uses a constant velocity model (named SMMJDT2). Fig. 2.6(a) shows the detection results of all the three competing methods. The x-axis is the frame number and the y-axis is the existence probability of having an object, $P(E_t = 1|Z^t)$. The horizontal dashed line at existence probability of 0.5 indicates a tunable threshold that determines whether

the object exists or not. The results show that MMMJDT can detect the object consistently with an existence probability above 0.5 as the embedded multiple motion models handle the object's motion changes to a certain extent. In contrast, the constant velocity model JDT method, SMMJDT2, loses the track when the object moves with a motion pattern inconsistent with its assumed single motion model and, as a result, its existence probability drops repeatedly every 10 frames, leading to incorrect detection decisions. Moreover, the naive random walk method, SMMJDT1, amounts to assuming that no prior motion information is available. To deal with the possible change of motion model, SMMJDT1 has to diverge its prediction with a large variance. As a result, when the appearance model is not accurate enough, the object cannot be detected as shown in Fig. 2.6(a).

In the following, we compare MMMJDT and Adaptive MMMJDT to investigate the input of the motion model prediction function. In this case, the prediction function exploits a trivial prior knowledge that one may extract from this experiment. Indeed, the object is more likely to switch from its constant velocity model to the bouncing model near the two locations close to the left and right borders of the image. This prior knowledge can be learned from the training set as presented in Section 2.2.2, and it is illustrated in Fig. 2.7(a) by the two individually estimated density distributions $p(x_{t-1}|\alpha_t = m_1)$ and $p(x_{t-1}|\alpha_t = m_2)$. In this specific case, the kinematic state x_{t-1} is simply the object location on the column axis, and the two motion models m_1 and m_2 are respectively the constant velocity model in Eq. (2.17) and the bouncing model in Eq. (2.18). By embedding this prior knowledge in the motion model prediction function according to Eq. (2.12), the new function can predict the current motion model adaptively based on the previous motion model and kinematic state as explained in Fig. 2.7(b). However, when such knowledge is ignored, the original prediction function predicts the current motion model based on the previous one with a fixed transition probability. As a result, the detection performance of the Adaptive MMMJDT employing the new prediction function is further enhanced in comparison to what MMMJDT has reached. This is mainly illustrated by Fig. 2.6(b). One can also notice that the Adaptive MMMJDT outperforms MMMJDT in terms of the existence probability mainly because it handles

better the object's motion changes. In these situations, the MMMJDT curve drops slightly before the method corrects itself and the curve goes up again. Contrarily, the Adaptive MMMJDT predicts such situations adaptively and the corresponding curve maintains the same level.

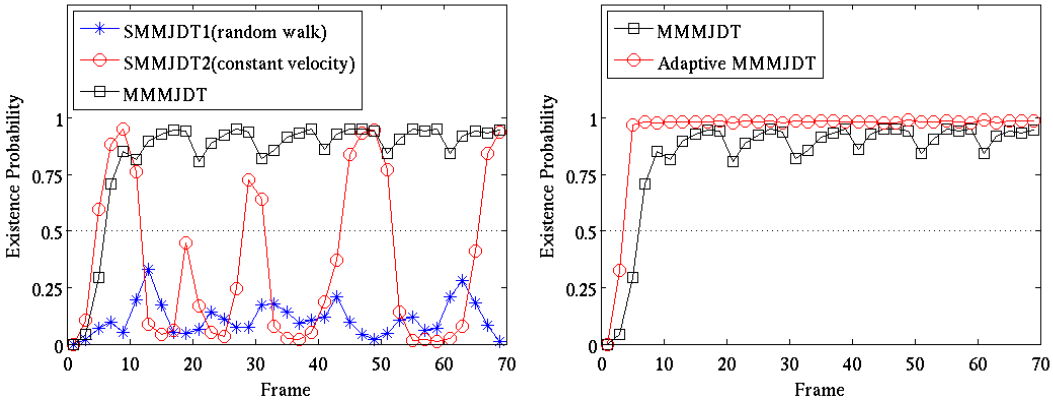
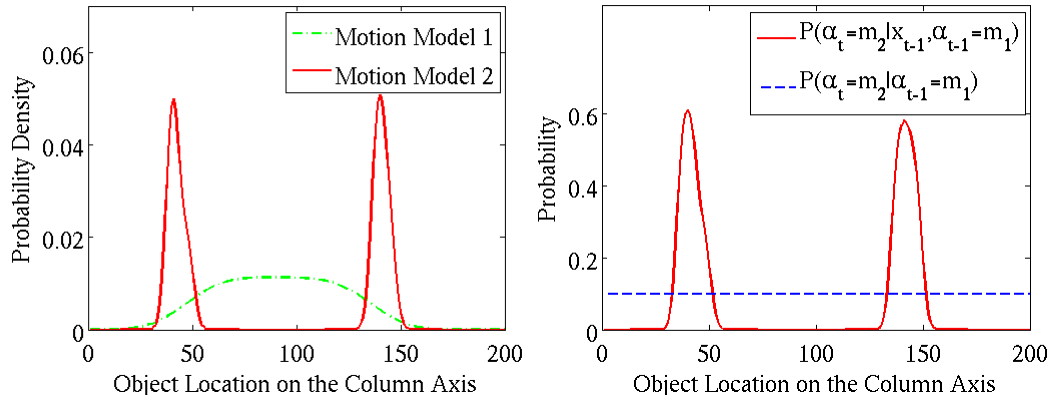


Figure 2.6: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step), SMMJDT1 (random walk method) and SMMJDT2 (constant velocity model method). (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

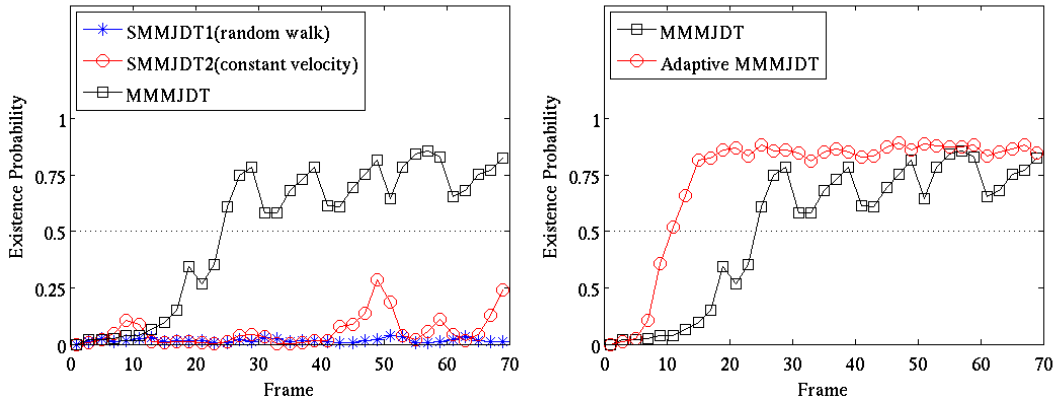
Furthermore, we also tested the previous four methods on the same video but perturbed with a high level of noise. Similarly, the results, in terms of existence probability, are depicted in Fig. 2.8. In spite of the high level of noise in this case, MMMJDT is able to accumulate the evidence consistently even though the object changes its model, and finally detect the object after strong enough evidence is accumulated. Contrarily, the constant velocity model method, SMMJDT2, loses its accumulated evidence every time the object changes its model, and as a result the object can never be detected. The same behaviour is noticed for the random walk method, SMMJDT1, when significant noise exists, indicating that conservative prediction is unable to detect the object. We then applied Adaptive MMMJDT to exploit the prior knowledge on motion model switching as we did in the previous experiment. With this additional piece of information, the detection performance is considerably enhanced as it is demonstrated in Fig. 2.8(b). Consistent with the results in Fig. 2.6, we can see that the proposed JDT method has the best performance



(a) (b)

Figure 2.7: (a) The two estimated probability density distributions of the synthetic object's location on the column axis given the two potential motion models, the constant velocity model (m_1) and the bouncing model (m_2). The dashed line corresponds to $p(x_{t-1}|\alpha_t = m_1)$ and the solid line corresponds to $p(x_{t-1}|\alpha_t = m_2)$. (b) The comparison between the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ and the original prediction function $P(\alpha_t|\alpha_{t-1})$, using the probability of the transition between the previous motion model m_1 and the current motion model m_2 as an example.

dealing with the situations where the object changes its motion model.



(a) (b)

Figure 2.8: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step), SMMJDT1 (random walk method) and SMMJDT2 (constant velocity model method) when significant noise exists in the video. (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

2.4.2 Large Lump Detection

We have also tested our proposed JDT method in the large lump detection application which is the main motivation behind this work although the proposed technique is not limited to this application. In the following, we will use a large lump example to show the detection performance of our proposed method compared to the existing JDT methods with single motion models.

With a simple visual inspection of the large lump sequence, we conclude that lumps, generally, move from top to bottom along a verticale line. Initially, lumps move slowly and then gradually faster and faster on a conveyor, and finally they drop when they reach the end to fall into the crusher. Therefore, lumps may be considered to have two motion models, a small constant acceleration model on the conveyor

$$v_t^y = v_{t-1}^y + a_s + w_{t-1}$$

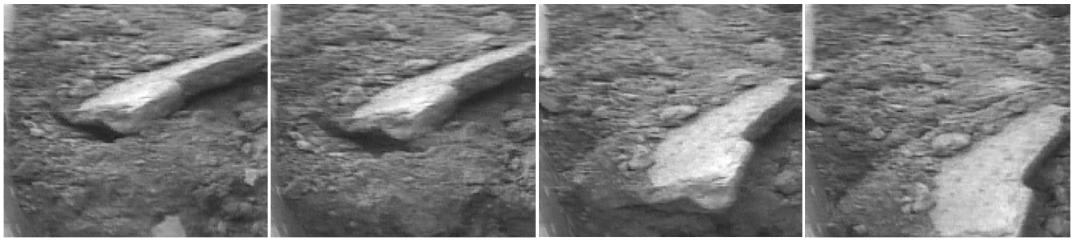
and a large constant acceleration model off the conveyor

$$v_t^y = v_{t-1}^y + a_l + w_{t-1}.$$

a_s and a_l are the small and large constant acceleration rates estimated in the training set. v_t^y is the velocity in the vertical direction. In both models, the velocity in the horizontal direction is constant, i.e., $v_t^x = v_{t-1}^x + w_{t-1}$. We utilize these two motion models adaptively in our method to detect lumps and compare the results with classic JDT methods using only one motion model. The appearance model used in these algorithms can be found in [97] where a feature detector is proposed specifically for this large lump detection problem.

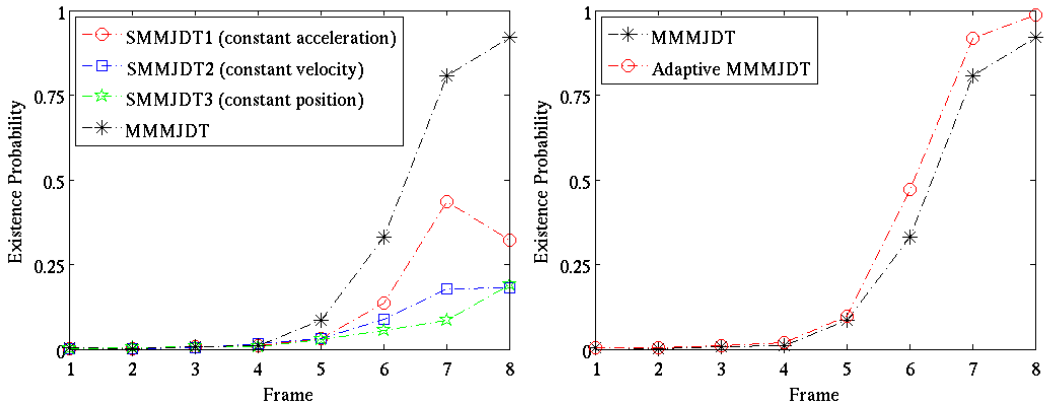
Fig. 2.9 shows a few consecutive frames where one lump moves down into the crusher. Initially, it moves really slowly, for example from frame 1 (Fig. 2.9(a)) to frame 2 (Fig. 2.9(b)), and then gradually faster until frame 7 (Fig. 2.9(c)). From frame 7 (Fig. 2.9(c)) to frame 8 (Fig. 2.9(d)) the lump moves much faster than before. Fig. 2.10(a) shows the comparison of four methods, MMMJDT using two constant acceleration models and three SMMJDTs with respectively a small constant acceleration model, a constant velocity model, and a constant position model. The three SMMJDTs methods do not match the actual lump motion model very well

especially from frame 7 to frame 8 when the lump changes its acceleration rate and they obviously lose the track as well as the detection of the lump. MMMJDT, in contrast, continues accumulating the evidence and finally detects the lump. Even from frame 7 to frame 8, it can still adjust itself by switching the filter's model to fit the actual lump motion model. Now we incorporate the prior knowledge that lumps tend to move to the second motion model when they become very close to the bottom of the image. This correlation information is learned following Eq. (2.12), and it is discussed previously in Fig. 2.3. By employing this correlation information in the new motion model prediction function, the Adaptive MMMJDT reaches even better performance compared to MMMJDT as illustrated in Fig. 2.10(b).



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

Figure 2.9: Sequence of a lump. (a) Frame 1, (b) Frame 2, (c) Frame 7 (d) Frame 8.



(a) (b)

Figure 2.10: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step) and SMMJDTs (three single motion model JDT methods) detecting the large lump in Fig. 2.9. (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

2.4.3 Additional Experiments in Object Detection

In this subsection, we compare the detection performances of the proposed multiple motion model JDT method and the JDT methods using a single motion model on three video sequences involving the tracking and detection of different objects. All the three sequences are extracted from YouTube. The purpose is to show that the proposed JDT method is flexible enough to be able to make performance improvements in general. In all the three examples, the appearance model used in all the competing methods is based on color histogram as defined in Eq. (2.19).

The first sequence *ping-pong* contains 200 frames, where a ping-pong ball is the object of interest for detection. Fig. 2.11 shows a sample frame where the ball is highlighted by a big rectangle for better visualization. This sequence shows an example of real world objects that undergo multiple motion models. In this case, the ping-pong ball's motion can be described by three models: a constant velocity model

$$v_t = v_{t-1} + w_{t-1}, \tag{2.21}$$

a vertical direction bouncing model (constant speed while opposite direction vertically)

$$v_t^y = -v_{t-1}^y + w_{t-1}, v_t^x = v_{t-1}^x + w_{t-1}, \tag{2.22}$$

and a horizontal direction bouncing model (constant speed while opposite direction horizontally)

$$v_t^x = -v_{t-1}^x + w_{t-1}, v_t^y = v_{t-1}^y + w_{t-1}. \tag{2.23}$$

The latter two models are due to the ball hitting the table and the bats. This sequence presents a difficulty that the object switches among three motion models frequently. To tackle this difficulty, our proposed method first embeds the three motion models into the JDT framework. Then it exploits the correlation between these models and object kinematic state when predicting the current motion model. Intuitively, a correlation comes out, in this case, from the fact that the three motion models are more likely to appear at different specific locations. Therefore, it can be learned from the training sequence and utilized by our method following Eq. (2.12). This learned information is illustrated in Fig. 2.12(a) by the three individually estimated

density distributions $p(x_{t-1}|\alpha_t = m_1)$, $p(x_{t-1}|\alpha_t = m_2)$ and $p(x_{t-1}|\alpha_t = m_3)$. In this case, the object kinematic state x_{t-1} is simply the object location and the three motion models m_1, m_2 and m_3 are respectively the ones described in Eqs (2.21), (2.22) and (2.23). As shown in Fig. 2.12(a), the kinematic state distributions corresponding to the second and third motion models, $p(x_{t-1}|\alpha_t = m_2)$ and $p(x_{t-1}|\alpha_t = m_3)$, peak at certain locations, while the distribution corresponding to the first constant velocity motion model is not biased towards any specific location. By embedding the information described above, the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ predicts adaptively with regard to the value of the object kinematic state as shown in Fig. 2.12(b). Contrarily, the original motion model prediction function $P(\alpha_t|\alpha_{t-1})$ depends only on the previous motion model and remains a constant regarding the previous object kinematic state. Fig. 2.12(b) shows this comparison using the transition between the motion model m_1 and m_3 as an example, i.e., $P(\alpha_t = m_3|x_{t-1}, \alpha_{t-1} = m_1)$ and $P(\alpha_t = m_3|\alpha_{t-1} = m_1)$.

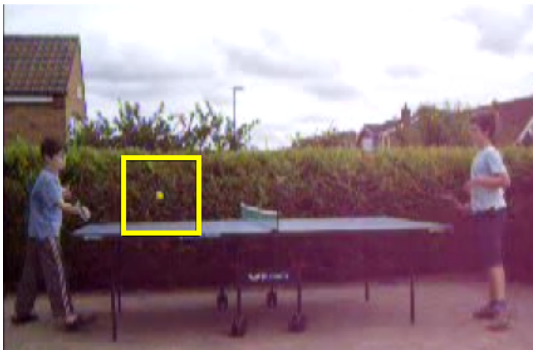
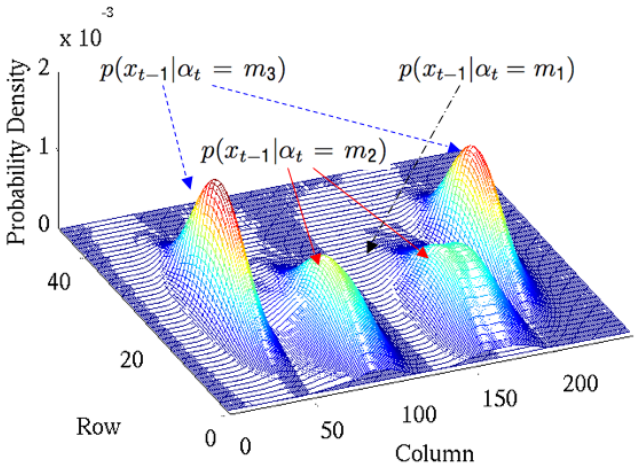
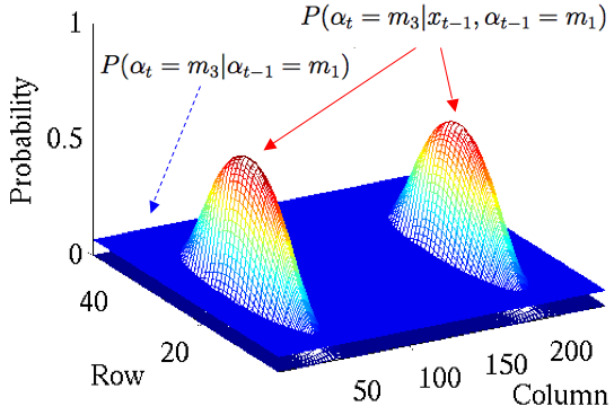


Figure 2.11: An example frame of the sequence *ping pong*.

In the following, two comparisons as in the previous experiments will show the effects of each component of the proposed method on this *ping-pong* sequence. Fig. 2.13(a) shows the comparison between MMMJDT and two SMMJDTs. The latter two methods employ a model with constant velocity in both vertical and horizontal directions (indicated by SMMJDT1) and a model with constant velocity in horizontal direction while constant acceleration in vertical direction (indicated by SMMJDT2). From this figure, it is shown that with a single motion model, SMMJDT1 and SMMJDT2 cannot accumulate the detection evidence properly. In other words,



(a)

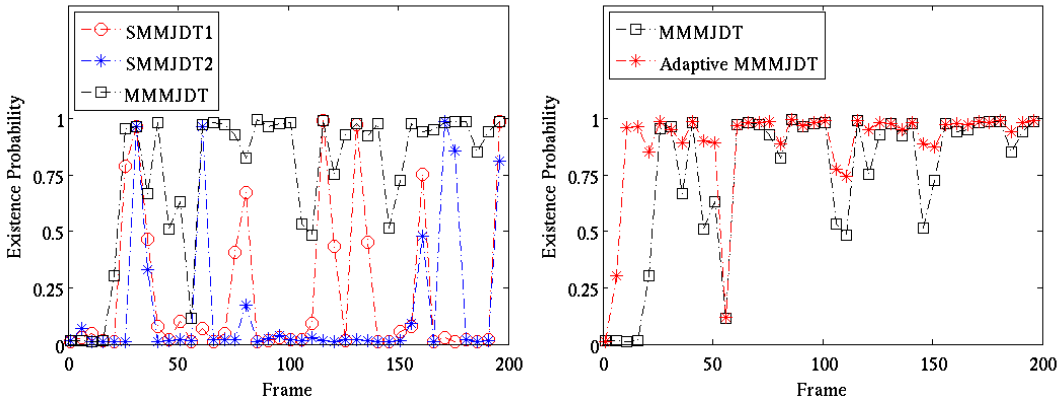


(b)

Figure 2.12: (a) The three estimated probability density distributions of the ping-pong ball's location given the three potential motion models, $p(x_{t-1}|\alpha_t = m_1)$ (corresponding to the constant velocity model), $p(x_{t-1}|\alpha_t = m_2)$ (corresponding to the vertically bouncing model), and $p(x_{t-1}|\alpha_t = m_3)$ (corresponding to the horizontally bouncing model). $p(x_{t-1}|\alpha_t = m_2)$ and $p(x_{t-1}|\alpha_t = m_3)$ peak at certain locations illustrated in the figure, while $p(x_{t-1}|\alpha_t = m_1)$ does not peak prominently as shown in the figure. (b) The comparison between the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ and the original prediction function $P(\alpha_t|\alpha_{t-1})$, using the probability of the transition between the previous motion model m_1 and the current motion model m_3 as an example.

the ball cannot be detected consistently. However by embedding multiple motion models in JDT, a better detection performance can be achieved and this is illustrated by the higher estimated existence probability corresponding to MMMJDT. On this basis, the correlation information can be exploited to further improve the detection

performance as shown in Fig. 2.13(b). Our proposed method, the Adaptive JDT, has a higher estimated existence probability than MMMJDT. It demonstrates that a more accurate model prediction as well as detection decision can be made when the correlation information is available and is used.



(a) (b)

Figure 2.13: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step) and SMMJDTs (two single motion model JDT methods) detecting the ping pong ball in sequence *ping pong*. (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

The second sequence *soccer* contains 130 frames where a girl jogs a soccer ball. A sample frame of this sequence is shown in Fig. 2.14. The soccer ball shows another example where a real world object may have multiple motion models. In this example, the soccer ball's motion can be described roughly by a constant acceleration model

$$v_t^y = v_{t-1}^y + a + w_{t-1} \tag{2.24}$$

and a vertical direction bouncing model

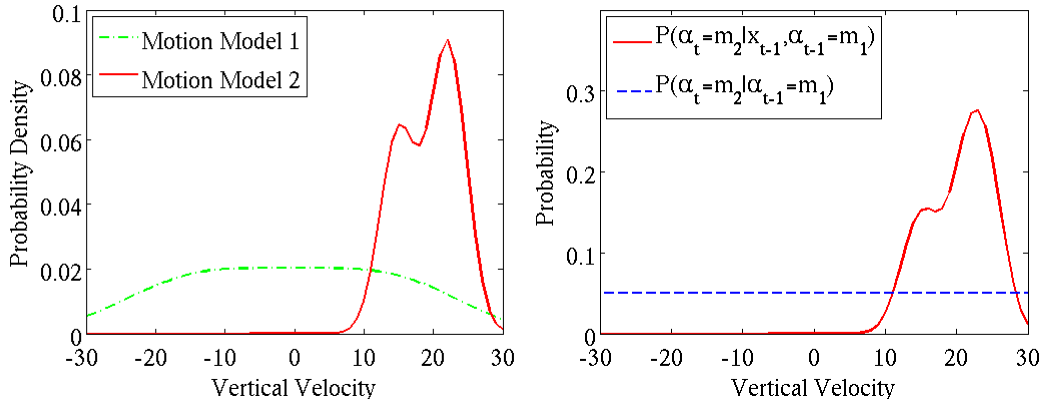
$$v_t^y = -v_{t-1}^y + w_{t-1}. \tag{2.25}$$

a is the constant acceleration rate estimated from the training set. For both motion models, the object maintains a constant velocity along the horizontal direction, $v_t^x = v_{t-1}^x + w_{t-1}$. Our proposed method embeds both motion models in the framework, and then it exploits the prior knowledge, in the model prediction function, that the faster the ball falls down the bigger probability it may switch to the second

motion model from the first one. This exploited knowledge is illustrated in Fig. 2.15(a) by the two individually estimated density distributions $p(x_{t-1}|\alpha_t = m_1)$ and $p(x_{t-1}|\alpha_t = m_2)$. In this case, m_1 and m_2 are the two motion models described in Eqs (2.24) and (2.25), and the object kinematic state x_{t-1} is the vertical velocity v^y_{t-1}. This shows that the kinematic state employed to provide the correlation information is not limited to object locations (as used in all the previous experiments) but is more general. By exploiting such correlation information, the probability of predicting the current motion model depends on the previous object kinematic state. Again, this is illustrated by Fig. 2.15(b) via a comparison against the original prediction function using a constant probability. In the following, two comparisons were conducted to demonstrate the contribution of each level of the proposed technique to JDT. Fig. 2.16(a) shows the performance comparison between SMMJDT (the JDT method with a single constant acceleration model) and MMMJDT. From this figure, we can see that with a single motion model, the existing JDT method fails to detect the soccer ball every time the ball bounces. Therefore, it is obviously important to use multiple motion models in JDT to detect objects that change motions. Fig. 2.16(b) shows the detection performance comparison between MMMJDT and Adaptive MMMJDT. The results show that with the prior knowledge about the motion model transition, a better detection performance is achieved by Adaptive MMMJDT because an additional piece of information is embedded in the motion model prediction function.

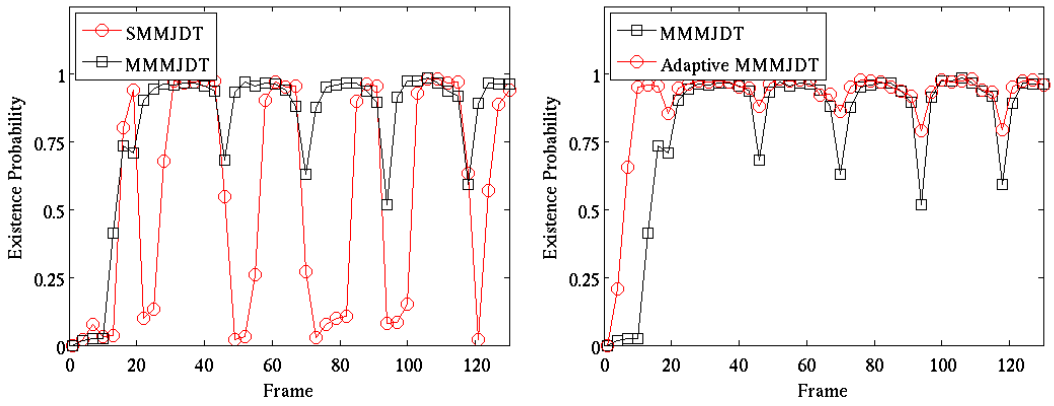
Figure 2.14: An example frame of the sequence *soccer*.

The third sequence *slide* contains 28 frames where a person slides down as



(a) (b)

Figure 2.15: The two estimated probability density distributions of the soccer ball's vertical velocity given the two potential motion models, $p(x_{t-1}|\alpha_t = m_1)$ (corresponding to the vertically acceleration model) and $p(x_{t-1}|\alpha_t = m_2)$ (corresponding to the vertically bouncing model). The dashed line corresponds to $p(x_{t-1}|\alpha_t = m_1)$ and the solid line corresponds to $p(x_{t-1}|\alpha_t = m_2)$. (b) The comparison between the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ and the original prediction function $P(\alpha_t|\alpha_{t-1})$, using the probability of the transition between the previous motion model m_1 and the current motion model m_2 as an example.



(a) (b)

Figure 2.16: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step) and SMMJDT (constant acceleration model JDT method) detecting the soccer in sequence *soccer*. (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

shown in a sample frame in Fig. 2.17. The person is highlighted by the yellow rectangle in the figure for visualization reasons. This sequence shows another example

where an object changes its motion from an acceleration model on the slide

$$v_t^y = v_{t-1}^y + a + w_{t-1}, \tag{2.26}$$

to a random walk model off the slide

$$\mathcal{X}_t = \mathcal{X}_{t-1} + w_{t-1}. \tag{2.27}$$

$\mathcal{X}_t = (x, y)_t$ represents the object location. Our proposed JDT method first embeds both models in the framework so that it can switch the system model when the person actually changes its motion. Then, to predict the motion model accurately, it exploits the prior knowledge that the person moves more likely in a specific motion model around certain locations. This prior knowledge can be learned following Eq. (2.12), and it is depicted in Fig. 2.18(a) by the two individually estimated density distributions $p(x_{t-1}|\alpha_t = m_1)$ and $p(x_{t-1}|\alpha_t = m_2)$. In this case, m_1 and m_2 are the two motion models described in Eqs (2.26) and (2.27), and the object kinematic state x_{t-1} employs the object location along the vertical axis. Similarly to the previous experiments, Fig. 2.18(b) shows the comparison between the new motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ that exploits the correlation information described above and the original motion model prediction probability $P(\alpha_t|\alpha_{t-1})$. Obviously, the former predicts the motion model adaptively to the object's state, while the latter does not. In the following, two similar comparisons, as previously, are conducted in this case. Fig. 2.19(a) demonstrates that embedding two motion models in JDT provides a better detection performance than using a single motion model. Three single motion models are employed in this comparison, constant velocity, constant acceleration, random walk. None of them was able to describe the actual object motion models as accurately as MMMJDT. Fig. 2.19(b) demonstrates that when prior knowledge on motion model transition is exploited, a slightly better detection performance is achieved especially after the person has changed his motion model around the 20^{th} frame as shown in Fig. 2.19(b). A relatively high estimated existence (detection) probability is *maintained* by Adaptive MMMJDT compared to MMMJDT.

Figure 2.17: An example frame of the sequence *slide*.

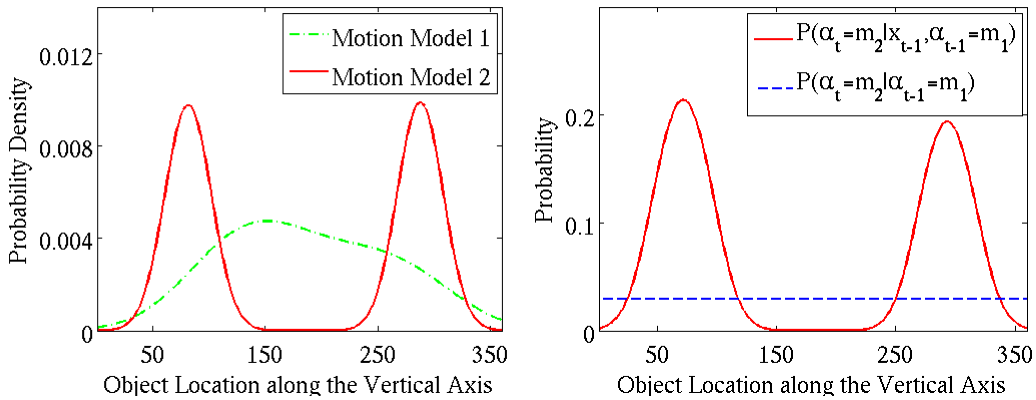


Figure 2.18: The two estimated probability density distributions of the object's vertical location given the two potential motion models, $p(x_{t-1}|\alpha_t = m_1)$ (corresponding to the random walk model) and $p(x_{t-1}|\alpha_t = m_2)$ (corresponding to the vertically acceleration model). The dashed line corresponds to $p(x_{t-1}|\alpha_t = m_1)$ and the solid line corresponds to $p(x_{t-1}|\alpha_t = m_2)$. (b) The comparison between the proposed motion model prediction function $P(\alpha_t|x_{t-1}, \alpha_{t-1})$ and the original prediction function $P(\alpha_t|\alpha_{t-1})$, using the probability of the transition between the previous motion model m_1 and the current motion model m_2 as an example.

2.5 Summary

An adaptive multi-motion model JDT method is proposed for detecting objects that may undergo sudden changes in motion. The proposed method tackles simultaneously the problem of detecting objects and handling objects' sudden changes in motion, which have been always treated individually in the literature. This is achieved by employing two sets of interacting models in an IMM estimator, one for handling

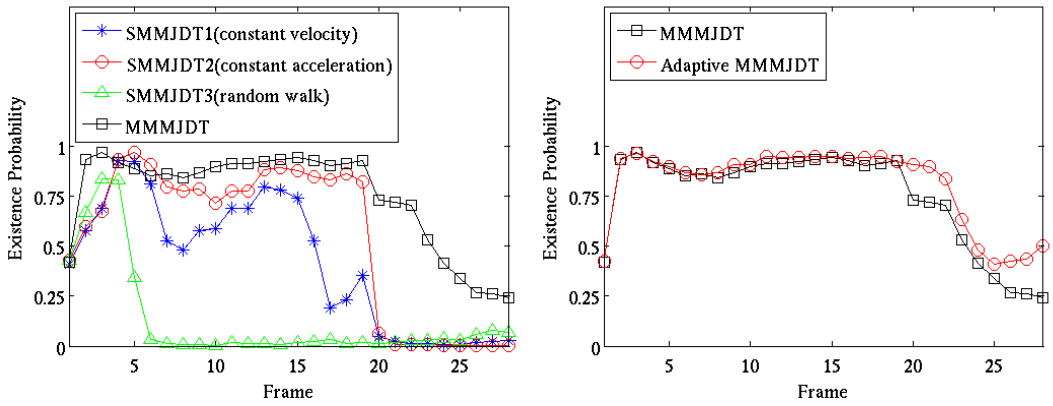


Figure 2.19: (a) Existence probability curves of MMMJDT (the JDT method with our first modification step) and SMMJDTs (three single motion model JDT methods) detecting the person in sequence *slide*. (b) Existence probability curves of MMMJDT and Adaptive MMMJDT (our final proposed JDT method).

varying number of objects and the other for handling variable motion models. The proposed method then exploits the correlation between the motion models and object kinematic state, never exploited in the existing IMM estimators. In this way, the prediction of the motion model is more accurate when the correlation information is available. Experiments on both synthetic and real examples illustrated that the proposed JDT method detects objects more accurately than the existing JDT methods when objects change their motions.

Chapter 3

Shape Based Appearance Model

In the previous chapter, we mainly investigated the motion model in the Bayesian recursive estimator to achieve an accurate object state prediction. Based on the predicted state, an appearance model can update it into posterior with new observation. Therefore, this chapter focuses on the appearance model in the Bayesian recursive estimator. Here we give an overview on the essential properties of the work in the previous and current chapters, so that we can clarify the connection between them as well as their connection with the Bayesian recursive estimation framework. The adaptive multi-motion model proposed in Chapter 2 is mainly designed for a JDT framework, which is a Bayesian framework estimating the varying number of present objects using a discrete variable in the object state. Instead, the appearance model to be proposed in this chapter is designed for a kernel based tracking framework, which is a Bayesian framework estimating the object state with a simple object representation. Therefore, the two novel techniques in the previous and current chapters are proposed respectively for these two specific frameworks, and therefore have distinctive assumptions and generalities. Specifically, the adaptive multi-motion model in the previous chapter is proposed within the JDT framework. However its object representation is not limited to a concise kernel format. The shape based appearance model in this chapter is proposed within the kernel based tracking framework. However, its object state does not require to contain the discrete variable for the purpose of joint detection and tracking. In spite of the different assumptions of the two proposed techniques, both of them together constitute the two critical components in the Bayesian recursive estimator.

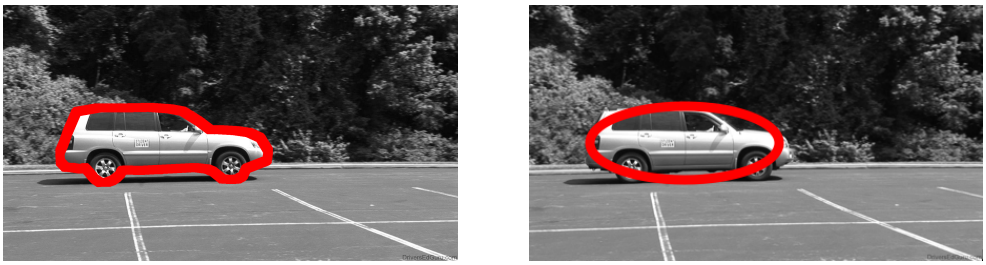
As mentioned above, the appearance model is a critical component in Bayesian recursive estimators, including JDT methods as well as many other detection and tracking methods. Appropriate design of this model plays an important role in detection/tracking performance. Specifically, we will propose an appearance model based on shape knowledge for kernel based trackers. A kernel based tracker typically tracks an object with a primitive geometric kernel, and then estimates the object state by fitting the kernel such that the appearance model is optimized. Most of the appearance models in kernel based tracking utilize the textural information within the kernel, although a few of them also make use of the gradient information along the kernel boundary. Interestingly, shape information of a general form has never been fully exploited in kernel tracking, despite the fact that shape has been widely used in silhouette tracking at the cost of intensive computation. Therefore, in the following we propose an original way to incorporate shape knowledge into the appearance model of kernel based trackers to significantly improve their tracking accuracy, and in the meanwhile preserve their computational advantage versus silhouette based trackers.

3.1 Introduction

Object tracking consists of following moving objects through an image sequence. It is a challenging problem in computer vision which serves various applications such as vehicle navigation, medical image processing, and video analysis [102].

To locate a given object from one image to the next in a sequence, the current trackers are generally based on the photometric characteristics or shape of the object. The photometric appearance is used assuming that the object has a profile which is distinctive against the background [34, 73]. Differently, shape based trackers follow the object having a given geometric description modulo some deformations [27]. Other visual cues have also been used to assist tracking such as 3D object views, but photometric characteristics and shape constraints remain the most useful and they determine the efficiency of the tracker considerably. Besides the efficiency concern, there are also other practical needs which determine considerably

the framework adopted for designing the tracker. In some applications, tracking consists of following the object grossly placed about a mean position [23, 102] and only some bounding figures, such as rectangles or ellipses, are used to circumscribe the moving object. However this is insufficient for other applications where accurate object boundary is rather needed to identify the object, classify it, or interpret its behavior [27, 102]. As a result, two major groups of tracking methods, silhouette based and kernel based, have been proposed depending on their specific advantages and on the targeted applications. Silhouette based trackers employ rich but inefficient contour models to estimate accurate object shape as shown in Fig. 3.1(a), while on the contrast kernel based trackers employ rough but efficient primitive geometric kernel models to estimate the object's basic information, e.g., location and scale as shown in Fig. 3.1(b).



(a) (b)

Figure 3.1: (a) An example of a silhouette based tracker estimating the object boundary. (b) An example of a kernel (ellipse in this case) based tracker estimating the object state. Silhouette based trackers are mainly used when the exact object boundary is required, which implies a high computational load. Kernel based trackers are rather used when only basic object information (location and scale, etc) needs to be estimated, which generally requires less computations.

In silhouette based tracking, the object contour is generally described by a closed curve which encloses the region corresponding to the photometric and/or shape constraints. The variational framework has been the most convenient for this kind of methods [73, 61, 27], where the problem constraints are embedded in an objective functional whose minimum corresponds to positioning the curve by the region of interest boundaries. The shape constraint has been intensively used to assist tracking by active contours. In [17], a Fourrier-based shape prior model is used to constrain the deformation of a classic active contour (snake). In [27], it

has been demonstrated that object tracking can be improved considerably when a model of the moving object boundary shape is available. Principal component analysis (PCA) is used to learn a geometric shape description of the moving object from a training set of various object outlines. The principal components are systematically embedded in a shape prior of an active curve/level set formulation. The tracking process updates continuously the shape prior based on the previously observed contours. Moreover, shape priors have been also used in [103] for multi-object tracking but it has been activated only when occlusion is detected. The tracker is mainly based on the appearance term which acts when there is no occlusion between moving objects. Meanwhile, the shape prior model is updated using the observed contours in the tracking process. All the above algorithms pose the silhouette tracking as a minimization problem, and the minimization of such functionals relies on gradient descent. As a result, the algorithms converge to local minima, can be affected by the initialization and, more importantly, are notoriously slow in spite of the various computational artifacts which can speed their execution [86]. To overcome these problems, another group of algorithms pose the silhouette tracking as an inference problem, and the goal is to estimate the posterior density of a contour given image observations. In this context, the shape information constrains the object model to transform only in ways characteristic of the object of interest. For instance, in [44, 76], an object is represented with an active shape model that undergoes constrained transformation defined by the shape prior information. While all the aforementioned shape models are shown to drastically improve the tracking performance, they still suffer from the inefficiency problem due to the high dimensional solution space, and they are of no consequence where the object need not be identified nor its behavior interpreted based on its shape. Practically, the inefficiency in terms of execution load of these silhouette based tracking methods is the major impediment in many applications, particularly those which require real time processing.

In contrast, kernel based trackers have been widely adopted because of their relative simplicity and low computational cost. The purpose is to estimate the object state efficiently by computing the motion of the object and modeling it with a

primitive geometric shape. In this framework, many efficient and accurate methods have been proposed. These methods differ mainly in terms of their appearance representations which did not fully exploit shape knowledge due to the lack of the information of object boundary. Most of the cues that have been used in this framework are based on regional information, and some of them on boundary information. Among the regional based cues, color histogram might be the most popular. It is scale and rotation invariant and it can handle occlusion to a certain extent (partial occlusion) [52, 41, 74]. As global statistic information, color histogram does not provide discriminative localization. In addition, there are also models based on histograms of edge orientation [12, 52] and motion [84, 74]. These two cues are quite sensitive to clutter. Similarly, texture has also been used as a cue in the appearance models of [11] and [12], which are based on wavelet and steerable pyramid, respectively. Another approach to modeling the object appearance is using PCA to build a subspace representation from a set of templates [84, 81, 53]. SIFT (Scale Invariant Feature Transform) descriptors have also been used as features in appearance models in [16, 56] to increase the discriminative ability. All the regional based cues mentioned above are computed from some manually designed statistics. There are also a few papers which use machine learning methods to learn appearance models directly from training sets. For example in [2], the authors use a support vector machine to learn the appearance model from a set of templates, and in [3] the authors learn the appearance model via Adaboost from a set of Haar-like features. All the existing regional cues used in the above works are based on the information within the kernel (not the object) boundary that contains absolutely no shape information.

Perhaps the closest works to our shape cue are [31, 55, 54] that try to utilize the kernel boundary information although still in a different level from shape information. In [31], the authors use a boundary cue measured by the distance between kernel's contour pixels and their closest edge points. Similarly, in [55], a hypothesized object kernel's boundary cue is measured according to the amount of edge pixels near the kernel's contour. To be more accurate, based on the observation that object boundaries are usually perpendicular to the orientation of the projected object edges, the boundary cue is computed in [54] by rewarding the gradients with

the tangent direction to the kernel contour, which are in close proximity to the kernel contour. Although our proposed shape cue is similar to the above three works in the sense that all of them try to extract information along the kernel boundary, our work differs and excels in the following ways:

- The proposed shape cue embeds high level prior knowledge. Our shape constraints bias the kernel tracker towards objects meeting specific shape characteristics. However, by estimating the gradient along the kernel, the above mentioned methods assume implicitly that the object boundaries coincide with kernel boundaries, which is generally not true.

- The shape is described in a general form. Indeed, the description is versatile enough to allow various constraints on the shape from being linearly distributed around a mean shape to being restricted to some geometric forms such as concavity and convexity.

Additionally, the kernel based trackers enhanced by the proposed shape appearance model can still maintain the computational advantage compared to the shape benefited silhouette trackers, and they can also handle partial occlusion problem to a certain extent.

The rest of this chapter is organized as follows. In Section 3.2, we first present the general kernel based tracking framework, and then for this framework we propose an appearance model to utilize shape information in a novel way. Section 3.3 includes the experimental results that demonstrate the properties of the proposed shape appearance model in different ways. Finally the work is summarized in Section 3.4.

3.2 Kernel Based Tracking

Since the shape based appearance model is proposed to be integrated into a kernel based tracker, we first give an overview of the kernel tracking framework and then describe, in details, the proposed appearance model. A kernel based tracker generally represents an object with a primitive geometric kernel, such as rectangle

or ellipse, whose object state at time t, X_t, is defined by a small set of parameters concisely. Tracking consists of estimating the object state at the current time t given all the observations up to time t referred to by $Z^t = \{Z_1, \ldots, Z_t\}$. The posterior probability density function (pdf) of the object state is generally formulated in a Bayesian form,

$$p(X_t|Z^t) \propto p(Z_t|X_t) \int p(X_t|X_{t-1}) p(X_{t-1}|Z^{t-1}) dX_{t-1}, \tag{3.1}$$

where $p(X_t|X_{t-1})$ is the state prediction model, and $p(Z_t|X_t)$ is the appearance model. The key contribution of the work in this chapter is to embed prior shape constraints in the latter conditional probability. Tracking an object of interest over a sequence of images with the proposed appearance model can be done by estimating the current object state which maximizes the posterior pdf $p(X_t|Z^t)$ as follows:

$$\hat{X}_t = \arg\max_{X_t} p(X_t|Z^t). \tag{3.2}$$

3.3 Shape Based Appearance Model

An appearance model defines the likelihood of associating an observation with a given particular object state. Therefore, in the following we will first describe how the shape constraints corresponding to an object state are extracted. Then we will formulate the proposed appearance model embedding these constraints.

3.3.1 Shape Observation

Given a candidate kernel (an ellipse in this case as depicted in Fig. 3.2) associated with a potential object, a set of N points are evenly sampled along the contour of the kernel, and a normal line is drawn from each sampled point. As shown in Fig. 3.2, s_n is the n^{th} sampled point along the kernel contour and l_n is the normal drawn from s_n. Along each normal line, 1-D edge detection is performed as in [59] and only the most prominent edge point is extracted (refer to the red dots in Fig. 3.2). Consider the normal line l_n in Fig. 3.2 as an example. There are more than one edge pixels on l_n: z_n^1 and z_n^2. Only z_n^1 is extracted, supposing that it is the strongest edge pixel on l_n.

Although this procedure may extract some points which do not correspond to the real object edge (such as the point e in this example), most of the extracted points coincide generally with the object boundary [73, 49, 44]. This claim is based on the assumption that the targeted object presents boundary segments which are distinctive from the nearby background, i.e., the image gradient is relatively high along the object boundary. When this assumption holds, the most prominent extracted points likely correspond to boundary points. In some spots of the target, the boundary point may not be the most prominent and thus, another non-boundary point might be mistakenly extracted (such as the point e). However, this does not happen frequently and does not affect the shape representation in most of the cases. Indeed, assuming that edge points correspond generally to high intensity gradients is common. In [73, 49], for instance, object segmentation/tracking is performed using the active contours/snakes which evolve towards the target based on the stopping function. This function depends only on the *intensity gradient* by assuming that the object boundary corresponds to high values of intensity gradients (very small values of the stopping function). In [44], a similar 1-D high contrast search is performed along the spline curve normals. The hypothesized shape is represented as a parametric spline curve. Similarly, many relevant edge detectors make the same assumption regarding the edge points (Canny, Sobel, Prewitt, etc.). Therefore, the object boundary can be fully determined by the extracted edge points, and represented using a radius vector function as follows,

$$r_n = \|z_n - \hat{z}\|_2, n \in \{1, ..., N\}. \tag{3.3}$$

$$\hat{z} = \frac{1}{N}\sum_{n=1}^{N} z_n. \tag{3.4}$$

r_n indicates the distance between each extracted edge point (boundary point), z_n, and the shape centroid, $\hat{z}$. Then, the normalized (scale-invariant) N-dimensional radius vector,

$$r(X) = \frac{[r_1, ..., r_N]^T}{\sum_{n=1}^{N} r_n/N}, \tag{3.5}$$

is extracted as the shape observation corresponding to the object state X.

The obtained radius vector is ordered starting from the radius corresponding to the sampled boundary point in the direction of the ellipse orientation θ, assuming

the elliptical kernel is configured by the object state,

$$X = [x, y, a, b, \theta]. \tag{3.6}$$

(x, y) is the center, a and b are the lengths of the two principal axes, and θ is the orientation of the major axis. This starting point corresponds to a unique end of an ellipse axis. For example in Fig. 3.2, r_1 corresponds to the sampled point s_1. Then we traverse the sampled boundary points clockwise, so that the radius vector is always aligned. Due to this alignment, the proposed method avoids the need to an explicit one-to-one correspondence between the points. These kind of correspondances are rather present in snake-based tracking where the evolving contour is determined by a set of points which need an inter-frame correspondence. The reason is that the proposed method poses tracking as an inference problem implemented using a particle filter. Hence, each potential candidate object (corresponding to a candidate ellipse) is sampled independently in its actual frame. The only correspondance that exist is rather implicit because the points keep always the same index in the state vector, and this is straightforward because of the alignment.

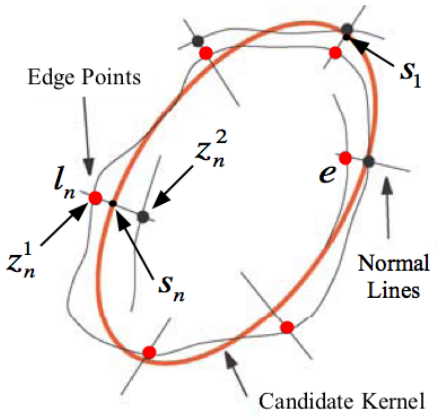


Figure 3.2: Illustration of extracting the shape observation

3.3.2 Shape Cue

Once the shape observation $r(X)$ is extracted, then the similarity between $r(X)$ and the pre-learned shape prior model can be used as a shape cue by the appearance model in kernel tracking. We define this cue in a general form as a dot product of

two vectors,

$$f(X) = En(r(X)) \cdot w, \tag{3.7}$$

where $En(r(X))$ is an energy vector computed from the shape observation $r(X)$ that indicates the shape energy at each extracted boundary point. The lower a certain energy vector element is, the more the corresponding boundary point satisfies the shape constraint. w is a weighting vector which assigns different weights to boundary points to obtain useful attributes. Therefore, the proposed shape cue is a weighted sum of all the extracted boundary points' shape energies. In the following we will discuss the meanings of the energy vector $En(r(X))$ and the weighting vector w in more details through some examples.

The energy vector $En(r(X))$ can be designed in different ways to extract the shape constraints. Many shape feature analysis techniques such as PCA and Fourier-based model can be used to define this energy vector. In the following, we show different ways of defining the energy vector $En(r(X))$ each of which leads to a specific shape constraint.

PCA reconstruction – Given a set of p training shapes represented with normalized radius vectors, one can learn the mean shape of all the training samples $\overline{r}$ and the eigenvectors of the covariance matrix of shape radius vector denoted by $U_{N\times p}$. The first m columns of $U_{N\times p}$ are the m eigenvectors corresponding to the first m principal components of the training shapes. Any new shape observation $r(X)$ can be projected into the learned PC-subspace composed of only these m eigenvectors:

$$\hat{r}(X) = \overline{r} + U_{N\times m}(U_{N\times m})^T(r(X) - \overline{r}). \tag{3.8}$$

Then the energy vector $En(r(X))$ can be defined as the absolute difference vector between the extracted shape observation and its reconstruction. This absolute difference vector is computed on element-by-element basis. Specifically, each element in the energy vector, $En(r(X))_n$, is computed as

$$En(r(X))_n = \|r_n(X) - \hat{r}_n(X)\|, \tag{3.9}$$

where $\|\ .\ \|$ is the norm applied independently to the scalar components of the difference vector and, hence, can be either the L1 or the L2 norm. $r_n(X)$ is the

n^{th} element (normalized radius) in the shape observation vector $r(X)$. $En(r(X))_n$ indicates the shape energy on the n^{th} extracted boundary point. By choosing the weighting vector w as a uniform vector with the same length as $r(X)$, the shape cue $f(X)$ defined in Eq. (3.7) becomes equal to the sum of the energy vector $En(r(X))$ components which indicates the total reconstruction error. If the shape observation $r(X)$ and the shape prior belong to the same class, then this reconstruction error shall be small [48]. This can be interpreted intuitively as follows: the uniform weighting means that all the boundary points are assigned equal importance. Thus, minimizing the sum of the individual energies, amounts to minimizing equally each component of the energy vector $En(r(X))$.

Geometric constraints – The PCA shape cue based on Eq. (3.9) measures how close a candidate object's boundary is to a unique shape shared by the class of objects of interest. In this paragraph, however, the shape cue is intended to measure the likelihood that a candidate object's boundary has a given geometric constraint such as smoothness and convexity. Consequently, the energy vector $En(r(X))$ is defined as follows

$$En(r(X)) = \frac{abs(D^2(r(X)))}{S(r(X))}, \tag{3.10}$$

where $D^2(r(X))$ is the discrete second order difference of the normalized radius vector. Each element in $D^2(r(X))$ is computed as $D^2(r(X))_n = r_{n+1}(X) - 2r_n(X) + r_{n-1}(X)$. $abs(D^2(r(X)))$ computes the absolute values of the components of $D^2(r(X))$, so that we only focus on the magnitude of the second order difference of the radius vector. This energy term $abs(D^2(r(X)))$ can be interpreted in a similar way as the active contour's internal energy term [14], which is generally associated with the smoothness of the evolving active contour. The second energy term $S(r(X))$ is the solidity of the object boundary corresponding to $r(X)$, and it captures the convexity of the object shape. $S(r(X))$ can be computed as,

$$S(r(X)) = \frac{A_o(r(X))}{A_h(r(X))} \tag{3.11}$$

where $A_o(r(X))$ is the area of the object region specified by the object boundary representation $r(X)$, and $A_h(r(X))$ is the convex hull area of the same object. The detailed steps for computing $A_o(r(X))$ and $A_h(r(X))$ have been omitted here and

can be found in [89]. Similar to the previous PCA based cue, a uniform vector is chosen as the weighting vector w here, and different weighting vectors with other effects will be used later. The shape cue based on the energy vector $En(r(X))$ in Eq. (3.10) is useful to distinguish smooth boundaries in convex shapes from others. Therefore it can be adopted to distinguish some natural objects from background [87] such as lumps in the large lump detection (LLD) problem which will be discussed in detail in the experimental section. In this kind of situations, objects belonging to the class of interest are not normally distributed around a mean shape to be captured using the shape cue based on Eq. (3.9). Instead, they have a certain common global shape property which is rather characterized by cues such as the one based on Eq. (3.10).

The two examples of energy vectors presented in Eqs (3.9) and (3.10) have been adopted in our experiments. Besides, as previously outlined, other shape descriptors can also be used to define energy vectors according to the available shape constraints in the targeted applications. Therefore, the formulation in Eq. (3.7) is rather general and flexible, and any energy vector can be embedded systematically.

Besides the flexibility with the energy vector $En(r(X))$, the weighting vector w can also be defined in various ways depending on the importance to be assigned to the different parts of the extracted boundary. In the following, we define a weighting vector for the cases where a part of the object boundary is occluded or presents weak intensity gradients. For $n \in \{1, ..., N\}$,

$$w(n) = \begin{cases} 1 & \text{if} \quad \alpha - B/2 < n < \alpha + B/2 \\ 0 & \text{else,} \end{cases} \tag{3.12}$$

is a rectangular window function determined by the window center α and the window width B. With this weighting vector, the shape cue is only enabled/computed on the portion of the extracted object boundary overlapping with the window. Hence it is possible to omit the part which might be occluded or intrinsically presents very weak gradients. For instance, in Fig. 3.3(a) a proper candidate kernel is presumably located on the artificially occluded fish. From this candidate kernel, the boundary points (the red dots in the figure) are first extracted as explained before. Then, the corresponding PCA reconstruction, based on the learned shape prior, is represented

in Fig. 3.3(a) by the green stars. Obviously, the boundary of the occluded fish head is not extracted correctly, but the tail part matches the shape prior well. In such cases, the role of the weighting vector w becomes more important and can be easily understood. For instance, the mask defined in Eq. (3.12) can be used to enable the shape cue exclusively on the non occluded (tail) part of the fish. Fig. 3.3(b) shows the radius vector of the extracted boundary ($r(X)$ represented by the red dots) and the corresponding PCA reconstruction ($\hat{r}(X)$ represented by the green stars). As stated in Eq. (3.9), the energy vector $En(r(X))$ is consequently defined as the absolute difference between these two vectors. The suitable weighting vector w corresponds to the window between the two vertical dashed lines in Fig. 3.3(b). To determine this weighting vector, the width of the window, B, is fixed in advance as a proper size of the portion of the boundary on which the shape cue is enabled. However the window center α has to be computed dynamically by examining all the possibilities. This can be implemented by convolving the energy vector $En(r(X))$ with a one dimensional mean filter h having the same length as the window width B. Then the location which corresponds to the smallest response indicates the expected window center α. The reason is that α locates the boundary part that matches the shape prior best and excludes the part not extracted correctly due to either occlusion or intrinsically weak gradients. α is determined as the index $k \in \{1, ..., N\}$ satisfying

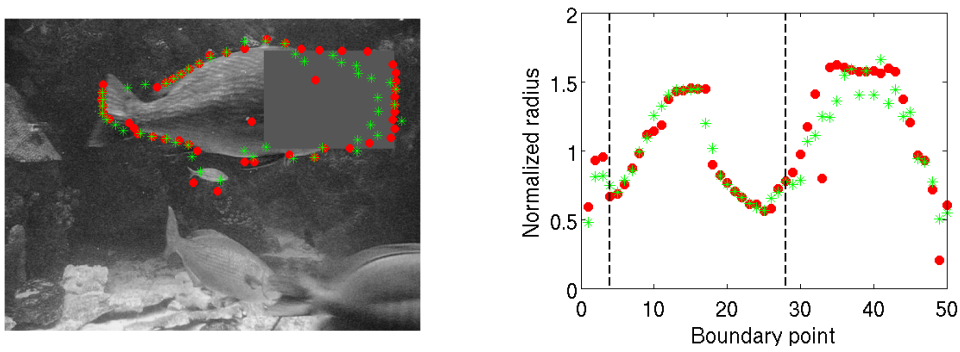
$$V_k = \min_{n=1:N} \{V_n\},$$
$$V = En(r(X)) \otimes h. \tag{3.13}$$

V is an N dimensional vector, and it is the convolution of the energy vector $En(r(X))$ with the one dimensional mean filter h.

3.3.3 Appearance Model

Once the shape cue $f(X)$ in Eq. (3.7) is defined, an appearance model based on it can be built systematically as an exponential distribution,

$$p(Z|X) \propto exp(-\lambda \cdot f(X)), \qquad f(X) \geq 0. \tag{3.14}$$



(a) (b)

Figure 3.3: (a) An example of partial occlusion. The red dots show the boundary extracted as the shape observation from a candidate kernel on the fish. The green stars show the PCA reconstruction of this shape observation in the subspace learned from the training set. Only the tail part with better extracted boundary is expected to be used to compute the shape cue by applying a proper weighting vector. (b) The corresponding normalized radius vectors of the extracted boundary and its reconstruction in (a). The two vertical dashed lines indicate the window of the weighting vector w which locates on the portion of the radius vector corresponding to the tail boundary with the minimum reconstruction error.

λ is a tunable parameter, and $f(X)$ is the shape cue determined by the employed energy vector. As we mentioned before, different energy vectors embedding various shape constraints can be defined to obtain different shape cues. Multiple energy vectors are also possible to be integrated in the shape cue to convey the combined shape constraint. In this thesis, we do not investigate the combination problem of multiple energy vectors. Instead, we only use either one of the two shape energy vectors proposed in Eqs (3.9) and (3.10) in our experiments. By introducing this shape prior information, the proposed appearance model is intended to differentiate objects from the background when the regional features and boundary gradient features are not sufficient.

Note that the purpose here is to establish the proposed shape cue as a complementary model to be embedded into the appearance models of current kernel trackers. A combination of such a cue with the current models should improve the performance of the existing trackers in difficult cases as discussed in this section. The proposed shape descriptor is integrated in the kernel tracking framework in a structured manner which makes the method flexible and easily generalized. Indeed,

starting from a simple kernel (ellipse) and as explained in the previous sections, the shape descriptor is built through three main steps, each of which is able to feed various shape descriptors. First, the object boundary points are *detected* using image gradient information. This set of points can be used to construct a shape contour and apply any contour based shape descriptor [66]. Second, the edge points are exploited in order to define the *shape signature* using radius vectors as in Eq. (3.5). Again, a variety of point-wise shape signatures are available such as curvatures, tangent angles, chain codes, etc. Finally, the shape cue is formulated by embedding the point-wise shape signature in a global measure as in Eq. (3.7). This cue is formulated in order to encode implicitly the tracking constraints arising mainly from the application at hand. For instance, Eq. (3.9) embeds the shape signature in order to reconstruct a prior shape using PCA. However, in Eq. (3.10), the shape signature is rather utilized to impose a geometric constraint on the shape contour (convexity in our case). Other than PCA, various techniques may be used for encoding the shape cue such as Fourrier descriptor [89], wavelet descriptor [65], and curvature scale space [67]. Moreover, other than convexity, geometric constraints can also be imposed on the shape contour such as eccentricity, circularity, rectangularity, elongation, and orientation [89]. This flexibility is especially demonstrated through the experiments with the large lump detection application detailed in the experimental section. The lumps present in both training and test images are very different in terms of shape and size. However, in addition to photometric aspects, they obey to a certain geometrical shape constraint. By enforcing such shape constraint, as in Eq. (3.10), better results have been reached. This is not the case in other shape frameworks such as the Active Shape Models (ASM) [24] or the dynamical statistical shape priors [27] where the actual object's shape needs to be reconstructed as a linear combination of the principal components extracted from the training set. This, of course, assumes that the shapes are only allowed to slightly deform around a mean shape which is not the case in many real applications such as the large lump detection problem.

3.4 Experimental Results

Recall that the intended purpose of the proposed method is to afford a shape cue which provides complementary shape information to the existing appearance models used for kernel tracking. Therefore, the purpose of the following experiments is to demonstrate the following two aspects. First, the proposed shape description improves the tracking accuracy of kernel based trackers. Secondly, it preserves the computational efficiency of kernel based trackers compared to silhouette based trackers. To this end, we have two types of validation tests as follows.
(1) Comparative study:

- Computational load comparisons against silhouette based trackers.
- Tracking accuracy evaluations and comparisons against several kernel based trackers on different test videos. The performance is evaluated in terms of the accuracy of the object's state estimation, not the object's boundary details which are specific to silhouette based trackers.

(2) Extensive quantitative study with the large lump detection application, using more than $64,000$ images.

Additional experiments will also be given in order to investigate the behavior of the proposed method in presence of partial occlusion, with different numbers of sampled boundary points, and with different sizes of initialized elliptical kernels. Except when explicitly mentioned, all the experiments use the proposed shape cue based on PCA reconstruction in Eq. (3.9). In each experiment using PCA reconstruction, the PCA model including the mean shape and principal components is learned on a subset of frames from the corresponding test sequence. Indeed, one fifth of the total number of test frames are used for training. The number of contour points N is fixed to 50, and the number of the principal components m used in the PCA reconstruction is set to 15 for all the related experiments. The effect of such parameters on the tracking results is discussed later. The proposed algorithm, as well as the competing methods are all implemented using Matlab 2009 and run on a PC with 3 GHz Intel CPU.

3.4.1 Computational Efficiency

As outlined previously, this subsection aims to investigate the computational efficiency of the proposed shape based kernel tracker with comparison to silhouette based trackers. Recall that silhouette tracking is employed when the complete boundary of an object is required. This becomes more important when dealing with complex shapes and non rigid objects which cannot always be adequately described by the basic geometric shapes. Despite this, people still may use kernel trackers mainly for their computational efficiency rather than silhouette trackers. In our case, the purpose is to demonstrate that we still remain efficient over silhouette trackers even by embedding shape constraints into the observation model of our kernel trackers.

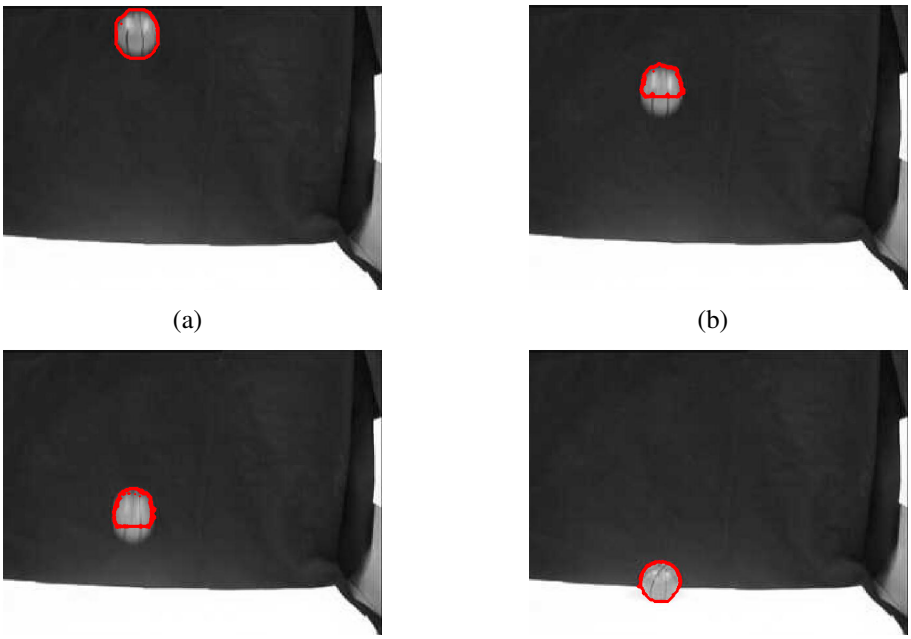


Figure 3.4: A sample of the results by the silhouette tracker [61] with the basketball sequence. (a) Frame 1. (b) Frame 3. The silhouette tracker with a maximum inter-frame offset of 15 pixels fails to follow the ball and loses the track thereafter. (c) Frame 5: the silhouette tracker with a maximum inter-frame offset of 25 pixels fails to follow the ball and loses the track thereafter. (d) Frame 60 (last frame): the silhouette tracker with a maximum inter-frame offset of 35 pixels succeeds to follow the ball until the lend of the sequence.

The first experiment consists of a video showing a basketball falling down with

a variable inter-frame offset. The maximum offset of the basketball between any two consecutive frames is around 35 pixels which is larger than most of the cases that silhouette tracking methods deal with [102]. Four frames among a total of 60 frames are depicted by Fig. 3.4. Besides the proposed algorithm, we run the level set implementation of the silhouette trackers of Mansouri [61] and Freedman et *al.* [34]. The method of Mansouri uses active contours formulated in a variational way. For each pixel of the object region, a flow vector is computed within a neighborhood. Using these flow vectors, an energy evaluated based on the brightness constancy constraint is iteratively minimized along with the active contour evolution. The unique a priori fixed parameter is the largest possible inter-frame offset which determines the maximum range of motion between any two consecutive frames. In this example and for comparison reasons, this parameter is varied from 15 to 35. Similarly, the method of Freedman et *al*. iteratively minimizes the mismatch between the actual intensity distribution and a prior model of the target object. The only tunable parameter is the maximum allowed iterations number for the level set function update. In this example, it is varied between 2000 and 3000 for comparison reasons.

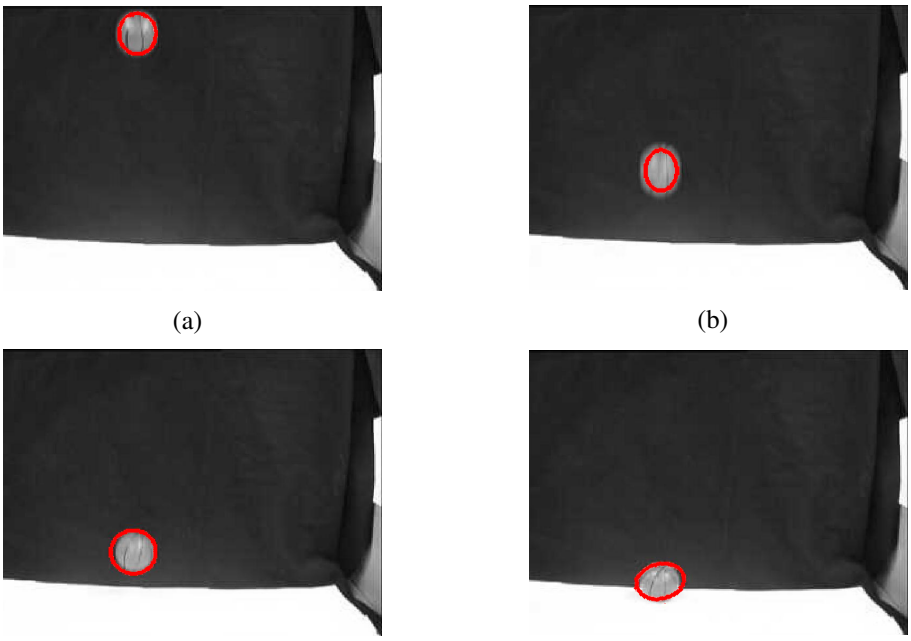
Table 3.1 reports the average computing time (mean and standard deviation) of the proposed tracker against two silhouette trackers (referred to by ST1 and ST2) using different values of their parameters. The sequence at hand in this experiment presents two main difficulties. First, the object of interest (the ball in Fig. 3.4) moves with a variable velocity so that any motion constancy assumption is not valid. Second, the contrast along the moving boundary is weak, particularly when the ball rebounds from the floor. As shown in Table 3.1, the proposed method (referred to by SKT) clearly outperforms the silhouette trackers in terms of computational time. The tracking is performed in less than one second for all the sequence frames. A sample of the obtained tracking results, depicted in Fig. 3.5, show that our method succeeds in tracking accurately the moving object through the entire sequence. In contrast, the silhouette tracker ST1 [61] has an average computational time of more than 6 seconds in the best case where the inter-frame offset is set to 15 pixels. In this specific case, ST1 loses the target in the third frame because the actual inter-

Table 3.1: Computational time comparison between our shape based kernel tracker (SKT) and the silhouette trackers by Mansouri (ST1) [61] and Freedman et al. (ST2) [34] using different values of parameters on the basketball sequence.

Method	SKT	ST1(15)	ST1(25)	ST1(35)	ST2(2000)	ST2(3000)
Time (s)	0.7 $\pm$0.1	6$\pm$0.5	14$\pm$1.2	23$\pm$0.7	8$\pm$0.8	14$\pm$0.7

frame displacement is larger than 15 pixels. Similarly, ST1 loses the target by the fifth frame when we set the maximum inter-frame offset to 25 pixels as shown in Fig. 3.4(c). Obviously, this requires more computational time because the search is performed in a larger neighborhood. This tracker succeeds in keeping track of the moving object only when the offset is set to the maximum inter-frame motion-35 pixels for this sequence-as depicted by Fig. 3.4(d). This case corresponds to the highest computational time recorded by ST1 on the sequence at hand as reported in Table 3.1. Similarly, although the silhouette tracker ST2 [34] is faster than ST1, it is still slower than our kernel tracker. For instance, when the maximum number of allowed iterations is set to 2000, ST2 takes around 7 seconds per frame on average. However, similarly to ST1, it fails to track the basketball successfully. Only when the maximum number of iterations is increased to 3000, ST2 tracks the basketball through the whole sequence. However this requires an average of about 14 seconds per frame.

An additional experiment is conducted on the *taxi* sequence using the same three competing methods as shown in Fig. 3.6. Similar results to the previous experiment have been noticed as reported in Table 3.2. The proposed method, SKT, performs the tracking successfully in less than 0.2 seconds per frame. It clearly outperforms both silhouette trackers in terms of computational time. The silhouette tracker ST1 is run using three different values of the inter-frame offset parameter (1, 5, and 10 pixels). With the first two values, ST1 fails to track the taxi over the whole sequence, and succeeds in doing that only when that parameter is set to 10 pixels. However, the corresponding computational time evaluated to 4.3 seconds, is higher than that recorded by the SKT. Similarly, the silhouette tracker ST2 is run using two values of the maximum iterations number (25 and 100 iterations). Within 25



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

Figure 3.5: A sample of the results by our shape based kernel tracker with the basketball sequence. (a) Frame 1. (b) Frame 5. (c) Frame 30. (d) Frame 60 (last frame). The proposed tracker succeeds to keep track of the ball accurately over the whole sequence.

Table 3.2: Computational time comparison between our shape based kernel tracker (SKT) and the silhouette trackers by Mansouri (ST1) [61] and Freedman et *al.* (ST2) [34] using different tuning parameters on the taxi sequence.

Method	SKT	ST1(1)	ST1(5)	ST1(10)	ST2(25)	ST2(100)
Time (s)	0.15 $\pm$0.01	2.1$\pm$0.03	2.7$\pm$0.3	4.3$\pm$0.4	1.2$\pm$0.02	2.2$\pm$0.1

iterations, ST2 failed in following the taxi successfully through the entire sequence. When this parameter is set to 100 iterations, the tracker succeeded in keeping track of the taxi. However the computational time per frame increased to more than 2 seconds, which is higher than what the SKT has recorded.

3.4.2 Tracking Accuracy

In this subsection, we investigate the accuracy of the proposed shape based tracker against other similar kernel based trackers. In the following comparative studies, the proposed appearance model is compared to other existing appearance models

Figure 3.6: An example frame of a sequence containing a taxi being tracked by ST1 [61].

using the same basic kernel tracker implemented in Section 2.3. The purpose is to show that the proposed shape cue helps in improving tracking results especially in specific cases where the existing cues are ambiguous. In the following comparative appraisal, we apply the proposed method with four state-of-the-art kernel trackers on three different sequences: *face, cup* and *pedestrian*. The first competing tracker employs an appearance model based on intensity histogram as in [78], and the others utilize appearance models embedding different boundary gradient descriptions as in [31, 55, 54]. In more details, the intensity histogram based appearance model can be described as

$$p(Z|X) \propto exp(-\lambda \cdot f_{int-hist}(X)), \qquad f_{int-hist}(X) \geq 0. \tag{3.15}$$

$f_{int-hist}(X)$ measures the distance between the histograms of the template and the observation extracted from the region specified by the object state X. Therefore the kernel tracker based on this appearance model will prefer the regions or objects with similar intensity histogram profiles as the template. Similarly, the gradient based appearance models can be described as

$$p(Z|X) \propto exp(\lambda \cdot f_{grad}(X)), \qquad f_{grad}(X) \geq 0. \tag{3.16}$$

$f_{grad}(X)$ measures the gradient strength within the vicinity of the kernel boundary specified by the object state. Therefore, the kernel trackers employing the three gradient based appearance models favor the regions or objects presenting strong gradients. These trackers differ from each other in the way they measure the gradi-

ent strength along the kernel boundary (as described in the introductory section of this chapter).

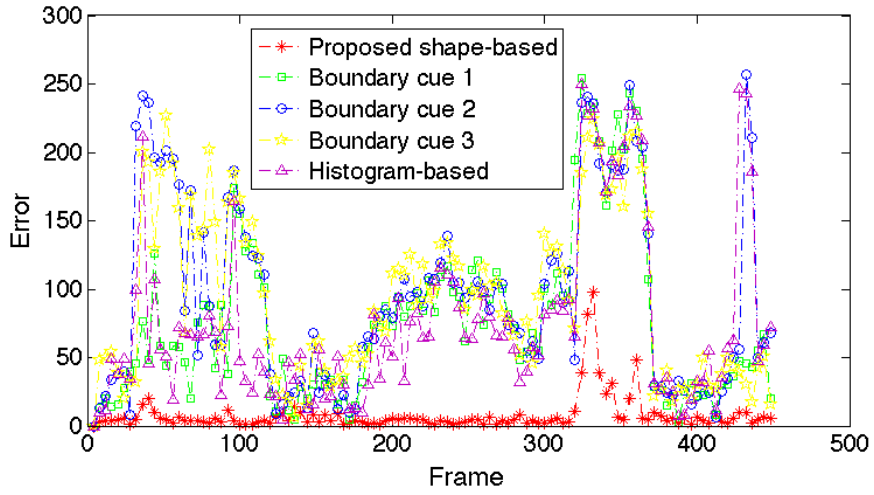
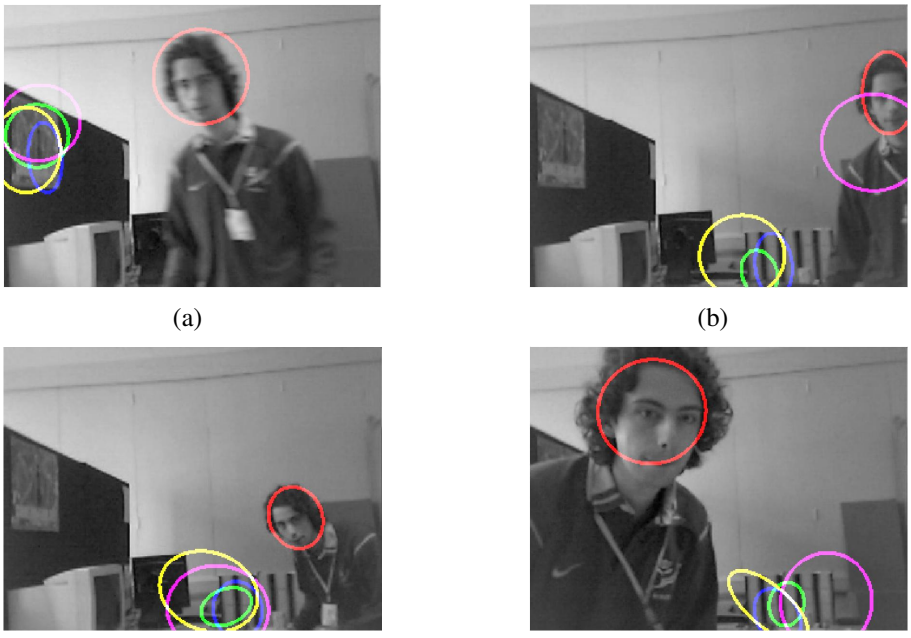


Figure 3.7: Comparison of the tracking errors recorded by the five competing trackers with *face* sequence. The error is defined as the distance between the automatic face location and the ground truth.

The sequence *face*[1], shot with a Logitech webcam, shows a moving person under a fairly constant illumination. This sequence belongs to SPEVI dataset [1] [60], and it contains 447 frames where the person undergoes large scale movements, as well as abrupt motions and partial disappearance from the scene. The five algorithms were applied in order to track the face of the moving person based on shape, intensity histogram, and boundary gradients respectively. The performances are evaluated by measuring the distances between each estimated face location and the ground truth[1]. As shown in Fig. 3.7, all the four competing trackers using regional and boundary appearance models have significantly higher and fluctuating errors than the proposed tracker with shape based appearance model. This is mainly caused by the fact that, first, the intensity profile of the face is similar to the nearby background and, second, the face boundary present many weak intensity gradients spots. As shown in Fig. 3.8, these ambiguities are behind the fact that the yellow, green, blue, and magenta ellipses corresponding to the competing trackers are at-

[1]The sequence *face* and the ground truth are available at `http://www.eecs.qmul.ac.uk/~andrea/spevi.html`

tracted by the neighboring regions having similar intensity and gradient profiles to the person's face. However, shape information is shown to be very useful to keep track of the person's face with high accuracy except for few moments about the frame number 350 because the face has disappeared from the scene.



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

Figure 3.8: A sample of the results by the competing trackers with *face* sequence. The red ellipse corresponds to our method results, and the others ellipses correspond to the competing trackers. (a) Frame 42. (b) Frame 92. (c) Frame 200. (d) Frame 350. Our tracker outperforms, in terms of tracking accuracy, the competing methods which are attracted by the nearby confusing areas.

The sequence *cup*[2] contains 260 frames where a desktop is monitored by a moving camera [85]. The purpose here is to track the cup on the desktop. This sequence presents two main difficulties. First, the cup has no distinct textural information and its intensity profile is similar to the background. Additionally, intensity gradients along the cup's boundary are lower than those in many areas in the background. Similarly to the previous experiment, we illustrate by Fig. 3.9 the tracking errors corresponding to all the five tested algorithms. The tracker using intensity histogram fails to track accurately the cup and presents high tracking errors because the kernel is attracted by the wall which has an intensity profile similar to that of

[2]The sequence is from `http://wwwcremers.in.tum.de/data/bottledata.zip`

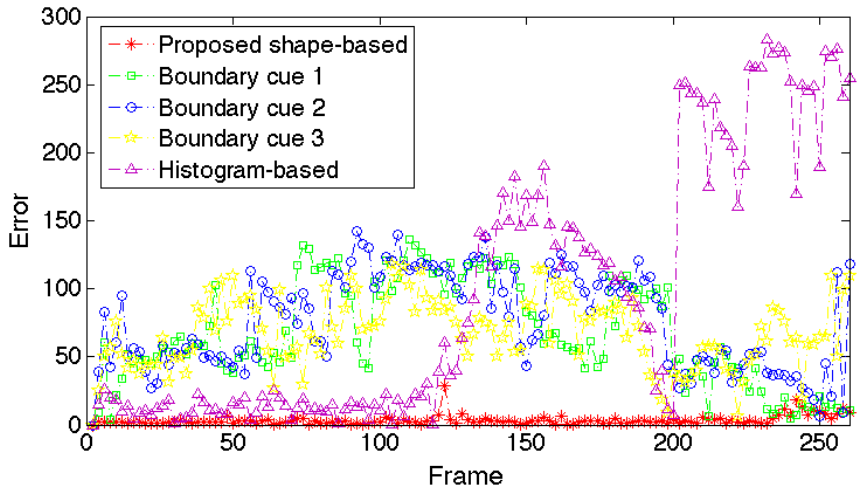


Figure 3.9: Comparison of the tracking errors recorded by the five competing trackers with *cup* sequence. The red curve corresponds to our shape based appearance model. The magenta curve corresponds to the intensity histogram based appearance model, and the green, blue and yellow curves correspond to the appearance models based on the three boundary cues respectively.

the cup. This is shown in Fig. 3.10 using the magenta ellipse in four frames from the entire sequence. Similarly, the trackers based on boundary gradients present big errors (refer to Fig. 3.9) because they get attracted by areas with higher gradients such as the keyboard. These trackers' kernels are represented in Fig. 3.10 using green, blue and yellow ellipses. In contrast, both Figs 3.9 and 3.10 demonstrate that our method is able to track the cup throughout the whole sequence except for a few moments when the cup disappears from the camera view. Again, the shape information we embedded in our appearance model is shown to be sufficient to distinguish between the target and the surrounding objects, and is necessary for having accurate results in such scenarios.

The sequence *pedestrian* belongs to the BEHAVE dataset[3] showing moving people in an outdoor scene. This sequence comprises 240 frames where the targeted pedestrian moves in front of the camera. Although non rigid, the shape of pedestrian body remains relatively consistent throughout the entire sequence, while its intensity profile is very similar to the nearby car. We run all the five competing

[3]The sequence *pedestrian* and the ground truth are available at `http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/INTERACTIONS/`

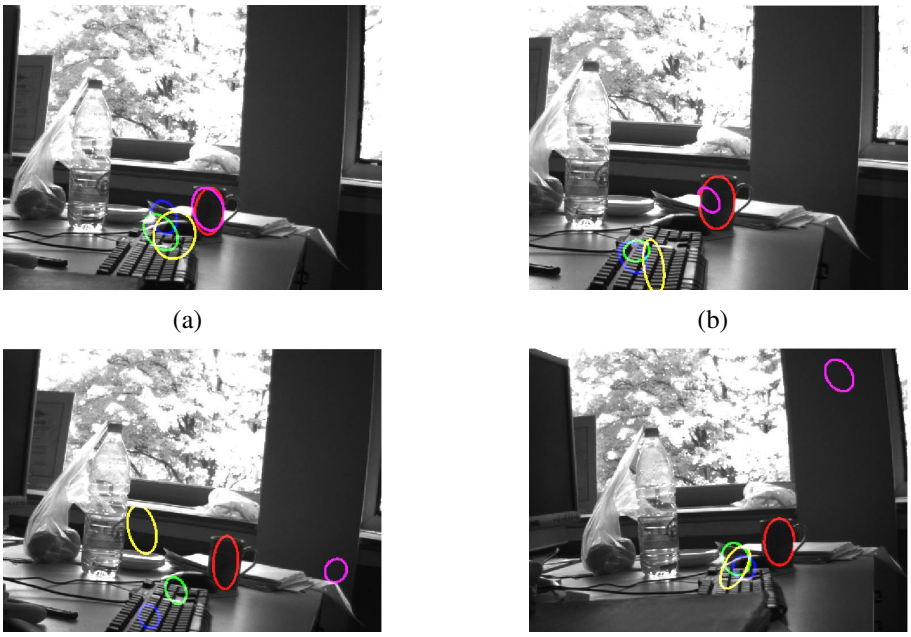


Figure 3.10: A sample of the results by the competing trackers with *cup* sequence. (a) Frame 28. (b) Frame 91. (c) Frame 158. (d) Frame 213. The red ellipse corresponding to our method is keeping track of the cup over the whole sequence. The intensity histogram based tracker (magenta ellipse) loses the track and gets attracted by the wall. The boundary gradients based trackers (green, blue and yellow ellipses) are attracted by the keyboard area.

algorithms on this sequence and we show a sample of the obtained visual results in Fig. 3.12 with the tracking errors in Fig. 3.11. We obtained similar results to the previous experiments with the *cup* and *face* sequences. In this case, the competing trackers get attracted by other misleading regions (car windows, car hood, and the street curb) which have similar intensity and gradient characteristics but different shapes from the pedestrian silhouette. The proposed method succeeds to track the body of the pedestrian accurately until the last frame as it is shown in Fig. 3.12 using the red ellipse.

We have demonstrated, using three challenging sequences, that our shape based tracker is able to track accurately the moving targets while the competing methods fail due to many difficult issues which are very likely to be encountered in real applications. We have showed also that the proposed tracker, even by embedding such shape description, remains computationally efficient compared with silhouette

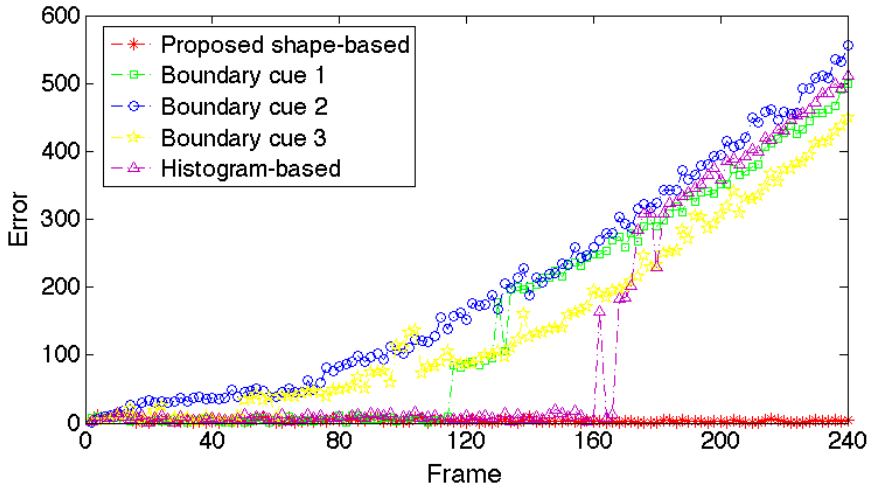


Figure 3.11: Comparison of the tracking errors recorded by the five competing trackers with *pedestrian* sequence. The red curve corresponds to our shape based appearance model. The magenta curve corresponds to the intensity histogram based appearance model, and the green, blue and yellow curves correspond to the appearance models based on the three boundary cues respectively.

trackers.

3.4.3 Quantitative Study: Large Lump Detection

In this subsection, we tackle the real Large Lump Detection application in oil sands mining as shown in Fig. 3.13. In this study, we run two kernel based trackers over a dataset which contains more than $64,000$ test images. The first tracker employs two regional cues combined with the proposed shape cue, and the second tracker utilizes the two regional cues only. Hence, such a comparative appraisal over the LLD dataset can help to investigate the contribution of the shape term. Furthermore, we also include the results obtained by applying a classification based method conceived specifically for the LLD problem [71].

As previously outlined, we tackle this problem using a kernel tracker utilized for joint detection and tracking (JDT) of lumps [97] from which arises two variants. These two variants differ from each other in terms of the employed appearance models. In the first variant, two regional cues are embedded in its appearance model and we refer to it by *region based JDT*. Similar to the shape based appearance model

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

Figure 3.12: A sample of the results by the competing trackers with *pedestrian* sequence. (a) Frame 62. (b) Frame 113. (c) Frame 123. (d) Frame 195. The red ellipse corresponding to our method is keeping track of the pedestrian over the whole sequence. The intensity histogram based tracker (magenta ellipse) loses the track and is attracted by the car area. The boundary gradients based trackers (green, blue and yellow ellipses) are attracted by the car windows and the street curb.

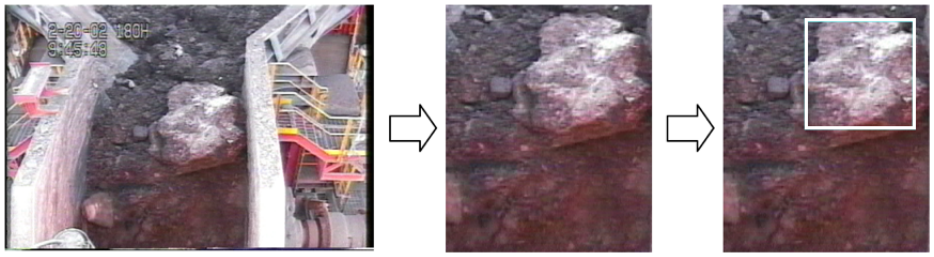
defined in Eq. (3.14), the region based appearance model is defined as

$$p(Z|X) \propto exp(-\lambda \cdot f_{reg}(F_{blob}(X), F_{sm}(X))). \tag{3.17}$$

$F_{blob}(X)$ and $F_{sm}(X)$ are the two regional cues that will be detailed in the following. $f_{reg}(\cdot)$ is a function learned by a support vector machine (SVM) from a training set, and it computes the signed distance between a feature point and the maximum-margin hyper-plane [26]. $f_{reg}(\cdot)$ indicates how likely the two input regional cues correspond to the foreground object or the background. The second variant embeds, in addition to the regional cues, our proposed shape cue and we refer to it by *shape aided JDT*. Specifically, the shape aided appearance model extends the above region based appearance model into

$$p(Z|X) \propto exp(-\lambda \cdot f_{sh-aided}(F_{blob}(X), F_{sm}(X), F_{sh}(X))). \tag{3.18}$$

$f_{sh-aided}(\cdot)$ is learned similarly to $f_{reg}(\cdot)$ based on the three regional and shape



(a) (b) (c)

Figure 3.13: Detecting large lumps in an oil sands video stream. (a) A view of the scene where a large lump is present in the feed just before falling in the crusher. (b) Region of interest which mainly comprises the dirt that is to be processed. (c) Detection and localization of the large lump.

cues. The shape cue $F_{sh}(X)$ that we adopt in this set of experiments is based on Eq. (3.7) employing the energy vector in Eq. (3.10) which captures geometric shape constraints. The reason is that lumps are more likely to have convex shapes with smooth boundaries, different from the arbitrary boundaries of the nearby fine material. The first regional cue $F_{blob}(X)$ employed in the above two appearance models uses a *blob* feature computed from the scale normalized Laplacian of Gaussian (LoG) operator [57]. The reason is that lumps appear more often as blobs which are brighter than the surrounding fine material. Given an input image $I(x, y)$, its scale-space representation L is computed by convolving it with a Gaussian kernel $g(x, y, \sigma^2)$ at the scale σ^2

$$L(x, y; I; \sigma^2) = g(x, y, \sigma^2) * I(x, y). \tag{3.19}$$

Then, the *Scale Normalized Laplacian* operator is defined as

$$\nabla^2_{norm} L(x, y; I; \sigma^2) = \sigma^2 (L_{xx} + L_{yy}), \tag{3.20}$$

and it measures how likely a specific region appears as a blob. L_{xx} and L_{yy} denote the second order derivatives of L along the x axis and y axis. Specifically for an elliptical object state X (as shown in Fig. 3.14(a)), we compute its blob response $F_{blob}(X)$ as follows,

$$F_{blob}(X) = \nabla^2_{norm} L(x, y; I_A; \sigma^2) \tag{3.21}$$

where I_A is an image affine-transformed from the original image I without considering translation, i.e., $I_A(x', y') = I(x, y)$ for all $[x'\ y']^T = H_A \cdot [x\ y]^T$. With this

transformation, the elliptical object is transformed to a circular object with the same area as shown in Fig.3.14(b). The transformation matrix

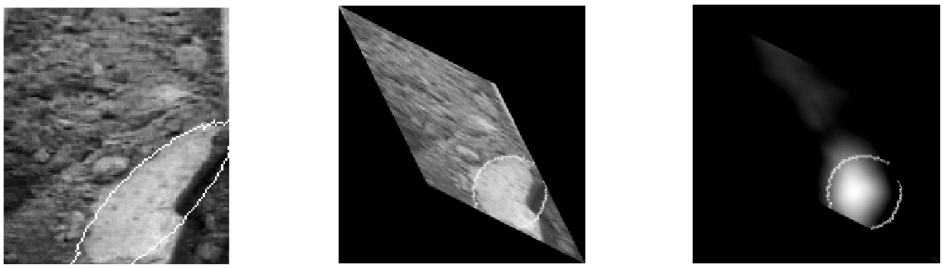
$$H_A = \begin{bmatrix} \sqrt{b/a} & 0 \\ 0 & \sqrt{a/b} \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & \sin\theta \\ -sin\theta & \cos\theta \end{bmatrix} \tag{3.22}$$

is determined by the elliptical object's shape parameters specified in Eq. (3.6). Then we convolve the transformed image I_A by the scale normalized LoG with the same scale σ^2 to the transformed circular object. It computes the blob response image F_{blob} shown in Fig. 3.14(c). The response at the object location, $F_{blob}(X)$, is finally extracted as the blob feature used in the appearance model in Eqs (3.17) and (3.18). The second regional cue $F_{sm}(X)$ embeds a *smoothness* feature computed from the coefficients of the discrete wavelet transform (DWT). The reason is that lumps have surfaces with a different smoothness from the neighboring fine material. For a specific object state X, its smoothness feature is defined as

$$F_{sm}(X) = \frac{1}{3|R_X|} \sum_{(x,y)\in R_X} (LH^2(x,y) + HL^2(x,y) + HH^2(x,y)). \tag{3.23}$$

LH, HL, and HH are horizontal, vertical and diagonal subimages of the original image's first level wavelet decomposition. $(x, y) \in R_X$ represents all the pixels within the object region R_X. For example, Fig. 3.15(b) shows the average of the three square wavelet subimages, $(LH^2 + HL^2 + HH^2)/3$, computed from the original image Fig. 3.15(a). The smoothness feature of the object state X indicated by the white ellipse in Fig. 3.15(b) is then the mean value within the elliptical object region.

Our dataset is an oil sands video sequence recorded during daytime (from 9:30am to 3:30pm) of three days in January 2010 (from 23rd to 25th). For a sampling rate of one image per second, we collected during the 18 hours, with a total of approximately 18 x 3600 = 64,800 images. The whole sequence contains a total of 29 different large lumps. The first 10 lumps, in addition to some images without lumps between January 23rd and January 24th are used for training. The rest of the images during January 25th and containing 19 lumps are used for testing. For detection performance evaluations, we use precision, recall, and F-score [94] measures defined as follows:



(a) (b) (c)

Figure 3.14: (a) A lump is described by an elliptical object state indicated by the white ellipse. (b) An affine transformation is conducted to the image in (a) so that the elliptical object becomes a circular object with the same area represented by the white circle. (c) The scale normalized LoG response of the transformed image in (b). The response at the object location indicated by the center of the white circle is extracted as the blob feature in the appearance model in Eqs (3.17) and (3.18).

(a) (b)

Figure 3.15: (a) A lump is described by an elliptical object state indicated by the white ellipse. (b) The average of the three square wavelet subimages computed from the original image in (a). The mean value within the object region inside the white ellipse is extracted as the smoothness feature of the corresponding object state.

$$
\begin{aligned}
Precision &= \frac{N_{TP}}{N_{TP} + N_{FP}} \\
Recall &= \frac{N_{TP}}{N_{TP} + N_{FN}} \\
F\text{-}score &= 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}
\end{aligned}
\tag{3.24}
$$

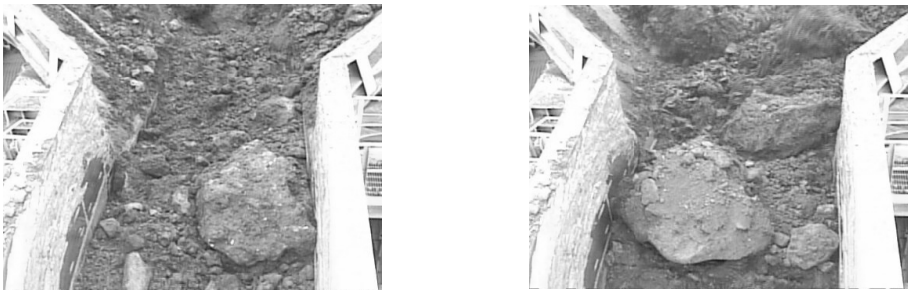
N_{TP}, N_{FP} and N_{FN} are respectively the numbers of true positives (TP), false positives (FP) and false negatives (FN). In this specific LLD application, these numbers are defined as event based. Indeed, true positive event implies that in the period during which a large lump is present on the scene, it is detected at least once. False positive implies that at least one small lump or fine material region (not considered as a large lump) is detected mistakenly during its period of presence on the scene.

Table 3.3: Comparison of the large lump detection performance among the shape aided JDT, the region based JDT and the classification based method in [71]

Methods	FP	TP	Precision	Recall	F-score
Shape aided JDT	0	10	100%	52.6%	0.69
Region based JDT	0	5	100%	26.3%	0.42
Classification based	0	1	100%	5.3%	0.1

Finally, false negative implies that during the period of presence of a large lump on the scene, it was never detected. Besides, the ground truth is provided by visually inspecting the sequence manually.

Table 3.3 shows the results of the JDT methods, with and without shape cue in the appearance models, and the classification based method in [71]. It illustrates the recorded quantitative measures in terms of precision rate, recall, and their combined score i.e. the F-score [94]. All the competing methods are tuned to have high precision rate which is preferred in this specific application. The embedded shape information enabled the JDT to detect 5 more lumps than the JDT without any shape information. Two such detection events are depicted in Fig. 3.16 showing that shape information is useful especially when the regional information is not strong enough to distinguish the object from the nearby background. The classification based method in [71] totally fails in this real situation although it recorded, as reported in the paper, a good performance on a selected small data set. Hence, the proposed model yielded the best results and provided a relatively acceptable F-score for this kind of problems. We also provide the Precision-Recall (PR) curves for the three competing methods recorded on this LLD dataset in Fig. 3.17. These curves allow comparing the different methods at specific precision rates and are used mainly because when the dataset is highly skewed, they provide a more informative picture of an algorithm's performance than others such as the Receiver Operator Characteristic (ROC) curve [30]. It can be concluded from the curves in Fig. 3.17 that the shape aided JDT always outperforms the other two methods.



(a) (b)

Figure 3.16: Two examples of lumps which are detected by the shape aided JDT method but not by the region based JDT method.

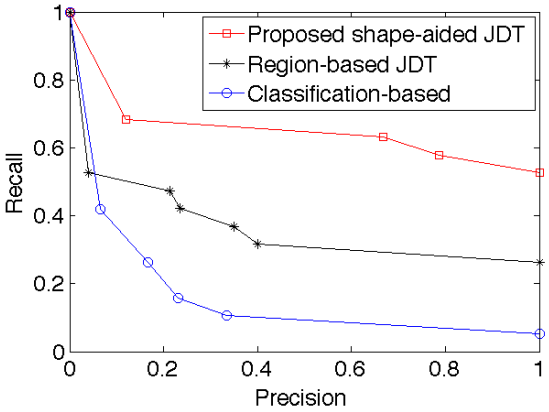


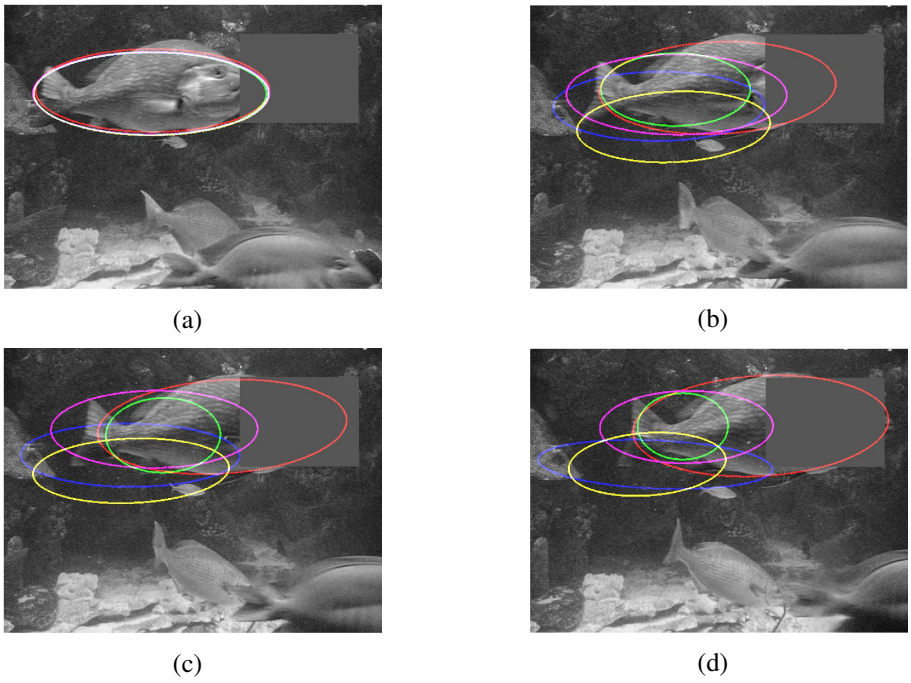
Figure 3.17: PR-curves corresponding to the shape aided JDT method, the region based JDT method, and the classification method in [71] applied to the LLD dataset.

3.4.4 Sensitivity

This subsection contains further experiments aiming to test the behaviour of the proposed tracker vis-a-vis certain design choices which may influence the quality of the tracking. For instance, we investigate its ability to handle partial occlusion, sensitivity to the number of sampled points along the kernel boundary, and the size of the initial kernel ellipse. For these explorations, the proposed algorithm is run on the sequence of a moving fish.

To show that the proposed method handles relatively well partial occlusion, we run the shape based kernel tracker on the artificially occluded fish sequence. Fig. 3.18 shows the tracking results of five competing trackers: our shape based kernel tracker and other four trackers based on intensity histogram and boundary

gradients. The red ellipse indicates our result that follows the fish correctly. The magenta ellipse indicates the result of the tracker based on intensity histogram, and the green, blue and yellow ellipses indicate the results of the three trackers based on boundary gradients. Due to the ability of handling partial occlusion brought by the weighing vector in the proposed shape cue, our shape based kernel tracker can locate the fish roughly at its centroid even though it is partially occluded. However, the other four trackers with the existing cues currently used in kernel based trackers fail to track the fish correctly because of the occlusion disturbance.



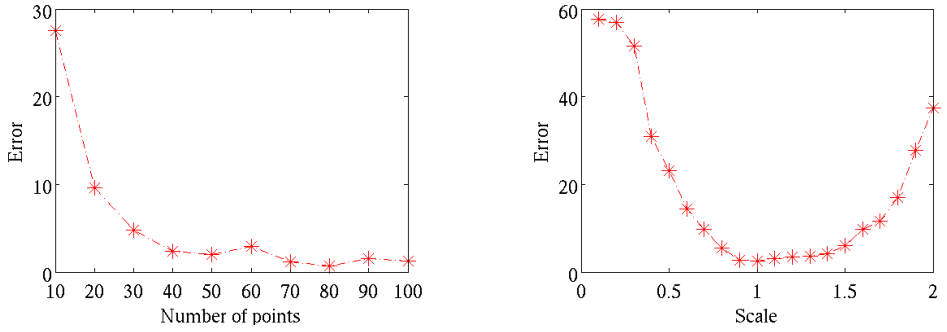
(a) (b)

(c) (d)

Figure 3.18: A sample of the results by the competing trackers with *fish* sequence in presence of occlusion. From (a) to (d), the fish becomes more and more occluded. Our tracker (refer to the red ellipse) locates relatively well the fish. The competing trackers are biased by the occluding strip.(a) Initial frame. (b) Frame 10. (c) Frame 14. (d) Frame 19.

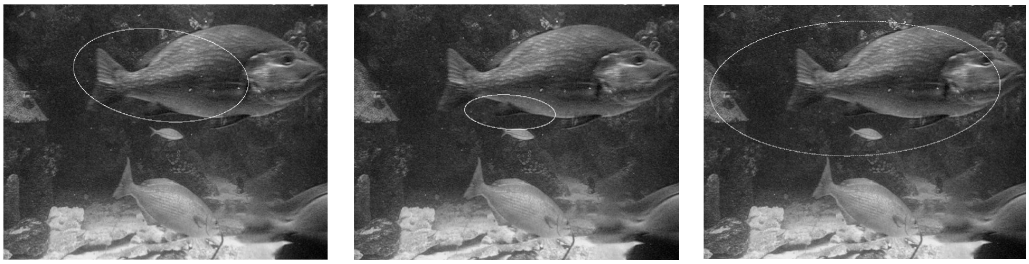
To explore the effect of the number of points sampled along the kernel boundary, we vary this number from 10 to 100 and learn the respective shape based appearance model. Fig. 3.19(a) shows the tracking results when the different numbers of points are used. The Y-axis reports the average per frame tracking error between the estimated object centroid and the ground truth. According to Fig. 3.19(a), it is shown that at a certain value of the number of boundary points, the correspond-

ing tracking error reaches its steady state. When the number of points is beyond that threshold, less than 20 points in this case, the tracking error becomes relatively high. This is not surprising since the shape information is captured via the contour points. Fig. 3.20(a) shows how the tracker fails when only 10 contour points are used.



(a) (b)

Figure 3.19: (a)The average tracking error on each frame of the fish sequence when different numbers of sampled boundary points are used in the shape based appearance model.(b) The average tracking error on each frame of the fish sequence when different sizes of the initialized ellipses are used.



(a) (b) (c)

Figure 3.20: (a) The failed track when only 10 sampled boundary points are used in the shape based appearance model. (b) The failed track when the initialized ellipse is only one tenth of the size of a properly initialized ellipse. (c) The failed track when the initialized ellipse is twice of the size of a properly initialized ellipse.

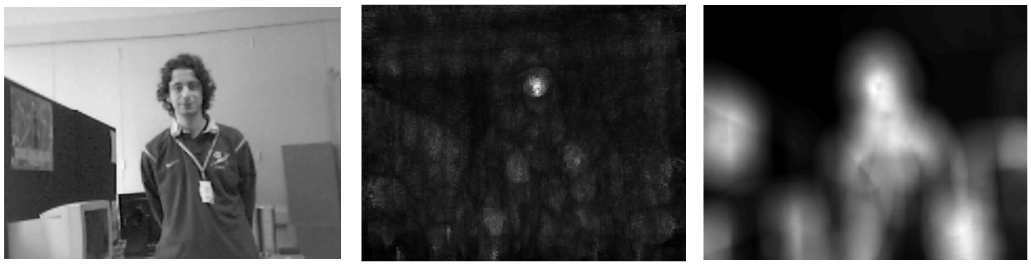
To explore the effect of initialization, we also run our shape based tracker using initial ellipses with different sizes. Fig. 3.19(b) reports the tracking errors corresponding to the different initializations. The X-axis indicates the relative size of the initial ellipse where the value of 1 corresponds to the size of the kernel that properly covers the targeted object. For instance, a relative size value of 0.5 means

an initial ellipse of half the size of the aforementioned ellipse (the one with relative size 1). Similarly, the Y-axis reports the average per frame tracking error on the fish sequence. It can be concluded from Fig. 3.19(b) that, away from the relative size of 1, the corresponding tracking errors get higher. For instance, Figs 3.20(b) and 3.20(c) show the poor tracking quality corresponding to relative sizes of 0.1 and 2 respectively. However, it is worth mentioning that acceptable tracking errors can be obtained for a large range of relative size values. Again, this behaviour is predictable because the object boundary cannot be captured if the kernel is too small/big compared to the size of the target. Recall that the object boundary is detected by looking for the highest image gradient along the normal lines going through the kernel boundary points.

3.4.5 Discussion

All the above experimental results mainly demonstrate the previously stated advantages of our shape based kernel tracker. It is shown to be more efficient than silhouette tracking methods in terms of computational load, and it is more accurate than the existing kernel based trackers by exploiting the proposed prior shape information. Furthermore, an additional ability of handling partial occlusion is also demonstrated in the last experiment. Since the purpose of our work in this Chapter was to demonstrate the benefits of using shape prior to achieve a robust kernel tracking, other parts of the tracking process were not within the scope of this paper. For instance, we initialized all the above experiments manually by positioning a kernel roughly on the object. We noticed that the performance of the proposed method is quite sensitive to random initializations. This is mainly due to the fact that the computed observation likelihood distribution is not smooth enough. Indeed, the shape based likelihood usually peaks around a meaningful fit, and if candidate object contours are not extremely close by the true fit, the likelihood assigned to the contours is often meaningless [10]. Fig. 3.21(b) shows the likelihood map of the proposed shape based appearance model. The greyscale value at a certain pixel location is proportional to the likelihood of being the face centroid. The likelihood distribution peaks only within a small vicinity of the face centroid and vanishes abruptly.

Although this is beneficial to distinguish the object from the background, it is inefficient when searching for the optimum from an arbitrary initialization. This problem can be alleviated when the shape based appearance model is combined with other existing models in kernel tracking. For example, Fig. 3.21(c) shows the likelihood distribution of the appearance model based on intensity histogram. Although it is not distinctive enough to separate the face from the background, it is smooth in large scale so that it can efficiently guide the tracker towards promising areas that may contain the object of interest. Therefore, the combination of our proposed shape based appearance model with the existing models might accentuate the advantages of each.



(a) (b) (c)

Figure 3.21: (a) Image of interest showing the target which is the person's face. (b) The likelihood distribution of the proposed shape based appearance model. It peaks at the close vicinity of the face centroid, with an advantage of being distinctive but a disadvantage of being inefficient in guiding the tracker towards the optimum when the kernel is initialized far away. (c) The likelihood distribution of the intensity histogram based appearance model. The likelihood distribution is obviously not distinctive, but it is smooth in a large scale so it can guide the tracker towards the promising solutions.

3.5 Summary

In this chapter, we proposed an original method to integrate shape into the appearance model of kernel based trackers. These shape based kernel trackers have two advantages. First, the exploited shape prior information is complementary to the existing regional textural or boundary gradient information that are currently employed in kernel based trackers. Second, the proposed method is more efficient than those used in silhouette based trackers. The above two advantages are demon-

strated by the experiments on various sequences. The results show that the shape based kernel tracker localizes the targets more accurately than the competing kernel trackers, and is faster than the competing silhouette based trackers in terms of the computational load.

Chapter 4

Steam Detection

Occlusion is a common problem in object detection, and it can be caused by the objects of interest themselves, self occlusion, or by other objects present on the scene. When occlusion is serious, i.e., the object of interest is totally occluded, there is no information available to perform detection. Hence, the system input images with such occlusion need to be labeled in order to stop the detection process. Because of the application motivating this work, the occlusion tackled in this chapter is mainly related to steam in the large lump detection problem. We propose a general steam detection method to overcome this problem. The existing steam detection methods feasible for the LLD application generally extract features from the transformed input image first and then feed them to a classifier in a completely independent step. In these methods, the step of feature extraction is usually cumbersome and application-dependent. Therefore, we propose a new steam detection method in this chapter by feeding directly the transformed image to an Adaboost classifier. By doing so, we discard the considerable computational load normally dedicated to feature extraction and benefit from the accuracy of the propoer classifier built by Adaboost.

4.1 Introduction

Steam is present in many image processing applications where it may occlude the objects of interest. For example, steam may occlude the input oil sand images where large lumps need to be detected [106]. This problem motivated our research

on steam detection, to ensure that the input images to the large lump detection algorithm are steam free. Apart from our specific application, steam detection is also important for many environment surveillance applications, e.g., steam leaking detection in industry environments. Furthermore, the similarity in appearance between steam and smoke makes steam detection techniques interchangeably used for the purpose of smoke detection and early fire warning. This fact further enhances the usefulness of steam detection techniques. In this chapter we investigate all this class of problems although we specifically consider the steam detection application.

Various features have been employed for steam/smoke detection, and they are mainly related to color, motion, area properties, texture, and energy. Color might be the most intuitive feature. Steam/smoke appearance spectrum is commonly assumed from white to gray. Many proposed methods distinguish steam/smoke images from steam/smoke-free images by analyzing the information in color spaces such as RGB, HSI, YCrCb,CIELab [40, 50, 98, 58, 15, 21, 13, 51, 104]. For instance, Chen $et\ al.$ proposed a smoke detector in [19] which assumes that smoke has a grayish color with certain chromatic conditions. This is not generally sufficient as steam/smoke changes color with noise and the scene lighting conditions [39]. Other approaches [98, 22, 93, 15, 21, 104] use motion information as a feature for steam/smoke classification. They implicitly assume that steam/smoke moves consistently in a certain direction for a certain period of time. However, any motion pattern of steam/smoke can be easily violated in presence of external effects such as wind. In addition to motion, other properties within the steam/smoke area are also exploited for classification in [40, 100, 101, 50, 98, 37, 93, 105]. Among the assumptions related to steam/smoke area properties are roundness, contour fluctuation, growth rate, etc. For example, in [19], smoke area is first segmented accurately from the background and then the method checks whether the smoke area's disorder measurement and growth rate follow certain characteristics. In these methods, a static background is required and the area properties are not clearly stated. For example, the smoke area is assumed to grow steadily in [98, 37], while it is assumed to grow irregularly in [93]. Texture features have also been used for smoke classification in [62, 93], such as surface roughness, contrast, inverse difference moment

and difference entropy. However, they are not popularly employed in this context possibly due to the difficulty in modeling the visual pattern of smoke [100].

A specific class of methods detect steam/smoke from an energy point of view in the wavelet domain. Their basic assumption is that steam/smoke introduces a smoothing effect to the image, i.e., the steam/smoke region should have different energy spectrum from the steam/smoke-free region. We are interested in this group of methods because it is suitable for our LLD application. The methods discussed in the previous paragraph are not preferred due to some practical constraints in the LLD application. For example, color based features might be insufficient when steam density and illumination change. Other features based on motion and area properties assume for a static background which is not the case in our application. However, in general, it is not necessary that energy based features are more general or robust than the others. Various methods have been proposed within this group of steam detection methods based on energy features. For instance, in [58, 13, 91, 75, 92] a composite image is computed to capture the image's energy variation in the wavelet domain by summing up the squared coefficients of each discrete wavelet transform (DWT) subband. A decrease in the image's energy is an indication of the appearance of smoke. However, a static background is always required in all these five methods. The closest works to our method might be [33] and [38]. In [33], a statistical hidden Markov tree (HMT) model is derived to characterize the image's steam texture from the coefficients of the dual-tree complex wavelet transform (DTCWT), and then a support vector machine (SVM) is used for classification. Two slightly different methods are proposed in [38]. The first method decomposes the input image by applying a discrete cosine transform (DCT), and then the 100 low frequency coefficients are selected as features to be fed into SVM for steam classification. The second method decomposes the input image by applying DWT, and then six statistical features are extracted from each sub-band and fed into SVM. The above three methods are fundamentally similar and all of them are based on three main steps. The first step, *image transformation*, employs a transformation method to obtain frequency and/or spatial domain information from the input image. The second step, *feature extraction*, employs a feature extraction

method to generate the features to be fed to a classifier. These features are then used to decide whether the input image is labeled as steam or steam-free image via a classifier in the third step, *image classification*. Among the weaknesses of the steam detection methods discussed above is the selection of the feature extraction method which is generally arbitrary so that a compromise between accuracy and efficiency becomes difficult. Indeed, in some cases the extraction of features is efficient in terms of execution time but leads to unsatisfactory results in terms of accuracy. Others focus more on the accuracy in the choice of the features and their extraction methods, making the technique time consuming. For this reason, we propose a new steam detection method to discard the *feature extraction* step by directly feeding the DWT image to an Adaboost [35] classifier, whose strength is the selection of a proper classifier. In this case, Adaboost selects a subset of the coefficients of the DWT applied to the image in a way which optimizes both accuracy and execution efficiency.

The remainder of this chapter is organized as follows. In Section 4.2, we first present how DWT characterizes both steam and steam-free images in order to enable their classification, and then how this classification is performed via Adaboost using the properly captured information. Section 4.3 includes the experimental results that illustrate the performance of this method. Finally, the work is summarized in Section 4.4.

4.2 Proposed Method

Given an input image, steam detection can be defined as the process of determining whether steam is present in the scene or not, and if so, localizing it. We solve this problem by partitioning the input image into a grid of small patches (as in Fig. 4.1), and then classifying each patch as steam or steam-free. In this way, the detected steam area may be blocky, but there is no assumption of a static background. Therefore, we focus in this chapter on how to classify an image patch (rather than a whole input image) as steam or steam-free. In short, the proposed method first transforms an image patch with DWT, and then classifies the image patch based on

the transformed wavelet coefficients using Adaboost.

Figure 4.1: An example of dividing an input image into patches for classification.

4.2.1 Steam Characterization via DWT

Image regions of steam are generally characterized by a smooth appearance while steam-free image regions rather have a coarse appearance with various edges and textures. Fig. 4.2 shows examples of steam image patches in the upper row and steam-free image patches in the bottom row. Considering the properties of smoothness and coarseness in frequency domain, coarseness from edges and texture in the image is mainly represented by high frequency information, i.e., its energy resides in high frequency bands. Steam generally has an effect of obstructing these edges and textures in the images. Therefore, when steam is present, the image becomes a low frequency signal and its energy is mainly concentrated in low frequency bands. This difference in frequency domain is the core idea which has been used by wavelet based steam detection methods to differentiate steam images from steam-free images. Specifically, the wavelet transform decomposes an image into different frequency components and based on that, a classification decision is made. Due to the specific advantages of wavelet transform such as being fast, linear and relatively less sensitive as other existing features for steam detection, wavelet based steam detection methods have become more and more popular.

A wavelet transform can be represented by a linear combination of a group of

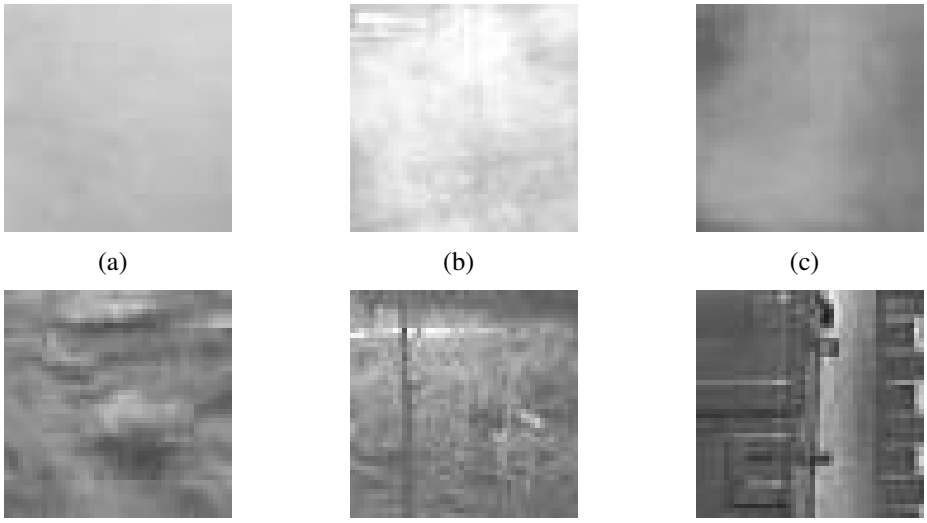


Figure 4.2: Examples of steam/smoke images in the first row and steam-free (smoke-free) images in the second row.

basis functions. Continuous wavelet transform (CWT) can be computed as

$$f_\psi(a,b) = |a|^{-1/2} \int f(t)\psi((t-b)/a)dt, \tag{4.1}$$

where $f(t)$ is the signal for processing, a is the scaling coefficient, and b is the translating coefficient. $\psi(t)$ is called the mother wavelet function. $|a|^{-1/2}$ is a normalization factor that ensures the transformed signal to have the same energy level in each scale [5]. In image processing, DWT is used as a compact representation of CWT. DWT is obtained by dyadic sampling of CWT in time and frequency domains, and can be computed efficiently by convolution of the original signal with a series of wavelets. In practice, the DWT decomposes an image with successive low-pass and high-pass filtering. For example, a three level decomposition of an image is demonstrated in Fig. 4.3. Similarly, k level decomposition of an image can be represented by $3k+1$ sub-bands:

$$[A_k, (H_1, V_1, D_1), ..., (H_k, V_k, D_k)], \tag{4.2}$$

where A_k is the approximation coefficients at the k^{th} frequency resolution, and (H_i, V_i, D_i) is the detailed coefficients at the i^{th} frequency resolution representing the detail information obtained by high-pass filtering in three directions: horizontal, vertical and diagonal.

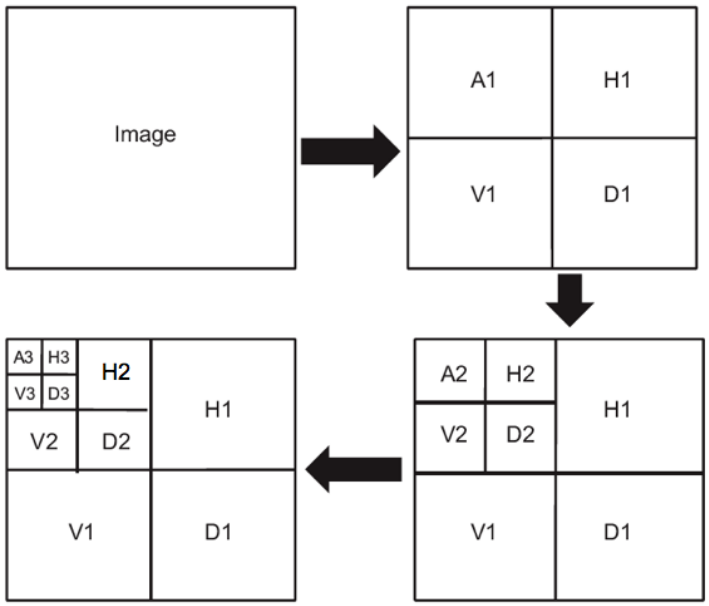


Figure 4.3: Multilevel (3 level) DWT decomposition of an image. The figure is from [38].

The DWT decomposition on one level provides the information corresponding to a certain frequency band. Therefore, multi-resolution analysis, as shown in Fig. 4.3, enables us to analyze the image in different frequency bands. In steam detection application, various situations can happen. Steam may vary from being dense to sparse, and background can have different textures. Therefore, exploiting information from limited scale analysis as in [33] is obviously not sufficient to deal with complex inputs. In our method, full-scale DWT decomposition is employed for the steam classification as preliminary results confirmed that employing information from more scales enhances the performance.

4.2.2 Discrete Wavelet Transform Utilization via Adaboost

In the previous subsection, information has been obtained in the full scale DWT decomposition of the input image, and the next step is to make a classification decision based on this information. Directly using the decomposed image as the input feature to a classifier is definitely inefficient and ineffective. Therefore, existing methods tried to generate some synthetic features from the decomposition image for two purposes. First, reducing the dimensionality of the input feature to

a classifier. Second, obtaining more compact and meaningful information from the raw DWT decomposition image whose information is diluted in high dimensional space. In this chapter, we demonstrate that Adaboost can fulfill these two purposes better than arbitrarily deriving some features as the existing methods do.

Adaboost is a meta-algorithm which can be used in conjunction with many other machine learning algorithms to improve their performances. It calls a weak classifier repeatedly in a series of rounds $t = 1, \ldots, T$. In each round the weights of incorrectly classified examples are increased, so that the new classifier focuses more on those examples. That way, a final strong classifier can be learned by Adaboost.

Utilizing the Adaboost's characteristics described above, the DWT decomposition image in the previous subsection can be naturally considered as a large set of features. These features can be used to build weak classifiers from which a strong classifier can be learned. Assuming we have an input image, we can calculate its DWT decomposition image, denoted by x. We consider each coefficient in the decomposition image as a feature from which a weak classifier may be learned. An example weak classifier $h_t(x)$ can be naive thresholding: $h_t(x) = sign(x_{id(t)} - th(t))$ where $id(t)$ specifies a certain coefficient entry and $th(t)$ is a threshold. Then in each round a specific weak classifier learned based on one coefficient is picked up and assigned a weight. After T rounds a strong classifier is learned by Adaboost according to the training set

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)). \tag{4.3}$$

Here, T is the number of weak classifiers used to build the final strong classifier $H(x)$. x is the input DWT decomposition image. $h_t(x) \in \{-1, +1\}$ is the weak classifier learned in round t and α_t is its weight. Notice that $h_t(x)$ is based only on $x_{id(t)}$, one dimension of x, although it appears taking the whole decomposed image x as input. The final classification decision can be made based on whether $H(x)$ is positive or negative.

By combining DWT with Adaboost, the proposed method can preserve both accuracy and efficiency for the following reasons. First, it selects wavelet coefficients from the decomposition image and combines them into a classifier in a proper way

learned from a training set. Second, the classification based on the selected coefficients is efficient and has a linear complexity.

4.3 Experimental Results

In this experimental section, we have conducted three types of tests. The first one is a comparative study against three competing steam classifiers in terms of both accuracy and time efficiency. The second test is an analysis of the specific contribution of each component of the proposed method. In the third test, we apply the proposed steam detector to the large lump detection problem.

4.3.1 Comparative Study

In the following comparative appraisal, we compare the proposed method with the three state-of-the-art steam/smoke classification methods in [33, 38] in terms of both accuracy and efficiency. We refer to the three competing methods in the following by DTCWT/HMT based method, DCT based method and DWT/statistics based method. The comparison is conducted on three data sets which are referred to as Steam, Smoky, and Wastebin. The first one is a steam data set with image patches cropped from oil sand mining images. This data set contains 200 steam-free patches and 150 steam patches, and all the patches are 48x48 pixels. Each patch is cropped from a distinct image so that the problem is not trivial. The second and third data sets are two smoke data sets collected from two smoke videos, each containing 900 frames. The two smoke videos are publicly available at `http://signal.ee.bilkent.edu.tr/VisiFire/Demo/SampleClips.html`. The second data set includes 150 smoke-free patches and 100 smoke patches, and the third data set includes 100 smoke-free patches and 100 smoke patches. By using three different data sets covering various steam, smoke and background cases, we aim to show the robustness and flexibility of the proposed method.

Table 4.1 shows the accuracy comparison of the four competing methods on the three data sets. The average classification accuracy values on the test set are collected using the bootstrap resampling method [32] 50 times. The bootstrap method

Table 4.1: Comparison of the three existing methods and our proposed method in terms of accuracy. Steam, Smoky, and Wastebin represent the three different data sets.

Method	Accuracy (%) ± std		
	Steam	Smoky	Wastebin
DTCWT/HMT	86.83± 2.4	84.15±2.74	78.18±6.15
DCT	91.18± 2.35	92.8±3.53	82.47±7.59
DWT/statistics	97.21± 1.54	97.87±1.34	95.1±3.96
Proposed Method	98.21±1.12	98.36±1.52	96.05±3.02

draws a set of training samples randomly with replacement from the whole data set each time. Consistent results are obtained on the three data sets. The DTCWT/HMT based method always records the worst accuracy, followed by DCT based method. The DWT/statistics based method has a comparative accuracy to our method, which provides the best mean accuracy among all the investigated methods.

Similarly, Table 4.2 shows the time efficiency comparison of all the four investigated methods. In this table, we list the corresponding theoretical complexities and computational times for all the three steps discussed in introduction, *image transformation*, *feature extraction*, and *image classification*. For the proposed method, the second and third steps are combined into one single step. The computational times are measured on a PC with 3 GHz Intel CPU, and they are collected based on an input image containing 9x9 small patches, each of which has a size of 48x48 pixels. The size of this image is just selected to follow the same size of the ROI processed in [33]. We can see that our method presents the lowest complexity and computational time in all the processing steps. Among all the three steps, notice that especially the combined second and third steps of our method have a much smaller complexity and computational time than the existing methods. This is due to the fact that Adaboost joints efficiently the feature selection and classification steps. The improvement in terms of time efficiency is particularly important for us because we use steam detection as a preprocessing step in our complete system for large lumps detection. As a result, the more this step is time efficient, the more time can be saved for the remaining important system components.

Table 4.2: Comparison of the three existing methods and our proposed method in terms of efficiency. n is the square image width. m is the iteration number for the expectation-maximization algorithm to estimate parameters. k is the number of support vectors. l is the feature dimensionality.

Method	Complexity			Computational Time (s)			
	Step1	Step2	Step3	Step1	Step2	Step3	Sum
DTCWT/HMT	$O(n^2)$	$O(n^2m)$	$O(kl)$	1.28	1.95	0.05	3.3
DCT	$O(n^2 \log n)$	$O(l)$	$O(kl)$	1.19	0.001	0.23	1.4
DWT/statistics	$O(n^2)$	$O(n^2)$	$O(kl)$	0.48	1.12	0.08	1.7
Proposed Method	$O(n^2)$	$O(l)$		0.48	0.007		0.5

4.3.2 Analysis Study

In the second experimental test, we investigate the effect of each component in our proposed steam classification method. First, we fix the image transformation method, DWT, and compare three classification methods, NN (nearest neighbor), SVM and Adaboost in terms of accuracy as shown in Table 4.3. The best performance of (obtained by) using Adaboost demonstrates that Adaboost actually does a good job by playing both the role of *feature extraction* and *image classification*. However without the *feature extraction* step, a high dimensional feature directly feeding a NN or SVM classifier does not work as well. Secondly, we fix the classifier, Adaboost, and compare the three different image transformation methods, DCT, DWT and DTCWT, which are used in the existing competing methods. Table 4.4 illustrates that, with different image transformation methods, the classification accuracy is more or less the same, i.e., these image transformation methods do no affect the final accuracy so much. Therefore, DWT is mainly preferred because it has the best computational efficiency as shown in Table 4.2.

4.3.3 Extended Study

In the following we run an experiment to test the efficiency of our complete steam detection method on a real sequence of large lump detection images. The sequence monitors the dry feed preparation stage which crushes ore material in oil sands mining industry. The sequence spans two hours and contains a total of 639 images

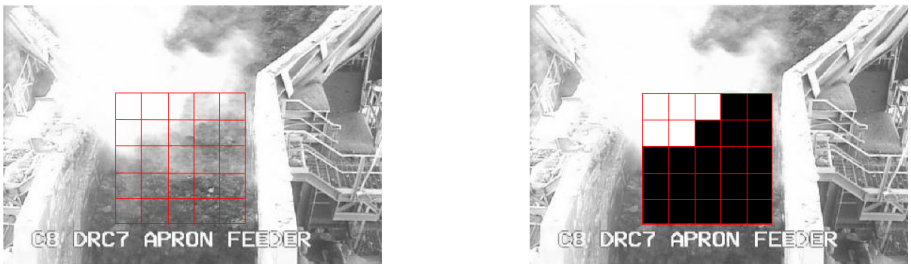
Table 4.3: Comparison of the classifiers when using the same image transformation method.

Method	Accuracy (%) ± std		
	Steam	Smoky	Wastebin
DWT+NN	66.81±4.01	63.6±4.48	74.41±5.64
DWT+SVM	90.07±1.86	91.99±3.19	89.17±5.17
DWT+Adaboost	98.21±1.12	98.36±1.52	96.05±2.35

Table 4.4: Comparison of the image transformation methods when using the same classification method.

Method	Accuracy (%) ± std		
	Steam	Smoky	Wastebin
DTCWT+Adaboost	97.95±1.43	98.58±1.24	95.9±2.13
DCT+Adaboost	98.17±1.37	98±1.62	97.73±2.35
DWT+Adaboost	98.21±1.12	98.36±1.52	96.05±2.35

obtained using a sampling speed of about one image every 10 seconds. A sample image of this sequence is depicted in Fig. 4.4(a). For each input image, a region of interest (ROI) is selected to be processed and it is partitioned into 25 patches. In this way, an input image can be labeled as steam or steam-free image depending on how many patches are classified as steam patch by the proposed classifier. In this set of experiments, we decide that an image is labeled as steam image when more than two patches are labeled so. For example, Fig. 4.4(b) shows the classification result of the input image in Fig. 4.4(a) where 5 patches are classified as steam patches and, as a result, the image is labeled as steam image. In the following, we evaluate the performance of our steam detection method on the whole sequence in terms of the accuracy rate. The ground truth is labeled by investigating the sequence of images manually. The results for the first and second hours of the sequence are reported in two different rows in Table 4.5. TP, TN, FP and FN are the numbers of true positives, true negatives, false positives and false negatives respectively. It is demonstrated that for each of the two hours, the proposed steam detector is able to obtain a high accuracy rate. Therefore, the proposed method is able to ensure that the subsequent processes in large lump detection will not be interfered by steam.



(a) (b)

Figure 4.4: (a) An example frame from a sequence of images monitoring the dry feed preparation stage which crushes ore material in oil sands mining industry. A region of interest is selected in the image and divided into patches for steam classification. (b) The steam classification result of the input image in Fig. 4.4(a).

Table 4.5: The performance of our proposed steam detection method on a sequence of images spanning for two hours. The results for the first and second hours are reported respectively in two rows.

Sequence segment	TP	TN	FP	FN	Accuracy
First hour	71	221	11	17	91.3%
Second hour	64	230	3	22	92.2%

4.4 Summary

In this chapter, a new steam detection method is proposed based on DWT and Adaboost. By feeding DWT coefficients directly to Adaboost, this general steam detection method has three advantages:

- It is computationally efficient, because both DWT decomposition and Adaboost classification are fast.
- It is accurate, because a proper classifier is learned by Adaboost based on the training set.
- It is automatic, because features are automatically selected by Adaboost rather than user dependent as in the existing methods.

The proposed method is also shown to be general enough to bear with other fundamentally similar problems. Indeed, some of the presented tests have shown very good results on two *smoke* data sets. We plan to apply this method for more smoke datasets and other similar problems.

Chapter 5

Large Lump Detection

In the previous chapters, we proposed the three components which make up our large lump detection system. The adaptive multi-motion JDT model introduced in Chapter 2 and shape based appearance model proposed in Chapter 3 provide an object detection method. The steam detection method presented in Chapter 4, a preprocessing component, is used to ensure that objects to be detected are not occluded by steam. In this chapter, all these components are integrated together in a complete system for large lump detection (LLD). This system will be first presented, and then performance evaluation study is performed.

5.1 Large Lump Detection System

The large lump detection problem can be formally described as follows. Given an input image, label it as one of these three states: steam, lump or lump-free. A system integrating all the proposed components (shown in Fig. 1.4) is designed to address the LLD problem as described in the Introduction section. (Note that Fig. 1.4 shows the flowchart of processing only one input image.) First, an input image similar to Fig. 5.1(a) is grabbed. The region of interest (ROI) is defined as a trapezoid region for this application. An example ROI with its set of coordinates is shown in Fig. 5.1(a) by black lines. The defined ROI is then cropped and stretched into a rectangle as in Fig. 5.1(b) to compensate for the distortion due to the camera angle. From now on, the processed ROI will be referred to as the input image. The input image is first fed into our proposed steam detector. It will be labeled as a steam

image or a steam-free image depending on how many steam patches are detected in it. This step is required because in the wintertime steam may occur frequently due to the water used in oil sands processing. This step ensures that lumps are visible and not occluded by steam for better detection accuracy. When an input image is labeled as steam image, its processing is interrupted at that level. Otherwise, it is fed into the following JDT component, to be labeled as lump image or lump-free image. In the following, the JDT step will be described in details.

(a) (b)

Figure 5.1: (a) An input LLD image with ROI specified by black lines. (b) ROI cut from (a) and stretched into a rectangle.

In the JDT step, three components need to be defined. The first one is the object state at time t, which can be defined as

$$X_t = [x_t \; y_t \; x'_t \; y'_t \; a_t \; b_t \; \theta_t \; \alpha_t \; E_t]^T. \tag{5.1}$$

(x_t, y_t) and (x'_t, y'_t) are the location and velocity of a lump. (a_t, b_t, θ_t) specifies the shape of a lump which is modeled as an elliptical patch. a_t and b_t are respectively the major and minor axis of the ellipse. θ_t is the orientation of the ellipse's major axis. α_t and E_t are respectively the motion model variable and existence variable. The second component which needs to be defined is the motion model set in our multiple motion model JDT. A simple inspection of the large lump video sequence shows that, generally, lumps move from top to bottom along a vertical line. They first accelerate slowly on a conveyor and then drop suddenly when they reach the end to fall into the crusher. Therefore, lumps may be considered to have two motion models: a small constant acceleration model on the conveyor and a large constant acceleration model off the conveyor. The last component which needs to be defined

in JDT is the appearance model. It employs the previously proposed shape aided appearance model (in Eq. (3.18)) discussed in Section 3.4.3. It combines three features together in the appearance model, blob feature, smoothness feature, and shape feature, which are fused by a machine learning method. Normally based on a training set, a machine learning method can learn a model that indicates how likely an input belongs to the foreground or the background. We used a support vector machine (SVM) in this problem to learn a function $f_{sh-aided}$ that computes the signed distance between the combined three features and the maximum-margin hyperplane. The final appearance model is built on this learned function $f_{sh-aided}$ using a simple exponential formulation.

$$p(Z|X) \propto exp(-\lambda \cdot f_{sh-aided}(F_{blob}(X), F_{sm}(X), F_{sh}(X))) \tag{5.2}$$

$F_{blob}(X)$, $F_{sm}(X)$, and $F_{sh}(X)$ are the three features mentioned above (blob, smoothness, and shape features) corresponding to the state X computed from the current observation. Details about these three features can be referred to Section 3.4.3. λ is a tunable parameter. This appearance model is learned automatically from a training set. Consequently, there are no critical parameters tuning the weights of different features. Furthermore, by utilizing the additional shape knowledge, this appearance model can differentiate lumps from the background better than the models which use regional features, $F_{blob}(X)$ and $F_{sm}(X)$, alone.

The LLD system, as described above, is implemented by the particle filter in Fig. 2.4 of Section 2.3. Some implementation details due to the practical computational concern are described here. In total, 1000 particles are used for approximating the object state distribution. For further efficiency, the object state variables a_t, b_t, and θ_t (in Eq. (5.1)) associated with the elliptical kernel configuration are discretized, and they are allowed to take values only from three predefined small domain sets as follows. $[a_t, b_t] \in \{[d_e \cdot \sqrt{2}, d_e/\sqrt{2}], [d_e \cdot \sqrt{1.5}, d_e/\sqrt{1.5}], [d_e, d_e]\}$ where $d_e \in \{40, 50, 60, 70, 80, 90\}$ represents the equivalent circular diameter in pixel. $\theta_t \in \{2\pi/s_\theta | s_\theta = 1, ..., 8\}$. In this way, the system can run on a PC with 3 GHz Intel CPU at a speed of one frame per second, which is about the current input sequence frame rate. This efficiency comes from the above setups considering

a balance between computational complexity and detection performance. By using more particles and finer object state configurations, the detection performance of the LLD system can be improved. However, the subsequent resultant low processing speed is undesired. Once the LLD system is implemented by the particle filter, a warning of a detected lump is triggered if the estimated posterior probability $P(E_t = 1|Z^t)$ (as described in Eq. (2.16)) is higher than a threshold 0.9.

To this end, the LLD system is complete and has been integrated in the oil sands mining company's online webpage for information processing as depicted in Fig. 5.2. Satisfactory results have been obtained with this system using the archived data sets, and they are presented in the following section.

5.2 Test Data Set and Evaluation

The proposed LLD system has been tested comprehensively on archived images monitoring the feed to a crusher in the dry feed preparation stage of oil sands mining (as shown in Fig. 5.1(a)). This section presents two major test results. One is conducted on a data set containing a whole month's images to provide accurate precision and recall measurements. The second one is conducted on a data set containing 13 lumps that actually caused crusher jamming events so that an interesting and meaningful performance of detecting these jamming events can be obtained. For both evaluations, the same training was done to learn the appearance model in JDT. The training set contains 160 large lump events collected during January 2010. From each large lump event, 2 or 3 images are used for extracting the features and training the appearance model in the same way as described in the experiments in Section 3.4.3.

The first data set contains the images from a double roll crusher in December 2010. From the 31 days of December 2010, only images during daytime (from 9:30 am to 3:30 pm) are selected for testing because these hours have daylight and night time is out of the focus of our work in the current stage. Therefore the dataset contains 669,600 images ($\frac{3600\ images}{hour} \times \frac{6\ hours}{day} \times 31\ days$). Visual inspection of this vast archive of 669,600 images yielded 146 large lump events as the ground

DirtTV

Last 1 hour(s) minimum	0.00 percentage
Last 1 hour(s) average	1 percentage
Last 1 hour(s) maximum	36 percentage

Feeder Speed Statistics For DRC7

Last 2 hour(s) minimum	0.00 percentage
Last 2 hour(s) average	24 percentage
Last 2 hour(s) maximum	90 percentage

Last Large Lump Detected on DRC7 (Detecting fragments larger than 1.5 meters; running from 9 AM to 3 PM)

Figure 5.2: The LLD system integrated in the online oil sands processing information webpage.

truth. This dataset contains all the images within a whole month except for the night time images. During such a long period, various unusual weather conditions

such as steam, snow, and bad lighting resulting from bright sunshine are included. Therefore, this data set is believed to be representative and realistic enough due to its wide coverage of the time period and weather conditions. Table 5.1 and Fig. 5.3 show the results of the proposed large lump detection method tested on this data set with 5 threshold values of lump sizes. When a lump's size is above the threshold, it is considered as large and should be detected. This test provides us with a quantitative measurement of the LLD system's performance via both precision and recall rates as defined in Eq. (3.24). By varying the size threshold, different pairs of precision/recall rates have been obtained. The size threshold of 61 provides a good balance ensuring a precision value above 90%. This high precision is desired because a lower precision tends to trigger more false alarms which are time and money consuming and not preferred in this specific application. With such a high precision rate above 90%, a 37% recall rate is obtained and it intuitively means that about one out of two or three lumps can be detected and proper operations can be done to prevent the potential jamming events.

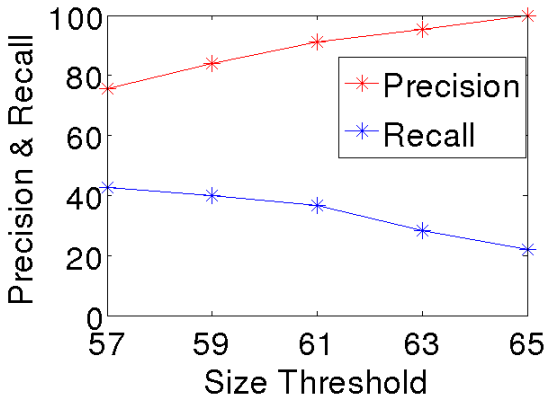


Figure 5.3: The precision and recall rates of the LLD system evaluated on the December 2010 data set.

The second data set contains all the jamming cases that occurred during day time and were caused by large lumps during January and February 2011. 13 such lumps have been found and included in this data set. This data set is interesting because these are the real cases that need to be detected since they actually caused a jamming. The large lump detection method was tested on these 13 lumps with a size threshold of 61. Recall that this value has been discussed in the previous experiment

Table 5.1: The precision and recall rates of the LLD system evaluated on the December 2010 data set.

Threshold	True Positive	False Positive	Precision	Recall
57	63	21	75.0%	43.2%
59	59	12	83.1%	40.4%
61	54	5	91.5%	37.0%
63	41	2	95.4%	28.1%
65	32	0	100%	21.9%

and believed to provide a high precision rate above 90%. Out of these 13 lumps, 7 were detected by the proposed method. This means that out of 13 important lump jamming cases, over a half could have been detected and attempts made to avoid jamming. In the meanwhile, few false alarms will be triggered given the high precision rate above 90% of the proposed large lump detection method. Although this precision rate is not obtained from the January and February images where these 13 jamming cases were collected from, it is reasonable to assume that images in January and February 2011 will not differ too much from those in December 2010 given the representativeness of December 2010 data set. The 7 detected lumps are shown in Fig. 5.4, and 6 missed lumps are shown in Fig. 5.5. We suspect that the six missed cases were not detected probably for the following reasons. First, the two features, brightness and boundary gradients, are not sufficiently prominent in some of these cases. Second, the lump appearance and motion change abruptly during these frames especially when lumps hit the side of the chute and flip. Third, portions of a lump are covered with dirt so that our method cannot get a long enough continuous boundary to extract a good shape feature.

The obtained results of the above two tests, especially the 37% recall rate of the first test might not appear as a high number compared to the recall rates in many other object detection applications. However, the achievable recall rate is not comparable across applications, and it is in large part determined by the nature of the application. Different practical difficulties can rise in real applications. For example, the experimental setup of this problem is much more hostile than those laboratory environments. Therefore, no additional sensory data can be easily ob-

tained due to the difficulty of mounting any other facilities following the company standards. In other words, the only available information is the surveillance video which is not even an ideal source for computer vision purpose compared to other videos from well-set cameras. More clearly, the surveillance video's camera angle, focus, etc. may vary arbitrarily. Therefore, a 37% recall rate of our system is reasonable and can be considered as good at the current stage. This claim is based on the following reasons. First, there is obviously no available commercial systems that can be directly applied to the LLD problem. 37% recall rate is an improvement over the existing practice (0% detection). Second, the only peer method solving this LLD problem, the classification based method [71] has been demonstrated in Section 3.4.3 to be in-comparative with our method. Third, a good number of failures to detect were caused by the large lumps being occluded by fines or appearing similar to fines (such as in Figs 5.5(a), 5.5(d), and 5.5(f)). No computer vision algorithm can detect these barely visible cases. Therefore, it is actually difficult to obtain a good result with both high precision and high recall rates. As far as the potential ideas that might help to improve the current result of our LLD system, we might be able to list the followings. For example, the classification based method [71] could be incorporated into the appearance model to potentially improve our LLD system's performance. Additionally, some other feature cues such as edges can also be tried to provide supplementary information in the appearance model, so that lumps can be better differentiated from background.

5.3 Summary

A large lump detection system is proposed and tested online. Experiments on two data sets demonstrated the significance of the proposed LLD system. First, the system is robust because it was tested comprehensively on the whole month's images and it achieved satisfactory results. Second, the system is useful and meaningful, because out of the 13 large lumps that actually caused jamming, more than a half of them have been detected by the proposed LLD system. This means that these detected jamming events could have been potentially avoided and significant pro-

duction loss would have been prevented.

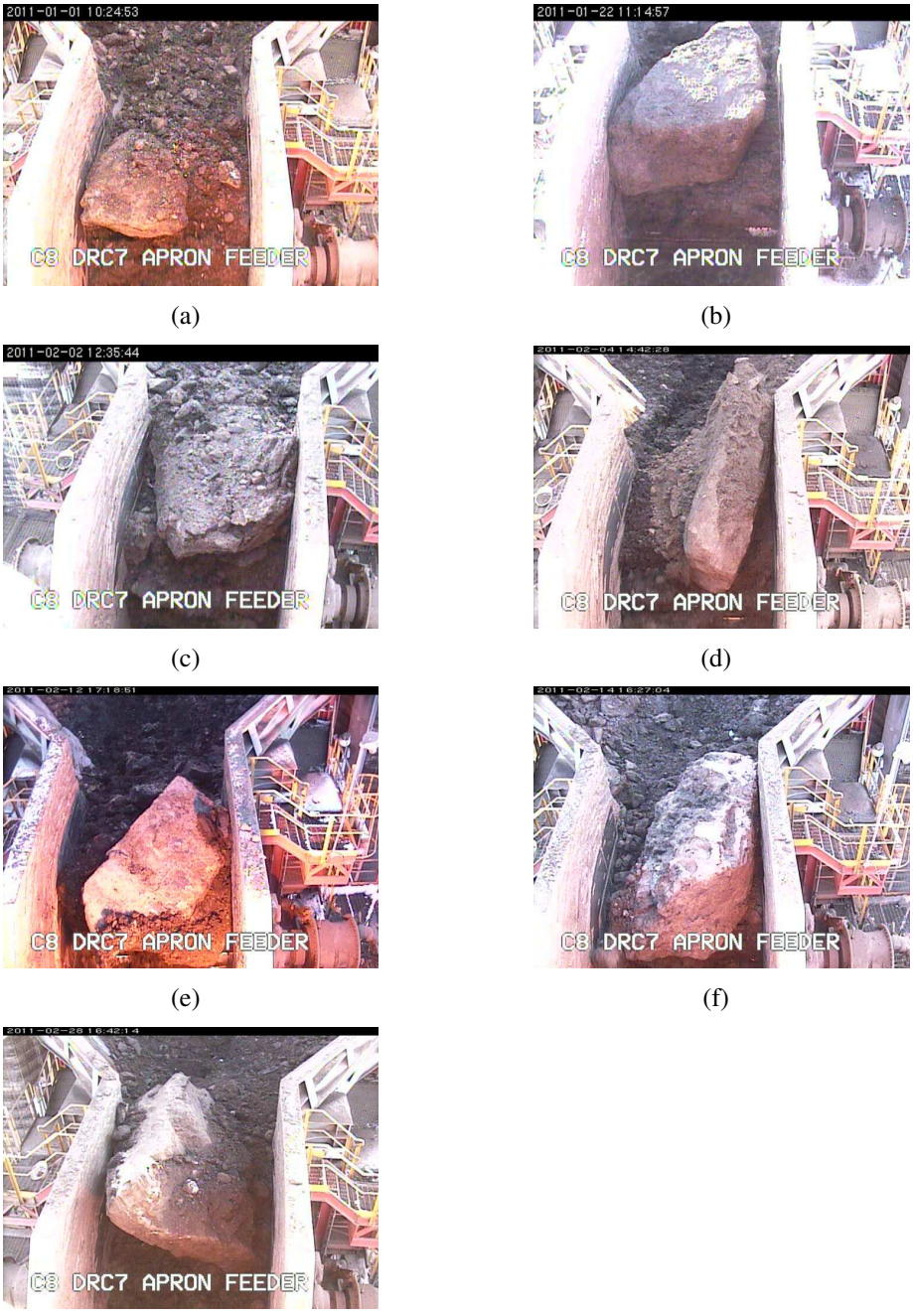
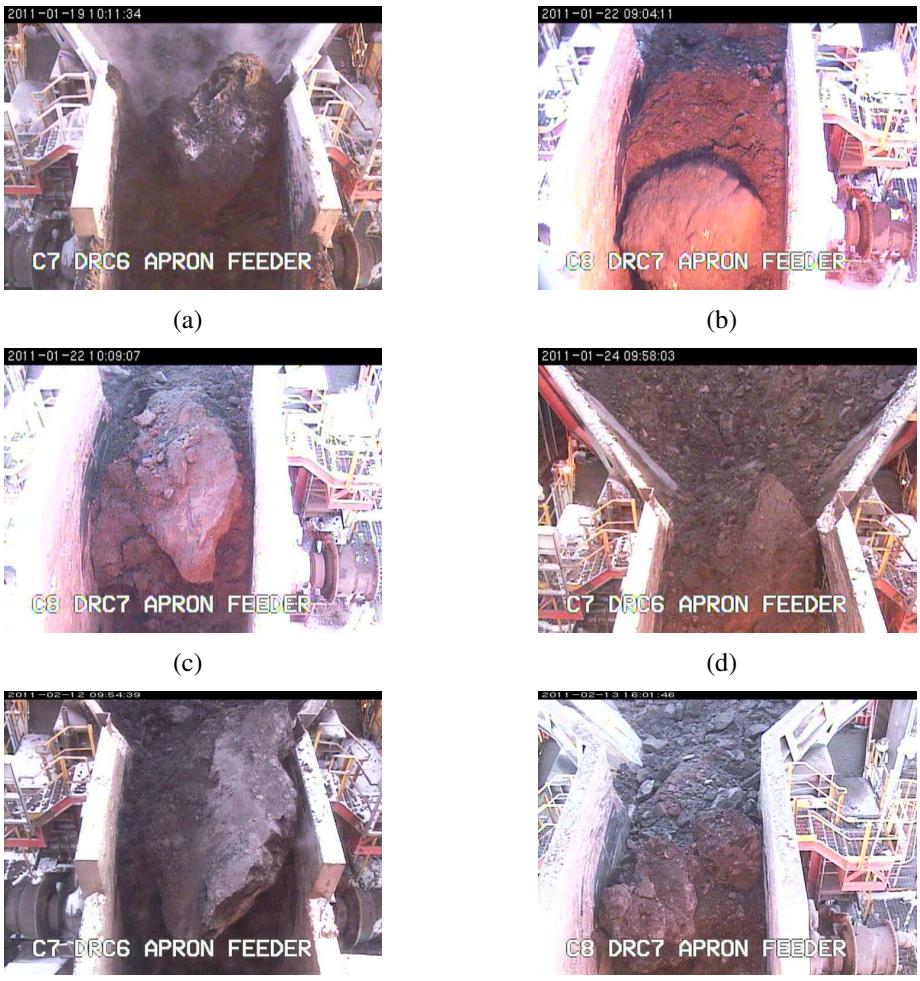


Figure 5.4: Seven large lump jamming cases that can be detected by the proposed large lump detection method.



(a) (b) (c) (d) (e) (f)

Figure 5.5: Six large lump jamming cases that cannot be detected by the proposed large lump detection method.

Chapter 6

Conclusion and Future Research

6.1 Summary

This thesis is essentially motivated by the problem of large lump detection in the feed to crushers in oil sands mining industry. To this end, a computer vision based solution addressing this problem is provided by this work. By automatically detecting large lumps with the proposed solution, precautionary warning can be provided when large lumps appear so that the risk of jamming the crushers as well as significant production losses can be minimized. From the research point of view and to achieve the proposed solution, a complete object detection system is proposed, with three independent but connected components. Three findings were drawn from the three proposed components as follows.

- The first finding is that formulating the JDT framework with multiple motion models helps detecting objects that undergo motion changes more accurately than the existing JDT methods using a single motion model. Furthermore, exploiting the correlation between motion models and object kinematic state enhances considerably the detection accuracy. Indeed, it has been established in the literature that jointly detecting and tracking objects can improve detection accuracy by accumulating evidence during tracking process. It has also been demonstrated that incorporating multiple motion models in a tracking system achieves accurate tracking results by adjusting the system's motion model dynamically according to the object's actual motion model. Therefore, by taking into account the two mentioned conclusions, one can provide

a method which detects objects that may switch between a few motion models accurately. On this basis, the correlation between the motion models and the object kinematic state can help in predicting the motion models adaptively and accurately. Therefore, an adaptive multi-motion model JDT method has been proposed in this thesis in order to provide an accurate detection of objects that change their motion models during the detection process.

- The second finding is that shape knowledge can be integrated into the appearance models of kernel based trackers to enhance their accuracy while preserving the computational advantage versus silhouette based trackers. In fact, kernel based trackers are important because they are computationally much more efficient than silhouette based trackers and therefore they are more suitable for time-critical applications. This is mainly because kernel trackers employ a primitive geometric shape to grossly locate the object while silhouette trackers aim to track the detailed object boundary. The proposed appearance model is formulated in a general way so that it can be used either in detection or tracking where objects are represented by simple kernel object models. For our specific application, objects need to be tracked/detected accurately within a limited time. Therefore, the proposed shape aided appearance model has been employed and provided considerable improvement since enough prior shape information is used in the tracking algorithm.

- The third finding is that combining DWT and Adaboost results in a new steam detection method with good accuracy and computational efficiency. DWT has mainly the advantage of efficiently extracting extensive spatial information in various scales. On the other hand, Adaboost's advantage is that it selects and combines a subset of extensive information in a useful and optimal way, learned from a training set. Therefore, sufficient information provided by the DWT applied to the input image is fed to Adaboost which extracts from it an optimal subset. Combined in a proper way, this subset allows detecting steam with high accuracy and efficiency. Computational efficiency comes from both DWT and Adaboost being linear complexity. Additionally, apart from steam

detection, this method can be applied to smoke detection, significant for early fire detection, and other real applications.

6.2 Recommendations and Future Research

In this section, we propose potential future works based on some limitations that each one of the three components of the proposed system may have.

- In the formulation of the multi-motion JDT model, there is a basic assumption which states that the set of potential motion models is known beforehand. This assumption could be relaxed if the motion models could be learned from a training set. Another problem arises because the objects of interest share the same representative appearance model. This is mainly problematic when objects need to be continuously identified especially when some objects leave the scene and others enter it during the tracking process. In applications where an object's identity is crucial, one should distinguish objects based on their appearances or other characteristics. One additional problem is related to the fact that the joint object state is, in fact, the concatenation of individual object states. This prevents the JDT method from detecting a large number of objects mainly due to computational reasons. Few researches have tackled this problem from the high dimensional object state point of view by employing, for instance, the partitioned particle filter [42] or the Gaussian Process Dynamic Models [18]. However, this still remains a difficult problem.

- About the proposed shape based appearance model, it is rather suitable for applications where the shape of objects of interest is not too complicated. To be more precise, the objects of interest are preferred to have star shapes. An object has a star shape if for any point p inside the object, all points on the straight line between the center c and p also lie inside the object [95]. Otherwise, when the proposed method extracts the object boundary, more than one point will correspond to the boundary, and then this results in ambiguities or mistakes. Similarly, when the background is cluttered, it is also difficult for the proposed method to extract the correct object boundary. Therefore,

it is desirable for the proposed method to handle non-star shapes and clutter in the future. Additionally, the proposed appearance model is expected to work in applications where objects of interest have certain consistent shape constraints. The current shape constraints that have been embedded include object shapes being normally distributed around some mean shape and object shapes satisfying certain geometric constraints such as convexity. More types of shape constraints are desired to be embedded in the proposed appearance model in the future.

- The proposed steam detection method is suitable for applications with controlled environments. Since the proposed method is based on the frequency information captured by DWT, it is quite specific to a certain dataset. Any changes of the environment, camera focus or image resolution, may have an influence on the steam detection performance. Therefore, one problem with the proposed steam detection method is that, for each different dataset new training is normally required. This is a common problem for energy based steam detection methods because features in the frequency domain are more sensitive than others such as color, steam area shape and motion pattern which do not change considerably in different scenarios. Therefore, how to combine all the aforementioned features together and more importantly how to automatically determine the most reliable ones at the current time are interesting questions for future research.

To conclude, in this thesis we provided a solution to the real problem of large lump detection in order to reduce significantly the production loss. When solving this problem, three research components have been contributed. First, an improved joint detection and tracking method is proposed to detect objects with multiple motion models more accurately than the existing JDT methods. Second, a shape based appearance model for kernel tracking is proposed to improve the accuracy of kernel based trackers and, in the meanwhile, the computational advantage of kernel based trackers against silhouette trackers is maintained. Finally, a steam detection method based on DWT and Adaboost is proposed to provide a better performance than its

peer methods when taking both accuracy and efficiency into account.

Bibliography

[1] M. S. Arulampalam, B. Ristic, N. Gordon, and T. Mansell. Bearings-only tracking of manoeuvring targets using particle filters. *EURASIP Journal on Applied Signal Processing*, 2004(1), 2004.

[2] S. Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064 –1072, 2004.

[3] B. Babenko, Ming-Hsuan Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 983 –990, 2009.

[4] Yaakov Bar-Shalom, X. Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation*. Wiley-Interscience, 2001.

[5] D. Bedekar, A. Nair., and D.G. Vince. Choosing the optimal mother wavelet for decomposition of radio-frequency intravascular ultrasound data for characterization of atherosclerotic plaque lesions. In *Medical Imaging 2005: Ultrasonic Imaging and Signal Processing*, volume 5750, pages 490 – 502, 2005.

[6] S. Bi and X. Y. Ren. Maneuvering target doppler-bearing tracking with signal time delay using interacting multiple model algorithms. *Progress in Electromagnetics Research*, 87:15 –41, 2008.

[7] M.J. Black and A.D. Jepson. Recognizing temporal trajectories using the condensation algorithm. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 16 –21, 1998.

[8] H.A.P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780 –783, 1988.

[9] Y. Boers and J.N. Driessen. Interacting multiple model particle filter. *IEE Proceedings - Radar, Sonar and Navigation*, 150(5):344 – 349, 2003.

[10] K. Branson and S. Belongie. Tracking multiple mouse contours (without too many samples). In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1039 –1046, 2005.

[11] P. Brasnett, L. Mihaylova, N. Canagarajah, Lyudmila Mihaylova, Nishan Canagarajah, and D. Bull. Particle filtering with multiple cues for object tracking. In *Proc. of SPIE's Annual Symp. EI ST in Video Sequences*, pages 430–441, 2005.

[12] Paul Brasnett, Lyudmila Mihaylova, David Bull, and Nishan Canagarajah. Sequential monte carlo tracking by fusing multiple cues in video sequences. *Image and Vision Computing*, 25(8):1217–1227, 2007.

[13] Simone Calderara, Paolo Piccinini, and Rita Cucchiara. Smoke detection in video surveillance: a mog model in the wavelet domain. In *International Conference on Computer Vision Systems*, pages 119–128, 2008.

[14] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. In *IEEE International Conference on Computer Vision*, pages 694 –699, 1995.

[15] Turgay Celik, Huseyin Ozkaramanli, and Hasan Demirel. Fire and smoke detection without sensors: image processing based approach. In *European Signal Processing Conference*, pages 1794 – 1798, 2007.

[16] Wen-Yan Chang, Chu-Song Chen, and Yi-Ping Hung. Discriminative descriptor-based observation model for visual tracking. In *IEEE International Conference on Pattern Recognition*, volume 3, pages 83 –86, 2006.

[17] M. A. Charmi, S. Derrode, and F. Ghorbel. Fourier-based geometric shape prior for snakes. *Pattern Recognition Letters*, 29(7):897–904, 2008.

[18] Jixu Chen, Minyoung Kim, Yu Wang, and Qiang Ji. Switching gaussian process dynamic models for simultaneous composite motion tracking and recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2655 –2662, 2009.

[19] Thou-Ho Chen, Yen-Hui Yin, Shi-Feng Haung, and Yan-Ting. The smoke detection for early fire-alarming system based on video processing. In *EEE International Conference in Intelligent Information Hiding and Multimedia Signal Processing*, 2006.

[20] Shi chuan Du, Zhi guo Shi, Wei Zang, and Kang sheng Chen. Using interacting multiple model particle lter to track airborne targets hidden in blind doppler. *Journal of Zhejiang University Science - Science A*, 8(8):1277–1282, 2007.

[21] Yu Chunyu, Fang Jun, Wang Jinjun, and Zhang Yongming. Video fire smoke detection using motion and color features. *Fire Technology*, 46:651–663, 2010.

[22] Yu Chunyu, Zhang Yongming, Fang Jun, and Wang Jinjun. Video smoke recognition based on optical flow. In *International Conference on Advanced Computer Control*, volume 2, pages 16 –21, 2010.

[23] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:564–577, 2003.

[24] T. F. Cootes, C. J. Taylor, , D. H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.

[25] Gregory W. Corder and Dale I. Foreman. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. Wiley, 2009.

[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

[27] Daniel Cremers. Dynamical statistical shape priors for level set-based tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1262–1273, 2006.

[28] Jacek Czyz, Branko Ristic, and Benoit Macq. A color-based particle filter for joint detection and tracking of multiple objects. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 2, pages 217–220, 2005.

[29] Jacek Czyz, Branko Ristic, and Benoit M. Macq. A particle filter for joint detection and tracking of color objects. *Image and Vision Computing*, 25(8):1271 – 1281, 2007.

[30] Jesse Davis and Mark Goadrich. The relationship between Precision-Recall and ROC curves. In *International Conference on Machine Learning*, pages 233–240. ACM, 2006.

[31] Wei Du and Justus H. Piater. A probabilistic approach to integrating multiple cues in visual tracking. In *European Conference on Computer Vision*, pages 225–238, 2008.

[32] B. Efron. Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7(1):1–26, 1979.

[33] R. J. Ferrari, H. Zhang, and C. R. Kube. Real-time detection of steam in video images. *Pattern Recognition*, 40:1148–1159, 2007.

[34] D. Freedman and T. Zhang. Active contours for tracking distributions. *IEEE Transactions on Image Processing*, 13(4):518–526, 2004.

[35] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[36] X. Gao, F. Coetzee T. Boult, and V. Ramesh. Error analysis of background adaption. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 503–510, 2000.

[37] R. Gonzalez-Gonzalez, V. Alarcon-Aquino, R. Rosas-Romero, O. Starostenko, J. Rodriguez-Asomoza, and J.M. Ramirez-Cortes. Wavelet-based smoke detection in outdoor video sequences. In *IEEE International Midwest Symposium on Circuits and Systems*, pages 383 –387, 2010.

[38] J. Gubbi, S. Marusic, and M. Palaniswami. Smoke detection in video using wavelets and support vector machines. *Fire Safety Journal*, 44(8):1110 – 1115, 2009.

[39] Dongil Han and Byoungmoo Lee. Flame and smoke detection method for early real-time detection of a tunnel fire. *Fire Safety Journal*, 44(7):951 – 961, 2009.

[40] Chao-Ching Ho. Machine vision-based real-time early flame and smoke detection. *Measurement Science and Technology*, 20(4), 2009.

[41] Chang Huang, Bo Wu, and Ramakant Nevatia. Robust object tracking by hierarchical association of detection responses. In *European Conference on Computer Vision*, volume 5303, pages 788–801, 2008.

[42] Z. L. Husz, A. M. Wallace, and P. R. Green. Tracking with a hierarchical partitioned particle filter and movement modelling. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, PP(99):1 –14, 2011.

[43] M. Isard and J. MacCormick. Bramble: A bayesian multiple-blob tracker. In *IEEE International Conference on Computer Vision*, volume 2, pages 34–41, 2001.

[44] Michael Isard and Andrew Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[45] Michael Isard and Andrew Blake. A mixed-state condensation tracker with automatic model-switching. In *IEEE International Conference on Computer Vision*, pages 107–112, 1998.

[46] T. Jaeggli, E. Koller-Meier, , and L. Van Gool. Learning generative models for multi-activity body pose estimation. *International Journal of Computer Vision*, 83:121–134, 2009.

[47] R. Jain and H. Nagel. On the analysis of accumulative difference pictures from image sequences of real world scenes. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 1, pages 206–214, 1979.

[48] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, 2002.

[49] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[50] DongKeun Kim and Yuan-Fang Wang. Smoke detection in video. In *World Congress on Computer Science and Information Engineering*, volume 5, pages 759 –763, 2009.

[51] Damir Krstinic, Darko Stipanicev, and Toni jakovcevic. Histogram-based smoke segmentation in forest fire detection system. *Information Technology and Control*, 38:237–244, 2009.

[52] Cheng-Hao Kuo, Chang Huang, and R. Nevatia. Multi-target tracking by on-line learned discriminative appearance models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 685 –692, 2010.

[53] Junghyun Kwon, Kyoung Mu Lee, and F.C. Park. Visual tracking via geometric particle filtering on the affine group with optimal importance functions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 991 –998, 2009.

[54] I. Leichter, M. Lindenbaum, and E. Rivlin. Tracking by affine kernel transformations using color and boundary cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):164 –171, 2009.

[55] Ido Leichter, Michael Lindenbaum, and Ehud Rivlin. A general framework for combining visual trackers - the "black boxes" approach. *International Journal of Computer Vision*, 67(3):343–363, 2006.

[56] Min Li, Wei Chen, Kaiqi Huang, and Tieniu Tan. Visual tracking via incremental self-tuning particle filtering on the affine group. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1315 –1322, 2010.

[57] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:79–116, 1998.

[58] Li Ma, Kaihua Wu, and L. Zhu. Fire smoke detection in video images using kalman filter and gaussian mixture color model. In *International Conference on Artificial Intelligence and Computational Intelligence*, volume 1, pages 484 –487, 2010.

[59] John MacCormick and Andrew Blake. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision*, 39:57–71, 2000.

[60] E. Maggio and A. Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 221 – 224, 2005.

[61] A. Mansouri. Region tracking via level set pdes without motion computation. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 24(7):947 – 961, 2002.

[62] H. Maruta, Y. Kato, A. Nakamura, and F. Kurokawa. Smoke detection in open areas using its texture features and time series properties. In *IEEE International Symposium on Industrial Electronics*, pages 1904 –1908, 2009.

[63] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan. Interacting multiple model methods in target tracking: a survey. *IEEE Transactions on Aerospace and Electronic Systems*, 34(1):103 –123, 1998.

[64] S. McGinnity and G. W. Irwin. Multiple model bootstrap filter for maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 36:1006–1012, 2000.

[65] Y. Meyer. *Wavelets: Algorithms and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, 1993.

[66] Yang Mingqiang, Kpalma Kidiyo, and Ronsin Joseph. *A Survey of Shape Feature Extraction Techniques*. Pattern Recognition Techniques, Technology and Applications, InTech, 2008.

[67] Farzin Mokhtarian and Alan Mackworth. Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):34 –43, 1986.

[68] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In *IEEE International Conference on Computer Vision*, pages 1305–1312, 2003.

[69] N. Nandakumaran, A. Sinha, and T. Kirubarajan. Joint detection and tracking of unresolved targets with monopulse radar. *IEEE Transactions on Aerospace and Electronic Systems*, 44(4):1326 –1341, 2008.

[70] W. Ng, J. Li, S. Godsill, and J. Vermaak. A hybrid approach for online joint detection and tracking for multiple targets. In *IEEE Aerospace Conference*, pages 2126 –2141, 2005.

[71] Sharmin Nilufar, Nilanjan Ray, and Hong Zhang. Optimum kernel function design from scale space features for object detection. In *IEEE International Conference on Image Processing*, pages 861 – 864, 2009.

[72] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *IEEE International Conference on Computer Vision*, pages 555–562, 1998.

[73] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000.

[74] Patrick Perez, Jaco Vermaak, and Andrew Blake. Data fusion for visual tracking with particles. In *Proceedings of the IEEE*, pages 495–513, 2004.

[75] Paolo Piccinini, Simone Calderara, and Rita Cucchiara. Reliable smoke detection system in the domains of image energy and color. In *IEEE International Conference on Image Processing*, pages 1376–1379, 2008.

[76] Daniel Ponsa and Antonio M. Lopez. Variance reduction techniques in particle-based visual contour tracking. *Pattern Recognition*, 42(11):2372 – 2391, 2009.

[77] K. Punithakumar, T. Kirubarajan, and A. Sinha. Multiple-model probability hypothesis density filter for tracking maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, 44(1):87 –98, 2008.

[78] P. Prez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *European Conference on Computer Vision*, pages 661–675, 2002.

[79] B. Ristic and M. S. Arulampalam. Tracking a manoeuvring target using angle-only measurements: Algorithms and performance. *Signal Processing*, 83(6), 2003.

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

[81] David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1):125–141, 2008.

[82] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2009.

[83] M. G. Rutten, B. Ristic, and N. J. Gordon. A comparison of particle filters for recursive trackbeforedetect. In *International Conference on Information Fusion*, volume 1, pages 169–175, 2005.

[84] Jakob Santner, Christian Leistner, Amir Saffari, Thomas Pock, and Horst Bischof. Prost: Parallel robust online simple tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 723–730, 2010.

[85] T. Schoenemann and D. Cremers. Globally optimal shape-based tracking in real-time. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1 – 6, 2008.

[86] J. A. Sethian. *Level set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[87] Jichuan Shi, Hong Zhang, and N. Ray. Solidity based local threshold for oil sand image segmentation. In *IEEE International Conference on Image Processing*, pages 2385 –2388, 2009.

[88] C. Stauffer and W. Grimson. Learning patterns of activity using real time tracking. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 22, pages 747–767, 2000.

[89] D. Stoyan and H. Stoyan. *Fractals, Random Shapes and Point Fields: Methods of Geometrical Statistics*. Wiley, Chichester, 1995.

[90] Hao Sun, Cheng Wang, Boliang Wang, and N. El-Sheimy. Independently moving object detection and tracking using stereo vision. In *IEEE International Conference on Information and Automation*, pages 1936 –1941, 2010.

[91] B. Toreyin, Y. Dedeoglu, and A. Cetin. Wavelet based real-time smoke detection in video. In *European Signal Processing Conference*, pages 4–8, 2005.

[92] B.U. Toreyin, Y. Dedeoglu, and A.E. Cetin. Contour based smoke detection in video using wavelets. In *European Signal Processing Conference*, 2006.

[93] Truong Xuan Tung and Jong-Myon Kim. An effective four-stage smoke-detection algorithm using video images for early fire-alarm systems. *Fire Safety Journal*, 46(5):276 – 282, 2011.

[94] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[95] Olga Veksler. Star shape prior for graph-cut image segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 454 – 467, 2008.

[96] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *IEEE International Conference on Computer Vision*, pages 734–741, 2003.

[97] Zhijie Wang and Hong Zhang. Large lump detection using a particle filter of hybrid state variable. In *International Conference on Advances in Pattern Recognition*, pages 14–17, 2009.

[98] Zheng Wei, Xingang Wang, Wenchuan An, and Jianfeng Che. Target-tracking based early fire smoke detection in video. In *International Conference on Image and Graphics*, pages 172 –176, 2009.

[99] C. Wren, A. Azarbayejani, and A. Pentland. Pfinder: Real-time tracking of the human body. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 19, pages 780–785, 1997.

[100] Z. Xiong, R. Caballero, H. Wang, A.M. Finn, M.A. Lelic, and P.-Y. Peng. Video-based smoke detection: possibilities, techniques, and challenges. In *Suppression, Detection and Signaling Research and Applications - A Technical Working Conference*, 2007.

[101] Jing Yang, Feng Chen, and Weidong Zhang. Visual-based smoke detection using support vector machine. In *International Conference on Natural Computation*, volume 4, pages 301 –305, 2008.

[102] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.

[103] Alper Yilmaz, Xin Li, and Mubarak Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, 2004.

[104] Feiniu Yuan. A fast accumulative motion orientation model based on integral image for video smoke detection. *Pattern Recognition Letters*, 29(7):925 – 932, 2008.

[105] Dengyi Zhang, Aike Hu, Yujie Rao, Jinming Zhao, and Jianhui Zhao. Forest fire and smoke detection based on video image segmentation. In *Proceedings of SPIE*, volume 6788 of *MIPPR 2007*, 2007.

[106] Hong Zhang. Image processing for the oil sands mining industry. *IEEE Signal Processing Magazine*, 25(6):198–200, 2008.

[107] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *IEEE International Conference on Computer Vision*, pages 44–50, 2003.