Foreground Region Detection and Tracking for Fixed Cameras

by

Deniz TURDU

Submitted to
the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Master of Science

SABANCI UNIVERSITY

April 2010

FOREGROUND REGION DETECTION AND TRACKING FOR FIXED CAMERAS

APPROVED BY

Assist. Prof. Dr. Hakan ERDOĞAN (Thesis Supervisor)	....
Assoc. Prof. Dr. Berrin YANIKOĞLU	....
Assoc. Prof. Dr. Erkay SAVAŞ	....
Assist. Prof. Dr. İlker HAMZAOĞLU	....
Assist. Prof. Dr. Müjdat ÇETİN	....

DATE OF APPROVAL: ....

To my family…

Acknowledgements

I would like to express my gratitude to my thesis supervisor Hakan Erdoğan for his invaluable guidance, tolerance, positiveness, support and encouragement throughout my thesis.

I would like to thank TÜBİTAK for providing the necessary financial support for my masters education.

I would like to thank my thesis jury members Berrin Yanıkoğlu, Erkay Savaş, İlker Hamzaoğlu and Müjdat Çetin.

FOREGROUND REGION DETECTION AND TRACKING FOR FIXED CAMERAS

DENIZ TURDU

EE, M.Sc. Thesis, 2010

Thesis Supervisor: Hakan Erdoğan

Keywords: background modeling, object detection, foreground segmentation, tracking

Abstract

For real-time foreground detection on videos, probabilistic modeling for background and foreground colors are widely used. Stauffer and Grimson's model is very successful for foreground segmentation. In this method, each pixel is modeled independently with Gaussian mixtures. Explicit foreground probabilities for pixels are not calculated. Spatial and temporal continuity of pixels are omitted.

In this thesis, we obtain foreground probabilities for the pixels using Stauffer and Grimson's model and apply hysteresis thresholding to utilize spatial continuity of pixels. For the same purpose, we also use Markov Random Field modeling and optimizations. To leverage the temporal continuity of pixels, mean-shift tracking is integrated into the segmentation to increase accuracy. Wherever applicable, we combine some of these improvements together. Our work shows that using the probabilistic approach with different enhancements results in much higher segmentation accuracy.

SABİT KAMERALAR İÇİN ÖNPLAN BELİRLEME VE NESNE TAKİBİ

DENİZ TÜRDÜ

EE, Yüksek Lisans Tezi, 2010

Tez Danışmanı: Hakan Erdoğan

Anahtar Kelimeler: arkaplan modelleme, nesne tanıma, önplan çıkarımı, nesne takibi

Özet

Gerçek zamanlı önplan tanıma ve ayrıştırma uygulamalarında, önplan ve arkaplan modelleme onemli bir yer teşkil etmektedir. Stauffer ve Grimson metodu, önplan çıkarımında yaygın olarak kabul görmüş başarılı bir metottur. Bu mettota, her piksel ayrı ve bağımsız bir Gauss karışımı ile modellenir. Piksellerin önplan olma olasılıkları doğrudan kullanılmaz. Zamansal ve mekansal süreklilik göz ardı edilir.

Bu çalışmada, Stauffer ve Grimson metodundan hareketle, piksellerin önplan olma olasılıklarını belirleyip histeresiz eşikleme yaparak mekansal süreklilik bilgisini kullandık. Aynı amaçla, Markov Rasgele Alanları temelli modelleme ve optimizasyon uyguladık. Zamansal devamlılık bilgisini kullanabilmek için de; ortalama kaydırma metodu ile nesne takibini, önplan ayrıştırmaya dahil ettik. Uygun olan durumlar için birkaç metodu birlikte kullandık. Çalışmamızda, önplan belirleme başarımını önemli derecede arttırdık.

Contents

List of Figures

List of Tables

Abbreviations

BP	**B**elief **P**ropagation
CPU	**C**entral **P**rocessing **U**nit
FPS	**F**rame **P**er **S**econd
GMM	**G**aussian **M**ixture **M**odel
GMRF	**G**aussian **M**arkov **R**andom **F**ield
HSV	**H**ue **S**aturation **V**alue
ICM	**I**terated **C**onditional **M**odes
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
ISG	**I**ndependent **S**tauffer **G**rimson
LBP	**L**ocal **B**inary **P**attern
MRF	**M**arkov **R**andom **F**ield
PC	**P**ersonal **C**omputer
PETS	**P**erformance **E**valuation of **T**racking and **S**urveillance
RGB	**R**ed **G**reen **B**lue
VSSN	**V**ideo **S**urveillance and **S**ensor **N**etworks

Chapter 1

Introduction

1.1 Motivation

Object detection and segmentation are primary contexts for image processing studies which address many practical problems. Real world applications are spread on a wide spectrum and only some of these applications are: face detection/recognition, plate detection/recognition, automated character detection/recognition, surveillance, medical image analysis and sports games analysis. Multiple constraints exist for segmentation based applications. Simply, false object detections and missed objects might cause critical problems where there is a strict need for a very precise segmentation, like in medical imaging and surveillance applications. There are different implementations for this variety of problems. However, the main purpose in all these different methods and implementations is the same: segmenting target foreground object(s) out of the background robustly, meeting the required constraints.

Stauffer and Grimson's model is a widely accepted and used model which segments foreground out of the background using a mixture of Gaussians to represent per pixel color distributions. These Gaussians keep track of the historical color observations on each pixel and they are updated online with every video frame. Segmentation decision for each pixel is made separately, and the only criteria in segmentation decision is the weights of the Gaussian components within a mixture. A direct weight based thresholding, instead of an explicit class probability based thresholding, is applied for the Gaussian components for each pixel value observed. Foreground probability for pixels is not calculated. In addition, all pixels are modeled independently and spatial correlation is omitted. Therefore, fragmentation occurs in foreground detection. Moreover, foreground is not considered as a whole. Foreground location is not tracked and used. This causes many false detected pixels which are actually outliers.

In this thesis, we follow a different approach on this color model to address mentioned problems and increase segmentation accuracy. We use the ISG model, but instead of obtaining binary class labels for pixels directly, we introduce an intermediate process of calculating class probability values for the foreground class for each pixel. Then we exploit these foreground probabilities to enable different enhancements on the segmentation. First of all, a foreground probability based thresholding on pixels which reside on the vicinity of the convex hulls of foreground fragments detected by the ISG model, is applied to recover for the misses of the ISG method. We also use standard post-processing smoothing operations and in addition, we incorporate foreground edge -the gradient- information into the model for this first enhancement method, which is called "hysteresis thresholding". Hysteresis thresholding performs well to detect missed pixels and deals with the fragmentation problem of the ISG model which results in a poor performance in detecting foreground precisely and the method provides a clear identification of the number of foreground objects and the boundaries for each. Incorporation of the edge information acts as an indirect use of gradient features of the image, since the base model features are only the RGB color values.

As a second and a different enhancement, we apply Markov Random Field (MRF) smoothing by leveraging foreground class probabilities for the pixels as the data consistency terms to be used in the MRF model together with the smoothness criteria. MRF based method compensates for the independence assumption of the base model. This enhancement brings a better performance for segmentation, removing the wrong class labels on misclassified pixels by analyzing neighbor pixels, which makes it possible to detect the foreground as a whole and eliminate unexpected outlier foreground detections.

As the final enhancement, we incorporate mean shift tracking into the color model, to improve segmentation accuracy even more. Mean shift runs in parallel with the ISG model and informs the model on the location and movement rate of the foreground object. This is used in situations where the foreground object remains stationary for a long time in the scene and blends into the background of the ISG model. Since the location and movement rate is known by the mean shift, stationary foreground pixels will not be segmented as background, which is the case for the pure ISG model. The location information is also used for outlier elimination. Mean shift tracking is used together with the previous enhancement methods we propose, to compare the segmentation improvements versus the computational load and to be able to select the best option.

As another method in this thesis, not for the purpose of increasing the segmentation performance using tracking but for creating a robust tracker within the tracking context; we also use foreground probabilities calculated from the ISG model as a weight image for mean shift based object tracking. Since the ISG model assigns precise foreground probabilities to the pixels in our approach, this forms a robust weight image for a mean shift tracker and is shown to perform well compared to traditional and classifier based mean shift trackers.

The next section provides background information on modeling pixel colors in videos and scene background modeling studies.

1.2 Previous Work

1.2.1 Background Modeling

Some well known methods that are used in segmentation problems are thresholding based on colors, split and merge, connected component analysis, feature based clustering, frame comparison with a reference and frame subtraction and correlation methods to match target objects in an image [1]. There are also model based approaches. For instance, classification of pixels according to a presumed feature distribution for target objects or non-target objects is a model based segmentation technique. In foreground detection methods, the target to segment out is referred as the foreground, and the remaining non-target objects form the background. Therefore, the problem in this thesis is a segmentation problem taking two object classes into consideration: foreground and background.

There are numerous studies on modeling based segmentation especially when the observation is a set of image frames: videos. Modeling techniques try to obtain a robust representation of features of the background objects in a video instead of the foreground, since background features are more stationary. In a video, foreground features are usually harder to generalize and model, since these features tend to diverge more. Therefore, background is modeled statistically and used for estimating the labels in the image. There are usually two classes in background modeling into which each pixel should be classified: foreground and background. In some studies, besides these two classes, there are other classes like the shadow class [2, 3].

Most of the background modeling techniques assume that observations are made by a fixed camera [3–6]. The intuitive reason behind this is the fact that each pixel can then be referred with the coordinate, which always contains the same part of the scene. Hence, the movement of the camera does not possess a problem which would otherwise make it necessary to introduce intermediate steps to create location independent models which cannot rely on historical color observations per pixel easily.

Modeling can be adaptive or non-adaptive. Non-adaptive modeling forms a static model for the background. This model itself is usually a pre-observed frame, or the first frame, or any frame in an observed scene. It may also be a static average of some of the frames in the video. In short, the model is not updated at all. A single frame to represent background colors is not a tolerant and flexible way. Single frame based modeling rarely results in acceptable performance where background is not changing. That is the main reason why more complex, statistical -parametric or non-parametric- background color models are used much frequently in foreground segmentation. Kalman filter approach in [7] deals with the sudden illumination changes in the background, but it is also subject to a slow background recovery problem. When background moves, it takes very long for the Kalman filter to recover from that false segmentation. The method described in [8] does not have this problem. It uses a dedicated Gaussian distribution per pixel over the color values to represent the background colors on that pixel. Gaussian modeling of the background was first presented in this work. Every pixel is completely independent from others, and each pixel has its own Gaussian distribution function. Mean and variance of a Gaussian is updated when a new frame is received such that each frame contributes to the background model with a configurable learning rate. There are no foreground probabilities or background probabilities calculated. Only the color range of the Gaussian function is used for the binary classification in this approach. Major weak point of this approach is the fact that background color distribution for a pixel can not always be represented by a single Gaussian distribution. As a good example, visualize an outdoor scene: at day and at night, or at different seasons of the year, under different lighting conditions of the scenery, a pixel will have very different background colors. As another example, there might be some periodic motion in the scene. Background objects such as the leaves of a tree moving in the wind, or waves in the sea periodically rising are examples of movement in the background. In these examples, some of the background pixels will have color values concentrated around multiple means. In short, single Gaussian distribution is not reliable for cases where background is dynamic.

Instead of using a single Gaussian distribution, a mixture of many Gaussians is used for a single pixel to represent colors observed on that pixel in Stauffer and Grimson's well known model [9]. Single Gaussian based method in [8] is not as robust as Stauffer and Grimson's mixture of Gaussians based model, in terms of the segmentation performance. Each pixel is still independent in this mixture based approach. For this reason, we will refer to this model as the Independent Stauffer-Grimson (ISG) model. Multivariate Gaussian components in each mixture for each pixel, operate on RGB color space on three dimensional color vectors. In our work, we used this model as our base model and increased its segmentation accuracy with various robust approaches.

In ISG model, each component of a Gaussian mixture has a different weight. The model tries to obtain a label image, based on a single assumption: for any pixel, background is observed more frequently than foreground. High weighted Gaussian functions in a mixture represent dominant colors which are frequently observed on that pixel. Thus, the highest weighted Gaussians represent background colors. This is why the ISG model is referred as a background model, even though it does not model the background explicitly. There are no separate background or foreground color models in ISG. There are no class probability values assigned for pixels. Classification decision is very similar to that in [8]: ISG is only used for making a binary decision which is not directly and explicitly based on class probabilities.

There were similar approaches to ISG. In one of them [2], there are 3 different Gaussian functions in a mixture per pixel, and each Gaussian directly represents a different class. Three classes used are: background, foreground and shadows. These Gaussians are updated with every frame in the video just like in the ISG model, but the decision process differs from that of ISG. The most likely class for a pixel is selected as the label;

thus class probability values are directly and explicitly used in this work for decision making.

There are many other studies heading onwards from the ISG model. There are some local enhancements on the ISG model like shadow suppression [4]. One important study presents pre-processing and post-processing modifications to the ISG model [10]. In this study, the image is divided into blocks since it is computationally costly to analyze every pixel value. Classification is still binary: a label can be foreground or background. However, in this method, labeling is done for each pixel block. To phrase it in a simple way, if a block has no significant changes, no calculations are needed on that block. The block can directly be assumed to be the background. This decreases the number of CPU cycles significantly, but may obviously reduce pixel classification precision. A similar block based background model can also be seen in [5].

In the ISG model, all pixels are assumed to be independent of each other in terms of their colors and also their class labels. Color value of a pixel has no effect on others' color, just as the label value on a pixel does not have any effect on others' labels. In consistency with this assumption, every pixel has its own mixture of Gaussians. In practice, pixels are expected to be correlated with each other, intuitively at least with their local neighbors. To leverage this correlation, Markov Random Field (MRF) based segmentation and smoothing is widely used. We also employ MRF models in our work. The next section is on the previous studies on MRF based image modeling and segmentation.

1.2.2 Markov Random Fields Based Segmentation

Markov Random Fields smoothing on images and segmentation are explained in detail by Perez in [11]. Using MRFs to exploit correlations in pixel neighborhoods was first presented by Geman, in his paper combining statistical physics rules with image analysis [12]. In this paper, Gibbs distribution assumption hence Gibbs distribution is used for MRFs. Differently from the ISG model, conditional pixel probabilities are calculated by considering local neighbourhood relations between the pixels. Joint distribution of all random variables is reduced and represented by the local conditional probabilities of pixels. MRF models try to obtain a joint data energy and a joint smoothness energy for the overall image labeling and try to find the optimum labeling instance which results in minimum total energy, in other words the maximum labeling probability. For this, a data consistency energy can be produced using a model that defines the probabilistic relations between the observed data and the underlying labels. Some studies utilized single or multiple dimensional Gaussians as the model between the data and the labels within an MRF context [13, 14]. In studies like this, MRFs with Gaussian functions for the data are sometimes referred as GMRF models. Data energy in our MRF model is inherited from the probabilistic output of the ISG model. There can also be different smoothness energy models like the Ising model [15]. To find the optimum labeling in MRF context, different optimization techniques exist.

In Besag's work, Iterated Conditional Modes (ICM) algorithm, which is a greedy method for solving and optimizing MRFs, was presented [16]. ICM can sometimes obtain a global maximum, resulting in an optimal labeling. But, if the function being optimized is not convex, ICM will obtain local maximum values for the label probabilities, instead of the global maxima. ICM is explained later in this thesis. The problem with ICM is also well explained in Deng's work [17].

Another MRF optimization technique is Belief Propagation (BP) introduced by Judea Pearl [18] which takes a different approach to obtain an optimal result and attacks the sub-optimality problem as referred in [19]. This method relies on a trivial message passing algorithm between the pixels. Every pixel warns its surrounding pixels about its point of view on the neighbours. This method does not also guarantee to find the globally optimal labeling. BP is explained in detail later in this thesis. There are different types of BP algorithms used in image segmentation based on MRF optimization. In our work, we used Loopy Belief propagation to obtain the most probable labeling.

There are other MRF optimization techniques like Graph-Cut based algorithms [20, 21] which may result in better optimization for some cases. However, as indicated in [22], these methods are computationally costly. In addition, loopy belief propagation based optimization has a similar peak precision-recall performance to graph cuts.

Besides MRF based optimization for smoothing, we also integrated the location information of the foreground into the segmentation. To track the object and find the location of the foreground, we use mean shift based tracking. The next section introduces studies on mean shift algorithm and object tracking based on mean shift.

1.2.3 Mean Shift Tracking

Mean shift algorithm was introduced in [23] to find the mean of a probability density. It is used to calculate the center of gravity for a given cluster of weights (densities). Mean shift based tracking in image processing context was introduced in [24] and used in many tracking studies like [25], [26], [27] and [28]. In tracking, weights correspond to target object probabilities for pixels. In short, in image processing context, mean shift is used to find the mean of the target object being tracked in the video.

Mean shift based tracking needs a probability distribution to be used as weights. In image processing context, the weights are provided in the format of an image, containing individual weight values for each pixel. This distribution image is referred with different terms like “confidence map” or “weight image” in different studies on mean shift. In traditional mean-shift based tracking, color based features of the target object are used to form a weight image. A backprojection of the histogram of target object’s HUE color features is used to assign target probabilities to all pixels in [28]. In Avidan’s work [27], instead of characterizing target colors to form the weight image, classifiers and discriminative learning is used. A group of classifiers -called the ensemble- is trained on both target and non-target colors. This discriminative way outperforms traditional mean-shift trackers which only rely on the target model. Classifiers are then used to assign target probabilities to each related pixel. In all these different methods, the purpose is to create a robust weight image where each value on any pixel is a weight. The weight is actually the target probability value for a pixel, showing how likely the pixel is to belong to the tracked target. Mean shift iterates on this probability image to find the center of the weights inside a given video frame. The initial location of the object should be provided to the mean shift tracker manually at the first frame in which the foreground object becomes visible.

We utilized a decision tree classifier based mean shift tracker, similar to Avidan’s proposed tracker. Location and movement rate of the target foreground object is fed into the ISG model by the tracker. This extra information proves to be useful in scenarios where very stationary foreground objects exist. We combine the information retrieved from the tracker, which is actually a foreground probability for each pixel depending on the tracking; with the information retrieved using the ISG model. If the foreground is stationary, then the tracker information automatically becomes dominant in this combination. Otherwise, the ISG model probabilities for pixels are dominant in decision making. In short, outliers are eliminated, the problem of stationary foreground blending into the background is resolved and the segmentation performance is significantly improved with the incorporation of mean shift tracker.

1.3 Problem Definition

In this thesis, the main purpose is to segment foreground regions in a video. ISG model is used as the base color model for the pixels observed by a stationary camera. 3 channel video images (RGB) are taken into consideration. Saying that an observed frame $\boldsymbol{X}$ is an input, a segmentation method should produce a label image $\boldsymbol{Y}$. Some features can be obtained from $\boldsymbol{X}$, and let us say that $\boldsymbol{F}$ is the features matrix. The main purpose in all segmentation techniques is common: how to come up with $\boldsymbol{Y}$ such that $P(\boldsymbol{Y}|\boldsymbol{F})$ is very high. $P(\boldsymbol{Y}|\boldsymbol{F})$ is the conditional probability of the labels $\boldsymbol{Y}$, given the features $\boldsymbol{F}$ of the observation $\boldsymbol{X}$. In our study, and in most of other studies in the field, features are actually the observed RGB colors. Thus, the problem is to maximize $P(\boldsymbol{Y}|\boldsymbol{X})$.

In generic terms, we try to propose solutions to output a labeling $\boldsymbol{Y}$, which maximizes $P(\boldsymbol{Y}|\boldsymbol{X})$. There are two classes; foreground and background. Then the optimized labeling decision is binary; for all pixels, to select a combination of label values where a label y_s for a pixel $s = (i, j)$ can either be 0(background) or 1(foreground). This problem definition is also summarized in Figure 1.1.

In ISG model, segmentation decision is made based on the mixture of multivariate Gaussians for pixel s. The decision to find $y_s|\boldsymbol{x}_s$ on pixel s is independent of the decision to find $y_r|\boldsymbol{x}_r$ on pixel r, where $r \neq s$. All pixels are treated independently. Each pixel has its own mixture of Gaussians. This approach completely neglects the temporal and spatial correlation between neighbor pixels. Due to this fact, foreground is usually detected in a fragmented format.

ISG does not specify a way to calculate $P(y_s|\boldsymbol{x}_s)$ or $P(\boldsymbol{Y}|\boldsymbol{X})$. Instead, within every mixture, only color ranges and weights of each Gaussian component is used for binary decision making.

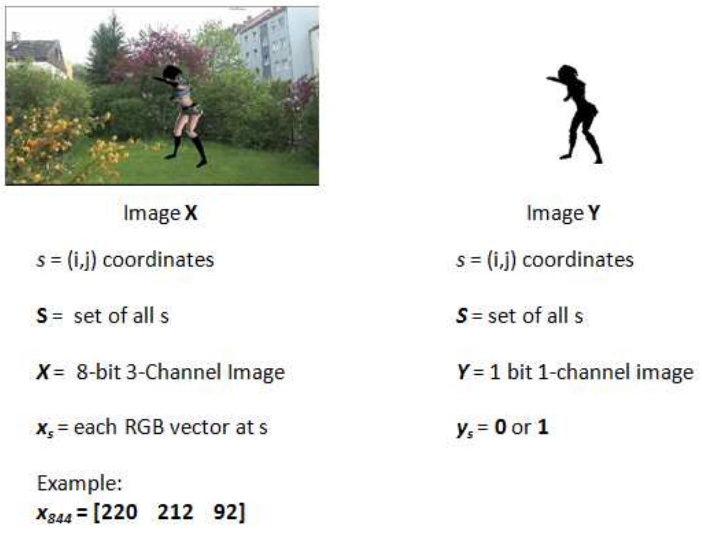


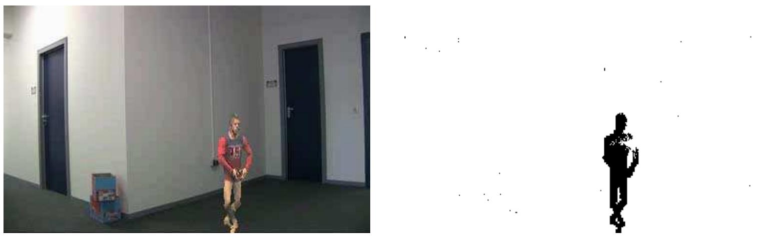
Image **X**
s = (i,j) coordinates
S = set of all s
X = 8-bit 3-Channel Image
$\boldsymbol{x_s}$ = each RGB vector at s
Example:
$\boldsymbol{x_{844}}$ **= [220 212 92]**
Image **Y**
s = (i,j) coordinates
S = set of all s
Y = 1 bit 1-channel image
$\boldsymbol{y_s}$ = **0** or **1**

Figure 1.1: Problem Definition

In our work, we follow a different approach than the base ISG model segmentation. We do not directly make a binary decision y_s for pixel s just by looking at the Gaussian mixture components' weights for s. Instead, we create an intermediate step. We obtain the target probability value v_s for each pixel s before any classification. v_s is the probability, also the weight value, which resides within the interval $[0, 1]$. To be more precise, we form a probability image $\boldsymbol{V}_I$; and every value v_s on $\boldsymbol{V}_I$ on pixel s is the foreground (target) probability of pixel s, given the color values observed on the pixel. Therefore, $v_s = P(y_s = 1|\boldsymbol{x}_s)$. We can also calculate the background probabilities $P(y_s = 0|\boldsymbol{x}_s) = 1 - v_s$, but this is not explicitly used since we concentrate on the foreground probabilities.

As seen in Figure 1.2, ISG directly results in Figure 1.2(b). Foreground probability values are not calculated. The mixture is only used to identify dominant color values frequently observed for a pixel. Our method, on the other side, obtains an intermediate image representing foreground probabilities for all the pixels, A sample probability image like this can be seen in Figure 1.2(c). Background probability image is not taken into consideration explicitly, since we try to detect foreground.

With this probability image approach, different enhancement options are made available. These probabilities are used as data probabilities in MRF based segmentation. In addition, a secondary probability threshold mechanism is applied on the probability image to make the segmentation more robust on situations where ISG decision criteria



(a) Video (b) ISG Detection

(c) Probability Image

Figure 1.2: Foreground Probability Image Sample

fails. Location information leveraged via mean shift tracking can be combined with a new tracker which operates on this probability image, to have a better estimation of the location of foreground. All these enhancements can be combined probabilistically, whenever appropriate.

1.4 Contributions of this Thesis

This work presents a way of forming a foreground probability image using the ISG method, instead of directly classifying pixels into two classes using their color models. Leveraging this, different improvements as explained below are realized.

- A method of utilizing the ISG model to inherit conditional class (foreground and background) probabilities for pixels and forming a probability image $\boldsymbol{V}_I$ which contains all foreground probabilities for every pixel s is introduced.
- Within ISG classification decision, a secondary thresholding on pixels based purely on their foreground probabilities given in $\boldsymbol{V}_I$ is introduced. This alleviates the fragmentation problem in ISG method, and increases the recall rate of the model significantly.
- An MRF based segmentation which utilizes the ISG model as the data model and which introduces smoothness constraints, is presented. A new way of calculating

foreground probabilities using this MRF model together with Belief Propagation optimization is introduced. A probability image, $\boldsymbol{V}_M$, containing foreground probabilities calculated via the MRF model is provided. The MRF model compensates for the independence assumption in the ISG model, and it outperforms ISG based segmentation, by increasing foreground detection accuracy significantly.

- Using a tracker which utilized a decision-tree based classifier, we incorporate location and tracking information into segmentation. The tracker provides a pure classifier based foreground probability image $\boldsymbol{V}_T$, together with the location and the movement rate of the foreground objects. Using this information, a robust outlier elimination is performed. In cases where foreground is very stationary, we also prevent the foreground to be classified as background by the ISG model using this additional information.

- $\boldsymbol{V}_I$ or $\boldsymbol{V}_M$ probabilities and the tracker probability image, $\boldsymbol{V}_T$, are averaged using dynamic weights that depend on the movement rate of the foreground object. If the foreground is stationary, the weight of $\boldsymbol{V}_T$ increases in these combinations to prevent foreground blending into the background, which would be the case if $\boldsymbol{V}_I$ or the $\boldsymbol{V}_M$ is dominant for the segmentation decision, since these probabilistic models do not consider the location and movement of the target foreground.

- Not as a segmentation method, but as a robust tracker, it is also shown that $\boldsymbol{V}_I$ and $\boldsymbol{V}_M$ can be well used as input weight images for mean shift trackers. This new tracker utilizing $\boldsymbol{V}_I$ or $\boldsymbol{V}_M$ can be combined with traditional or classifier based trackers, to achieve much better tracking performance in cases where well-known trackers tend to experience failures.

1.5 Outline of Thesis

This thesis is divided into 7 chapters, Introduction being the first one. In Chapter 2, detailed explanations on the ISG model is given and the way to obtain the foreground probability image $\boldsymbol{V}_I$ from the ISG model is shown. The next chapter, Chapter 3 explains the new thresholding technique, hysteresis thresholding, with some additional post-processing operations performed on the ISG model. In Chapter 4, the details of MRF smoothing with Iterated Conditional Modes (ICM) and Belief Propagation (BP) optimization methods are provided. Forming the MRF probability image $\boldsymbol{V}_M$ is shown in this chapter. Our final segmentation method which integrates mean shift tracking is explained in Chapter 5. How the tracker based probability image $\boldsymbol{V}_T$ is obtained and combined with the ISG model's probability image $\boldsymbol{V}_I$ or the MRF model's probability image $\boldsymbol{V}_M$, is also shown in this chapter. Chapter 5 also shows how we use the ISG model or the MRF model probability images for the purpose of creating a robust mean shift tracker. All parametric results for these different segmentation methods tested on different videos are given in Chapter 6. In this results chapter, besides segmentation experiments, the performance evaluation for the ISG/MRF based object tracker is given in the last section. Discussion on possible future improvements is explained in the last chapter, Chapter 7.

Chapter 2

Background Modeling

In this chapter, we first explain how the ISG model works. ISG was introduced in [9] and it has been one of the most successful methods in the algorithm competition of 4th ACM International Workshop on Video Surveillance and Sensor Networks (VSSN'06) [29]. Once the model is explained, we will show how we obtain conditional foreground and background probabilities using the ISG model. Segmentation results of the base model are provided in Chapter 6 for being able to compare the performance results to those of other methods. However, discussion and expectations on the results can be found throughout this chapter.

2.1 A Dynamic Background Model

ISG utilizes a color model to obtain a label image $\boldsymbol{Y}$ containing class labels for each pixel $s = (i, j)$ for a 3-channel observation image $\boldsymbol{X}$. Considering that $\boldsymbol{S}$ is the whole image lattice, we can say that $s \in \boldsymbol{S}$. $\boldsymbol{Y}$ contains label values y_s and $\boldsymbol{X}$ contains observed color vector $\boldsymbol{x}_s$ on pixel s. There are only two classes, foreground and background; hence y_s can either be 1 (foreground) or 0 (background). Therefore, it can be said that for $\boldsymbol{Y}$, $y_s = \boldsymbol{Y}(i, j)$.

In ISG, a dedicated and independent mixture of Gaussian distribution functions is created for each pixel s. This mixture, in a sense, keeps track of the historical color changes on that pixel. These functions are multivariate, operating on the 3-dimensional RGB color space. In a single mixture, there can be many Gaussians. Since the mixture itself

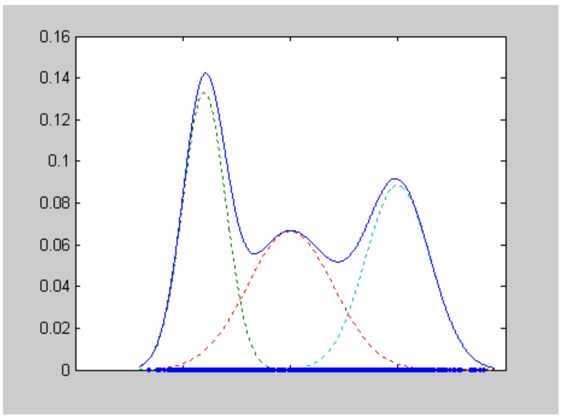


FIGURE 2.1: An Example: 1-D Mixture of Gaussians

is the color distribution for a pixel, the sum of weights of all individual Gaussians within the mixture adds up to 1. A mixture of Gaussians on a single dimensional color space with 3 Gaussian components is shown in Figure 2.1.

Every Gaussian function within a mixture can have a different weight. Some of the Gaussian components will have higher weights than the others inside the mixture. The assumption that ISG model relies on is that background always dominates the video. To be more specific, let us consider the mixture for pixel $s = (i, j)$. Many different color vectors $\boldsymbol{x}_s$ will be observed on s during the video at different times. Some of these vectors will belong to foreground, and the rest to the background. ISG assumes that for any pixel s, it is mostly the background that is observed. For most of the time, a pixel is expected to be covered with background objects, hence with the background colors. Foreground objects are not expected to appear very frequently on a pixel, compared to background. This assumption impacts the ISG model, since the weights of Gaussians in a mixture are updated with every video frame according to this assumption. For an observation $\boldsymbol{x}_s$, if one Gaussian in the mixture for pixel s can represent $\boldsymbol{x}_s$, then the weight of that Gaussian will increase compared to other components in the mixture. This means, the weights of all other components will decrease. In other words, if $\boldsymbol{x}_s$ falls within a confidence interval of one of the Gaussians in the mixture, ISG model assumes that $\boldsymbol{x}_s$ color vector is already represented by that Gaussian. Combining this matching rule with the model's assumption of frequent background, highly weighted mixture components are more likely to represent the background colors for the pixel. These components' weights have increased due to the fact that these colors represented with the components have been observed very frequently. This is the sole assumption and rule that the ISG classification depends on.

Then the decision is intuitive to make. If the observation value $\boldsymbol{x}_s$ falls in the confidence interval of one of the highest weighted components, then pixel s is classified as background. Otherwise, that pixel is probably not a background pixel. Then the problem of how to separate the components of a mixture by analyzing the weights emerges. How can we separate high weighted components in a mixture and what defines a "high weight" value? This question will be answered later in this section.

Our calculations and formulas are shown for a single Gaussian Mixture belonging to a single pixel s. These formulas do apply for every pixel and their mixtures. Thus, s indices are dropped from the variables in formulas in this chapter for simplicity. However, it should be kept in mind that these operations are valid for all Gaussian mixtures for all s.

In ISG, the probability of observing a color vector value $\boldsymbol{x}_t$ at time t and pixel location $s = (i, j)$ is modeled as:

$$P(\boldsymbol{x}_t) = \sum_{n=1}^{K} w_{n,t} * \mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\mu}_{n,t}, \boldsymbol{\Sigma}_{n,t}). \tag{2.1}$$

In ISG, a mixture for a single pixel can be composed of at most K different components. Some of these K components have higher weights and they represent background color distribution. Selection of the parameter K depends on user choice, but in [9] where the model was introduced, and also in many other studies on ISG, this value is taken to be 5. The probability of observing color value $\boldsymbol{x}_t$ at time t, is the sum of distribution values of these K different components in the mixture for the pixel. $\boldsymbol{\mu}_{n,t}$ is the mean value of the n^{th} Gaussian component in this mixture at time t, and $\boldsymbol{\Sigma}_{n,t}$ is the covariance matrix for this component. $w_{n,t}$ is the weight of the same Gaussian component. $\mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\mu}_{n,t}, \boldsymbol{\Sigma}_{n,t})$ represents n^{th} Gaussian distribution value for $\boldsymbol{x}_t$. All these parameters are time-dependent, and updated with every new frame in the video. This update mechanism will be explained in the following paragraphs.

Saying that $\mathcal{N}$ represents a Gaussian distribution, for the n^{th} component in the mixture, we can state that the multivariate color probability distribution for $\boldsymbol{x}_t$ can be calculated as:

$$\mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\mu}_{n,t}, \boldsymbol{\Sigma}_{n,t}) = sqrt\{\frac{1}{(2\pi)^d |\boldsymbol{\Sigma}_{n,t}|}\} \exp\left[-\frac{1}{2}(\boldsymbol{x}_t - \boldsymbol{\mu}_{n,t})^T (\boldsymbol{\Sigma}_{n,t})^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu}_{n,t})\right]. \quad (2.2)$$

Since we are using RGB color model, d value above is 3 in our implementation. $\boldsymbol{\mu}_{n,t}$ is the d dimensional mean vector and $\boldsymbol{\Sigma}_{n,t}$ is the $d \times d$ sized covariance matrix. In this model, for reducing calculation complexity on the inverse operation applied on the covariance matrices, d-dimensional pixel value vectors are assumed to have a diagonal covariance matrix. This relies on the assumption that different color channels are uncorrelated (e.g. R,G,B color channels are taken to be independent). In addition to this assumption, variance values for different channels are taken to be equal, for being able to write $\boldsymbol{\Sigma}_{n,t}$ in the form $\boldsymbol{\Sigma}_{n,t} = \sigma_{n,t}^2 I$, where $\sigma_{n,t}^2$ is the variance of a single channel at time t. Then the equation 2.2 simplifies to:

$$\mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\mu}_{n,t}, \boldsymbol{\Sigma}_{n,t}) = sqrt\{\frac{1}{(2\pi)^d \sigma_{n,t}^3}\} \exp\left[-\frac{1}{2\sigma_{n,t}^2}(\boldsymbol{x}_t - \boldsymbol{\mu}_{n,t})^T (\boldsymbol{x}_t - \boldsymbol{\mu}_{n,t})\right]. \quad (2.3)$$

In a mixture for a pixel, each component has its mean, variance and weight parameters, which are updated upon observation of a new value on that pixel. As explained above, if $\boldsymbol{x}_t$ falls into the confidence region of a Gaussian in a mixture, $\boldsymbol{x}_t$ is said to "match" with that Gaussian. Then a question may arise; how can we specifically confirm that $\boldsymbol{x}_t$ matches with a Gaussian? In ISG model, if $\boldsymbol{x}_t$ is within 2.5 standard deviation interval of a component, it is said that there is a matching condition. In other words, if $\boldsymbol{x}_t$ is within %99 confidence interval of one of the mixture components, it matches with that component.

A color vector might as well match with more than one component in the mixture. There should always be only one matching component for any pixel value, since our segmentation decision will be based on this match. For this purpose, we analyze the matching condition for $\boldsymbol{x}_t$, beginning from the highest weighted component in the mixture. Thus, once we detect a match condition between a vector and a mixture component, then we stop looking through other components.

Referring to equation 2.3, the mathematical way to evaluate a match between $\boldsymbol{x}_t$ and n^{th} Gaussian component of the mixture is explained with the equations below. If the condition given below is true, it means $\boldsymbol{x}_t$ is matching with the n^{th} Gaussian in the mixture of Gaussians.

$$\exp\left[-\frac{1}{2}(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})^T(\frac{1}{\sigma^2_{n,t}})(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})\right] \geq \exp\left[-\frac{1}{2\sigma^2_{n,t}}(2.5\boldsymbol{\sigma}_{n,t})(2.5\boldsymbol{\sigma}_{n,t})\right], \quad (2.4)$$

$$\left[(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})^T(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})\right] \leq \left[(2.5\boldsymbol{\sigma}_{n,t})(2.5\boldsymbol{\sigma}_{n,t})\right], \quad (2.5)$$

$$\left[(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})^T(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})\right] \leq \left[(2.5)^2 3\sigma^2_{n,t}\right]. \quad (2.6)$$

Notice that $\boldsymbol{\sigma}_{n,t}$ is actually a 3 dimensional vector above, and each element in this vector equals to $\sigma_{n,t}$, which is a scalar. σ_t is the variance for a single channel only, and same for all channels according to our simplification. Otherwise, assuming that variance values are different for R,G,B channels, matching rule should be stated as below:

$$\left[(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})^T(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})\right] \leq \left[(2.5)^2(\sigma^2_R+\sigma^2_G+\sigma^2_B)\right], \quad (2.7)$$

where individual variances for each channel are different.

Once the match condition is evaluated, our model records the matching Gaussian index for $\boldsymbol{x}_t$. Then, the model is updated with the new information. There is a real-time update mechanism for the mean, variance and the weight of Gaussians for each pixel mixture. In terms of the mean, basically, received pixel value shifts the distribution mean towards itself. Variance of the Gaussian also changes in addition to the shift in the mean value. Also, the component which matches with the value observed should become more dominant within the mixture. For this reason, "only for the mixture component that matches with the observed value $\boldsymbol{x}_t$", mean and variance values will be updated as follows:

$$\boldsymbol{\mu}_{n,t} = (1-\rho)\boldsymbol{\mu}_{n,t-1} + \rho\boldsymbol{x}_t, \quad (2.8)$$

$$\sigma^2_{n,t} = (1-\rho)\sigma^2_{n,t-1} + \frac{\rho}{d}(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t})^T(\boldsymbol{x}_t-\boldsymbol{\mu}_{n,t}), \quad (2.9)$$

where ρ is the parameter defining the learning rate for the Gaussian distribution. For computational simplicity, this parameter is fixed according to the observed scene characteristics. Notice that the matching component gets narrower or wider, meaning that the variance changes with the observation. One question that may arise here is: What if an observed value $\boldsymbol{x}_t$ does not match with any of the K Gaussian components in the mixture? How are we going to update the model when there is no match? In that case, inside the mixture, the component with the minimum weight is discarded. In place of the discarded component, a new component is created, with mean value equal to $\boldsymbol{x}_t$, and variance value equal to an initial variance parameter σ_{init}. As expected, in the first iteration of this method for the first video frame, since there are no Gaussian components that would be created before, there will not be any match conditions. In other words, during analysis of the first frame in a video, the first Gaussian components for each pixel mixture is created.

Although mean and variance are updated for only the matching component, weights are updated for all components in the mixture. Weight update for the n^{th} component in the mixture is realized as follows:

$$w_{n,t} = (1 - \alpha)w_{n,t-1} + \alpha M_n, \tag{2.10}$$

where M_n is 1 if the n^{th} component is the matching one with $\boldsymbol{x}_t$, and it is 0 for all other components in the mixture. This actually corresponds to increasing the weight of the matching component and renormalizing all the weights in the mixture to 1 again. α is the weight learning parameter and selected experimentally. To be able to make this a separate entity than the learning rate ρ which is used for the mean and variance update, another variable name is used in the implementation; anyway α equals ρ in our tests.

Up to this point, how each Gaussian mixture per pixel keeps track of the observations on that pixel has been shown. Now, the decision to estimate the labeled image $\boldsymbol{Y}$ out of the observation $\boldsymbol{X}$ will be explained.

The mixture for pixel s represents the overall distribution of color values on that pixel. Actually, it is not only the model of the background colors or the model of the foreground colors. It is the general color model for a pixel. In this model, our assumption was that the highest weighted components should represent the background, since our observations on that pixel are dominated by the background. Therefore, we can classify s into classes background($y_s = 0$) or foreground($y_s = 1$). Weight of the matching Gaussian with $\boldsymbol{x}_t$ is the important value here. Simply put, if the matching component is one of those highest weighted components in the mixture, then $\boldsymbol{x}_t$ is classified as background, thus $y_s = 0$. Otherwise, this pixel is foreground at the moment, and $y_s = 1$. But, how does ISG model identify those high weighted components inside the mixture? What is the "high" threshold for the weights as we have asked before in this chapter?

Below is the algorithm to separate high weighted background components out of a mixture, and then to classify a pixel as background or foreground:

Algorithm 1 ISG Decision Process

```
Ensure: $w_1 > .. > w_K$ for $K$ Gaussians
  Sum of Weights = 0.0 and
  Set Threshold $\tau_w \epsilon [0, 1]$
  for $k = 1$ to $K$ do
    Add $w_k$ to the Sum of Weights
    if Sum of Weights $> \tau_w$ then
      Break the Loop
    end if
  end for
  $B = k$, $m$ = Matching Gaussian Index for $\boldsymbol{x}_t$
  if $m < B$ then
    $y_s = 0$ (Background)
  else
    $y_s = 1$ (Foreground)
  end if
```

In ISG model, it is assumed that on a pixel, mostly background will be observed. In the long run, highest weighted components will be formed by the values observed in background objects. To demonstrate, visualize an outdoor scene. One of the pixels' mixture has two highly weighted Gaussian components and three low weight components, thus $B = 2$ and $K = 5$. Intuitively, those two components should represent the background. One of the mean values is probably around brighter color values learned during daytime. The other mean might be around darker color values for nighttime color observations. Three remaining non-background components can have their means around any color value, since ISG does not try to define foreground model. In this manner, multiple background color distributions due to changes in lighting and also periodic motion in the observed scene such as the leaves of a tree, can be captured. To rephrase the decision criteria, in the weight-wise ordered mixture, the first B components represent the background such that:

$$B = \arg\min_{b} \left(\sum_{n=1}^{b} w_n \geq \tau_w \right), \tag{2.11}$$

where τ_w is an experimental threshold value for the sum of weights.

As a result of this overall labeling for each pixel s, label image $\boldsymbol{Y}$ is obtained. In the results chapter, it will be visually shown that the ISG model detects the foreground objects in a fragmented format. Due to the uncorrelation assumption between pixels, instead of a single foreground object, the ISG detects multiple smaller objects that are positioned closely. To overcome this problem, we tried to combine these small independent object fragments using a thresholding approach based on foreground probabilities for the pixels. Before that, in the next section, how we obtain foreground class probabilities for the pixels using the ISG model will be shown. In the next chapter, the thresholding solution to the fragmentation problem which utilizes these probabilities is explained.

2.2 Inheriting Class Probabilities from ISG

We will now explain how we extract class probabilities using the ISG model. All explanations in this section will now be made for a specific time t, therefore time subscript is removed from the symbols used.

As explained in the previous section, for observation image $\boldsymbol{X}$ taken as input to the ISG model, a label image $\boldsymbol{Y}$ is the output. ISG assumes that all pixels are independent. Thus, probability of observing a label image $\boldsymbol{Y}$ from $\boldsymbol{X}$, which is $P(\boldsymbol{Y}|\boldsymbol{X})$ is not explicitly calculated and used in the ISG model for segmentation. The only value used is the weight of a matching Gaussian component as explained in Algorithm 1.

We modified the model and added an intermediate step of calculating $P(\boldsymbol{Y}|\boldsymbol{X})$. We can write this probability as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|\boldsymbol{Y})P(\boldsymbol{Y})}{P(\boldsymbol{X})}. \tag{2.12}$$

For accepting a possible label image $\boldsymbol{Y}_1$, which has a specific configuration of y_s values for all s, we simply can compare $P(\boldsymbol{Y}_1|\boldsymbol{X})$ to other $P(\boldsymbol{Y}|\boldsymbol{X})$ values for other label images. In this comparison, the denominator $P(\boldsymbol{X})$ in equation 2.12 will always be the same. $P(\boldsymbol{X})$ is the prior probability of observing image $\boldsymbol{X}$ in the formula, and this can be considered to be the normalizing constant for the probability function given above. $P(\boldsymbol{X})$ is simply the sum of $P(\boldsymbol{X}|\boldsymbol{Y})P(\boldsymbol{Y})$ for all possible $\boldsymbol{Y}$. Thus, from equation 2.12, the numerator $P(\boldsymbol{X}|\boldsymbol{Y})P(\boldsymbol{Y})$ is the significant term. Since all pixels are assumed to be independent in ISG, $P(\boldsymbol{Y}|\boldsymbol{X})$ and $P(\boldsymbol{Y})$ can be written as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{s \in \boldsymbol{S}} P(y_s|\boldsymbol{x}_s), \tag{2.13}$$

$$P(\boldsymbol{Y}) = \prod_{s \in \boldsymbol{S}} p(y_s). \tag{2.14}$$

In equations 2.13 and 2.14, y_s is the specific label value for pixel s and $\boldsymbol{x}_s$ is the color vector observed on s. Due to the independence assumptions in the ISG model, we can actually find conditional class probabilities on pixel s given $\boldsymbol{x}_s$ as follows:

$$P(y_s|\boldsymbol{x}_s) = \frac{P(\boldsymbol{x}_s|y_s)P(y_s)}{P(\boldsymbol{x}_s)}. \tag{2.15}$$

Since we only consider 2 classes, prior probabilities $P(y_s)$ can be trivially figured out as $P(y_s = 0) = \lambda$ and $P(y_s = 1) = 1 - \lambda$. λ can be altered according to the scene characteristics. A general estimation for λ can be made by observing the scene for some training time. The remaining $P(\boldsymbol{x}_s|y_s)$ term in equation 2.15 is inherited from the mixture model. This is the likelihood of color vector $\boldsymbol{x}_s$ when the class of the pixel is assumed to be y_s. Remembering that in the mixture for pixel s, there can be at most K Gaussian components and the first B components with highest weights represent background colors; we can separate the mixture into two different sub-mixtures. First of these sub-mixtures contains the highest weighted B components from the main mixture and this is the background class sub-mixture, representing the probability distribution of the background color values. The second mixture contains the remaining $K-B$ Gaussian components, and this mixture represents the foreground probability distribution. Weights of the Gaussian components for each sub-mixture should be normalized so that the sum of the weights becomes 1 for each sub-mixture. In formulas, we can state that:

$$P(\boldsymbol{x}_s|y_s=0)=\sum_{k=1}^{B}\frac{w_k}{\sum\limits_{j=1}^{B}w_j}\mathcal{N}(\boldsymbol{x}_s,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k), \tag{2.16}$$

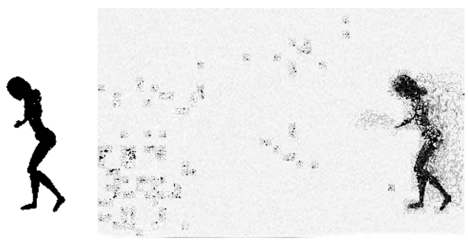
$$P(\boldsymbol{x}_s|y_s=1)=\sum_{k=B+1}^{K}\frac{w_k}{\sum\limits_{j=B+1}^{K}w_j}\mathcal{N}(\boldsymbol{x}_s,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k), \tag{2.17}$$

$$P(\boldsymbol{x}_s)=\sum_{y_s=\{0,1\}}P(\boldsymbol{x}_s|y_s)P(y_s), \tag{2.18}$$

Then we form a probability image $\boldsymbol{V_I}$ using equation 2.12. Each element of $\boldsymbol{V_I}$ on pixel s is v_s. v_s is the foreground probability for s, therefore $v_s = P(y_s = 1|\boldsymbol{x}_s)$. Figure 2.2 shows a visualization of a sample probability image on which, darker colors represent higher foreground probabilities. This is the intermediate step that enables different type of enhancements on the ISG model. In regular ISG model, there is a direct binary classification for each pixel, using the Gaussian components' weights in the mixture of that pixel. In our probabilistic ISG approach, we obtain a matrix containing the conditional foreground probabilities for each pixel and classification is based on the explicit foreground probabilities for the pixels. A thresholding operation on $\boldsymbol{V_I}$ is performed to obtain a new labeling image, $\boldsymbol{Y} = (\boldsymbol{V} > \tau_p)$ where τ_p is an experimental probability threshold.

This approach is also utilized in the next chapter in hysteresis thresholding context, and is explained in detail in that chapter.

Experimental results for the ISG based foreground segmentation are provided in Chapter 6 together with the results of all other methods which will be explained further in the thesis.



(a) Foreground - Groundtruth (b) Foreground Probabilities

Figure 2.2: An ISG based Probability Image

In the next section, we explain the new thresholding method which addresses the spatial correlation issue and the fragmentation problem in ISG based foreground segmentation.

Chapter 3

Improved Post-Processing

In the previous chapters, we used the label probability values obtained from the ISG model. In this chapter, we mostly focus on the binary output $\boldsymbol{Y}$ of the ISG model. We introduce post processing methods that can directly be applied on $\boldsymbol{Y}$. However, we also introduce a thresholding technique in this chapter, which still operates on the probability image $\boldsymbol{V}_I$.

In ISG method, all pixels have their own mixtures of Gaussians, and pixels are independent. This brings simplicity in terms of CPU cycles and processing speed, but in real life, there is usually a correlation between the pixels in an image. This spatial correlation is not taken into account in the ISG model. For this reason, single foreground objects might sometimes be detected as many number of smaller fragments. We address this fragmentation problem, which causes a significant drop in recall rate of the segmentation, in this chapter.

The first section describes additional post-processing on $\boldsymbol{Y}$ to achieve better segmentation performance. The next section explains a new secondary probability thresholding method to detect missing foreground pixels. We call this secondary thresholding as "hysteresis thresholding", or "relaxed thresholding in a hysteresis region". Final section of the chapter will show how we can also incorporate edge and gradient information of the foreground to precisely smooth the segmentation results, and to slightly improve the final performance of the operation.

3.1 Standard Post-Processing

After the foreground image $\boldsymbol{Y}$ is retrieved, it is subject to some post processing operations:

3.1.1 Opening and Closing

In the foreground, there are usually many small holes in the detected objects and there are also some small objects detected which are not really foreground, but occur due to the noise and the dynamic background. Since opening smooths the contour of an object by eliminating protrusions and closing smooths the contours by filling the gaps and holes on the contours [30], an opening and a closing operation are performed as standard post-processing morphological operations.

3.1.2 Connected Component Analysis

A connected component analysis is performed on $\boldsymbol{Y}$ after the morphological operations. The connected regions are obtained and the contours $\{C_1, C_2 \dots C_K\}$ of these regions on the foreground image are found. In this work, we only take the external contours into consideration, therefore it is assumed that the foreground objects will not have a hollow structure.

3.1.3 Minimum Area Filtering

Let A_{min} be the minimum area value determined experimentally. For an object with external contour C having a pixel wise area of A_C, the object will be filtered out of $\boldsymbol{Y}$ if $A_C < A_{min}$. Thus, very small pieces that were not eliminated through the morphological operations and that are too small to be foreground are left out.

After the standard post-processing, the set of contours surrounding the foreground regions is formed and an updated foreground image $\hat{\boldsymbol{Y}}$ is formed by union of interiors of these contours.

These post-processing methods provide enhancements, but these are not able to eliminate the fragmentation problem. To address this problem in a more strict manner,

(a) Fragmented Foreground (b) Real Foreground

Figure 3.1: ISG Fragmentation Problem

we introduce a secondary probability-based thresholding in a specific hysteresis region. After this, we also analyze foreground objects edge information to fine-tune the segmentation. These additional techniques will be explained in detail in the following section.

3.2 Hysteresis Thresholding

In ISG model, decision to classify a pixel into foreground or background was given in Chapter 2 in Algorithm 1. ISG model itself does not calculate $P(\boldsymbol{Y}|\boldsymbol{X})$ values. The only criterion used in ISG for decision making to label pixel s, is the weight of the Gaussian component that matches with the observation $\boldsymbol{x}_s$ on pixel s. In Chapter 2, we showed how we can also obtain $P(\boldsymbol{Y}|\boldsymbol{X})$ values using the ISG. $\boldsymbol{V}_I$ is the probability image that contains $P(y_s|\boldsymbol{x}_s)$ values for all s.

Relying only on the weight of the matching component might be problematic, when all pixels are also considered to be independent in the ISG model. Instead of a single foreground object, ISG tends to detect many smaller fragments of the object which are very close to each other. Due to the spatial independence, this proximity is not analyzed at all. Therefore, ISG cannot combine all these fragments by filling in the space between the fragments appropriately. To compensate for this weak point, we find a region that contains all the fragments of a possible single piece foreground object. As it is seen in Figure 3.1, when object colors are similar to those of the background for some pixels, a very apparent fragmentation occurs. This also reduces the detection performance significantly.

To deal with this problem, we define a search region, called the hysteresis region. Then in the hysteresis region, we search for pixels which might belong to the foreground, but that are classified as background due to color similarities with the background. In this search, we directly utilize the pixel probabilities obtained from the probability image $\boldsymbol{V}_I$. Simply put; for pixel s detected as background inside the hysteresis region (suspicious region), if foreground probability $P(y_s = 1|\boldsymbol{x}_s)$ is higher than a relaxed threshold value τ_r, then we say that this pixel is actually foreground, and we reclassify it such that $y_s = 1$. The first step in this relaxed thresholding is to find the hysteresis region.

3.2.1 Hysteresis Search Region

Let $d_{min}(C_n, C_m)$ be the minimum distance between two object contours C_n and C_m on $\hat{\boldsymbol{Y}}$. We iterate through all contour pairs and if the distance between two contours is less than the threshold D_{max}, we find the union of interiors of the convex hulls for those two contours, $\boldsymbol{H}_{nm}$. Then the union of such convex hull regions is taken to form a mask where the relaxed threshold will be applied. This union image operates as a mask combining pairs of regions that have a high probability for belonging to the same single region foreground object. Assuming there are N_R objects and their contours, the process below is performed:

Algorithm 2 Finding Hysteresis Region

```
Ensure: $\boldsymbol{H}(i,j) = 0$ for all $i, j$
  for $n = 1$ to $N_R$ do
    for $m = n + 1$ to $N_R$ do
      $D = distance(C_n, C_m)$
      if $D < D_{max}$ then
        $\boldsymbol{H}_{nm} = convexhull(C_n \cup C_m)$
        $\boldsymbol{H} = \boldsymbol{H} \vee \boldsymbol{H}_{nm}$
      end if
    end for
  end for
```

At the end of this process, the hysteresis search region indicator image $\boldsymbol{H}$ is obtained. It can be considered as a foreground image with the union of convex hulls of close contours in $\hat{\boldsymbol{Y}}$. In Figure 3.2, the union of object convex hulls can clearly be seen. In the same figure on the second image, gray areas represent hysteresis region. This region is the group of pixels classified as background, but these pixels lie within the proximity of some small foreground segments which are located close to each other.

(a) Union of convex hulls (b) Gray Hysteresis Region

FIGURE 3.2: Hysteresis Region for the Foreground

3.2.2 Relaxed Thresholding

For the pixels in $\boldsymbol{H}$, a secondary thresholding with a new threshold value τ_r which is lower than the ISG threshold τ_p is applied. In Stauffer-Grimson method, the sum of the weights of the Gaussian mixture components are compared to the single threshold value τ_w as it was explained in Chapter 2 Algorithm 1. Primary foreground image $\boldsymbol{Y}$ is formed with this thresholding. In hysteresis thresholding, we will update the new foreground image $\hat{\boldsymbol{Y}}$. The label for pixel s on this new image is $\hat{y}(s)$.

ISG thresholding mechanism causes the highest weighted Gaussian component to be considered as background regardless of its weight. This may cause problems when there is a highly variable background and the current foreground pixel value is covered by one of the background Gaussian components (which becomes more likely due to the variable background). Because of this, in relaxed thresholding, only the foreground probability of pixel s is compared to the new threshold, τ_r. Considering that $v_s = P(y_s = 1|\boldsymbol{x}_s)$ for pixel s, Algorithm 3 summarizes the procedure.

Algorithm 3 Relaxed Thresholding

```
Ensure: $\hat{\boldsymbol{Y}} = \boldsymbol{Y}$
  for $s = (i, j), s \in \boldsymbol{H}$ do
    if $y_s = 0$ then
      if $v_s >= \tau_r$ then
        $\hat{y}(s) = 1$
      end if
    end if
  end for
```

As it is seen in Algorithm 3, only pixels having a high foreground probability within the hysteresis region are modified in $\hat{\boldsymbol{Y}}$.

Dilation and erosion is then applied to $\hat{\boldsymbol{Y}}$ to obtain a better labeling. In the next step, using foreground edges as an enhancement is shown.

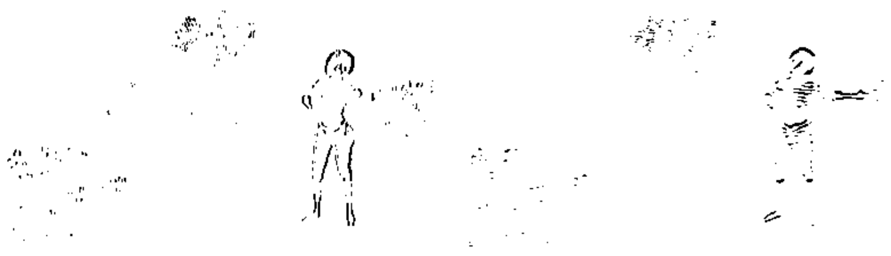
3.2.3 Foreground Edge Extraction

The gradient operator uses neighbors of a pixel to determine spatial derivatives of the intensity image. Gradient information in the background versus the foreground should be complementary to the color change information used in ISG method. Thus, in addition to relaxed thresholding, we also employ a foreground edge detection algorithm to determine foreground edge pixels inside the hysteresis search region. Foreground edges will mark only the edges of foreground objects. We consider only external contours to cover foreground objects, thus no hollow foreground objects are allowed. Because of this assumption, correctly detected foreground edge points will possibly aid in determining the whole foreground object as a single object. Foreground edge detection is realized by the Algorithm 4:

Algorithm 4 Foreground Detection

```
Apply Gaussian Smoothing on $\boldsymbol{X}$
$\boldsymbol{G}_x = \frac{\delta \boldsymbol{X}}{\delta x}$
$\boldsymbol{G}_y = \frac{\delta \boldsymbol{X}}{\delta y}$
$\boldsymbol{\Lambda}_{x,t}$ = History of horizontal gradient
$\boldsymbol{\Lambda}_{y,t}$ = History of vertical gradient
$\boldsymbol{\Lambda}_{x,t} = (1-\kappa_e)\boldsymbol{\Lambda}_{x,(t-1)} + \kappa_e G_x$
$\boldsymbol{\Lambda}_{y,t} = (1-\kappa_e)\boldsymbol{\Lambda}_{y,(t-1)} + \kappa_e G_y$
$\boldsymbol{D}_{edge} = \sqrt{(\boldsymbol{G}_x - \boldsymbol{\Lambda}_{x,t})^2 + (\boldsymbol{G}_y - \boldsymbol{\Lambda}_{y,t})^2}$
for all $s = (i,j) \in \boldsymbol{H}$ do
  if $D_{edge}(s) < \tau_e$ then
    $\boldsymbol{E}_{fg}(s) = 0$
  else
    $\boldsymbol{E}_{fg}(s) = 1$
  end if
end for
```

In this procedure, κ_e is the edge learning parameter and τ_e is the edge threshold parameter. $\boldsymbol{E}_{fg}$ is the labeling image resulting from the edge comparison only. A Sobel operator with a 3×3 kernel is used. For the x-derivative and for the y-derivative, kernels used are:



(a) Foreground Edge X Components (b) Foreground Edge Y Components

FIGURE 3.3: Edges of Foreground

$$\left\{\begin{array}{ccc} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{array}\right\} \text{ and } \left\{\begin{array}{ccc} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{array}\right\} \text{ respectively.}$$

Figure 3.3 shows the x-direction and y-direction components of the E_{fg} image for a sample frame, calculated via the algorithm above by differentiating the overall gradients.

We have also experimented with the Laplacian operator to find the gradient images; but our experiments show that Laplacian operator results in a slightly worse foreground change detection than the procedure above.

The foreground edge information is then used as follows. A pixel inside the hysteresis search region $\boldsymbol{H}$ is considered as a foreground pixel if it passes the relaxed threshold test or it is found as a foreground edge using the test above. In other words, we form a new foreground mask image by the following operation $\hat{\boldsymbol{Y}} \vee \boldsymbol{E}_{fg}$.

The main purpose in this method is to find a way to detect the missed foreground pixels which lie in th vicinity of ISG-detected foreground pixels. Since the ISG model considers each pixel independently, in the output image for the ISG model there might be some small clusters of pixels detected as background, lying between larger clusters of foreground pixels. This method simply looks for those small cluster of pixels. Obviously, any pixel that lies in between closely located foreground clusters should not directly be classified as foreground. At this point, this is the reason why we apply the secondary thresholding in a relaxed manner, with a lower threshold value. If a pixel is between close foreground clusters and it also qualifies to be a foreground according to its foreground probability v_s, then it should be a foreground pixel which has been missed by the ISG model due to the similarity between its color and the background color.

This method brings improvements in terms of segmentation recall rate -the ratio that defines what percentage of the real foreground has actually been detected- because of the mentioned reasons. Performance results on different test videos are shown in Chapter 6, in comparison with other methods in this thesis.

The next chapter explains how we leverage the ISG probability image $\boldsymbol{V}_I$ in MRF based segmentation. Just like in hysteresis thresholding approach, we also utilize MRF models to compensate for the spatial correlation between pixels. It can be said that MRF segmentation will accept $\boldsymbol{V}_I$ as an input and the data model, and give out a different label image $\boldsymbol{Y}$.

Chapter 4

Markov Random Fields Based Segmentation

Markov Random Fields are used to model spatial dependencies between random variables. MRF models represent the joint probability distribution of a group of random variables by using a set of local dependencies. Conditional dependencies between neighbor pixels is leveraged. MRF models are widely used in image smoothing, segmentation and restoration. In this chapter, we will describe how MRFs can be utilized using the probability image $\boldsymbol{V}_I$ we obtain from the ISG model. We explain two practical MRF energy optimization / probability maximization techniques: Iterated Conditional Modes (ICM) and Belief Propagation (BP). We also present experimental results for MRF segmentation using BP optimization in the results chapter, Chapter 6.

4.1 Markov Random Field Models

Markov Random Fields based segmentation will be used to obtain a familiar output, $\boldsymbol{Y}$, which is the label image. Let the image width be W and the image height be H in number of pixels. Remembering that s is the coordinate for a pixel, we define lattice $\boldsymbol{S}$ as: $\boldsymbol{S} = \{s = (i, j) | 1 \leq i \leq H, 1 \leq j \leq W\}$ is a lattice. In this case, the lattice includes the whole image. We have an observation image $\boldsymbol{X}$, where on every pixel s there is a color value $\boldsymbol{x}_s$. Pixel s will have a label value y_s on image $\boldsymbol{Y}$ which can be either 1 or 0.

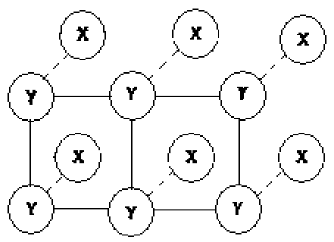


Figure 4.1: Graphical MRF Model

$\boldsymbol{Y}$ is the set of y_s variables and it is a Markov random field. $\boldsymbol{y}$ is a specific configuration of this set of random variables.

We will utilize MRF assumptions for getting the most probable $\boldsymbol{Y}$ given $\boldsymbol{X}$, hence the purpose is to model $P(\boldsymbol{Y}|\boldsymbol{X})$ and select $\boldsymbol{Y}$ that provides the highest probability. If the components of $P(\boldsymbol{Y}|\boldsymbol{X})$ are analyzed as in the previous chapters using the Bayesian rules, it is certain that $\boldsymbol{Y}$ configuration is dependent on $\boldsymbol{X}$. In addition, y_s variables in $\boldsymbol{Y}$ have their joint probability $P(\boldsymbol{Y})$. Observations $\boldsymbol{x}_s$ on different s pixels are independent of each other. $P(\boldsymbol{X}|\boldsymbol{Y})$ component of $P(\boldsymbol{Y}|\boldsymbol{X})$ can then be reduced to $\prod\limits_{s\in\boldsymbol{S}} P(\boldsymbol{x}_s|y_s)$. This concept is illustrated in a graphical MRF model shown in Figure 4.1.

On the other hand, $P(\boldsymbol{Y})$ depends on the joint distribution of y_s variables. Since we assume that y_s variables satisfy Markovian characteristics $P(\boldsymbol{Y})$ can be reduced to a set of conditional distributions between locally neighboring y_s. Before showing how this value can be obtained, we should define terms used in MRFs.

The neighborhood of pixel s is called as $\boldsymbol{N}_s$. For a neighbor pixel r where $r \in \boldsymbol{N}_s$, it is also valid that $s \in \boldsymbol{N}_r$ where $\boldsymbol{N}_r$ defines the neighborhood of r. That is to say, neighborhood in this context is a mutual rule. A pixel is not considered to be its own neighbor, so $s \notin \boldsymbol{N}_s$.

We first define pixel neighborhoods. A neighborhood for a pixel can be of some degree n where $n \geq 0$. For the n^{th} degree neighborhood of pixel s , which we can call $\boldsymbol{N}_s^n$, it can be stated that $\boldsymbol{N}_s^n = \{r : ||r - s||_2 \leq n, r \neq s\}$.

Another term should be explained. We define a set of pixels which is called a clique, $\boldsymbol{c}$. In a clique, every possible pair of pixels are mutually neighbors. So for a 4-neighborhood region, maximum size for a clique can be 2 pixels, otherwise all pairwise pixels inside $\boldsymbol{c}$ will not be mutual neighbors. Also, the set of all cliques in the image is $\boldsymbol{C}$. Cliques are dependent on the definition of the neighborhood relation.

We assumed that $\boldsymbol{Y}$ is a Markov random field where the $P(\boldsymbol{Y})$ can be reduced to a set of local conditional probabilities. For $\boldsymbol{Y}$ to be Markov random field, following properties must be satisfied:

- **Positivity property:** $P(\boldsymbol{Y}) > 0$ for any $\boldsymbol{Y}$, meaning that any labeling should be possible.
- **Markovianity property:** $P(y_s|y_r, r \neq s) = P(y_s|y_r, r \in \boldsymbol{N}_s)$. This means that the probability of y_s is only dependent on the other variables y_r within the neighborhood of s. The joint distribution for $P(\boldsymbol{Y})$ can be represented by the local dependencies between the neighbors. In this approach, a pixel is conditionally independent from all other non-neighboring pixels.

As we stated in the beginning, the most probable $\boldsymbol{Y}$ given $\boldsymbol{X}$ is the optimal labeling. This $\boldsymbol{Y}$ value should be selected, and classification of pixels should be defined according to this for a correct maximum probability based segmentation.

The Hammersley-Clifford theorem states that the joint probability distribution of any Markov random field $\boldsymbol{Y}$ can be written as a Gibbs distribution [17]. Thus, $\boldsymbol{Y}$ is both a Markov random field and a Gibbs random field. Gibbs distribution for random field $\boldsymbol{Y}$ can be written as:

$$P(\boldsymbol{Y}) = \frac{1}{Z}\exp(-U_M(\boldsymbol{Y})). \tag{4.1}$$

Z is used to normalize the probability distribution and $U_M(\boldsymbol{Y})$ is the MRF energy. To demonstrate this in a specific way: consider that $\boldsymbol{y}$ is a single configuration instance for $\boldsymbol{Y}$. Then, the probability for the specific configuration $\boldsymbol{y}$ can be written as:

$$P(\boldsymbol{y}) = \frac{1}{Z}\exp(-U_M(\boldsymbol{y})). \tag{4.2}$$

In equation 4.2, $P(\boldsymbol{y})$ should be normalized over all $\exp(-U_M(\boldsymbol{y}))$ values for any possible $\boldsymbol{y}$. So the normalization constant Z can then be written as:

$$Z = \sum_{\boldsymbol{y}'} \exp(-U_M(\boldsymbol{y}')). \tag{4.3}$$

We stated that $U_M(\boldsymbol{y})$ is the MRF energy. This energy is related to the joint probability of observing $\boldsymbol{y}$. Referring back to the purpose of MRFs, we should maximize $P(\boldsymbol{Y}|\boldsymbol{X})$. In Chapter 2, it was already stated in equation 2.12 that $P(\boldsymbol{Y}|\boldsymbol{X})$ can be written as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|\boldsymbol{Y})P(\boldsymbol{Y})}{P(\boldsymbol{X})}. \tag{4.4}$$

$U_M(\boldsymbol{y})$ in equation 4.3 arises from the probability $P(\boldsymbol{Y})$ in equation 4.4. Since this is the probability representing the consistency between all the labels in image $\boldsymbol{Y}$, we usually refer to this probability as the smoothness probability. Hence, $U_M(\boldsymbol{y})$ is the smoothness energy for specific a configuration of labels in $\boldsymbol{Y}$, which is $\boldsymbol{y}$.

$U_M(\boldsymbol{Y})$ in general depends on the joint probability $P(\boldsymbol{Y})$, which can be represented with local conditional dependencies between pixels. We can write $P(\boldsymbol{Y})$ as:

$$P(\boldsymbol{Y}) = \frac{1}{Z} \prod_{\boldsymbol{c} \in \boldsymbol{C}} \Psi_c(\boldsymbol{y}_c), \tag{4.5}$$

where $\boldsymbol{y}_c$ represents a clique. In our implementation, we used 4-neighborhood relations between pixels. We also used only the maximal cliques with size 2. Maximal cliques are the cliques in an image which cannot be extended by adding any more pixels in size. Thus, in this case $\boldsymbol{y}_c = [y_s, y_r]$ is the maximal clique. $\boldsymbol{y}_c$ is a group of two labels of the pixels in a clique and $\boldsymbol{C}$ is the set of all maximal cliques in the image. Ψ_c is a potential function of a clique that is used to form a valid probability from the labels of the pixels in the clique. If we try to obtain the smoothness energy $U_M(\boldsymbol{Y})$ arising from the probability $P(\boldsymbol{Y})$ in 4.5, then the $U_M(\boldsymbol{Y})$ becomes:

$$U_M(\boldsymbol{Y}) = \sum_{\boldsymbol{c} \in \boldsymbol{C}} V_c(\boldsymbol{y}_c). \tag{4.6}$$

Here, $V_c(\boldsymbol{y}_c)$ is called the potential function for a clique which is written in terms of the energy instead of probability. In equation 4.5, Ψ_c is the potential that is written in terms of probabilities (it is not a probability itself). Therefore, $V_c(\boldsymbol{y}_c)$ can be written as $-\log \Psi_c(\boldsymbol{y}_c)$. $U_M(\boldsymbol{Y})$ is found over the whole image using every clique $\boldsymbol{c}$ in clique set $\boldsymbol{C}$. Clique potential function $V_c(\boldsymbol{y}_c)$ can be defined using different functions, depending on the application.

For our assumption for the clique potential, we use the prototypical Markov random field model, which is the Ising model, since MRFs were introduced for the Ising model [15] and it is trivial to implement the model. In this model, if the labels of the neighbor pixels are different, this gives rise to a smoothness penalty. If we say that the smoothness energy around a single pixel s is defined as u_M, and if we only consider 1_{st} degree neighbors r of pixel s, inside its neighborhood N_s^1; then it is stated that:

$$u_M(s) = \left[\theta \sum_{\forall r} \delta(y_s, y_r)\right]; \tag{4.7}$$

and the overall MRF energy for the image containing all pixels s can be calculated as:

$$U_M(\boldsymbol{Y}) = \sum_{s \in S} u_M(s), \tag{4.8}$$

where $\delta(y_s, y_r) = -1$ if $y_s = y_r$ and $\delta(y_s, y_t) = 1$ if $y_s \neq y_r$. θ is used to adjust the energy scales for a reliable smoothness penalty on the image. Note that equation 4.8 is very similar to equation 4.6, where $V_c(\boldsymbol{y}_c) = \frac{\theta}{2}\delta(y_s, y_r)$.

According to this smoothness constraints, whenever the number of neighbors having the same label value with the pixel increases, the smoothness energy decreases. Regarding this decrease in the energy, the probability of this labeling will increase. The assumption that neighboring pixels should have a tendency to have the same labels can be clearly seen. In addition, using the Markovian property of $\boldsymbol{Y}$, overall sum of clique potentials is expressed in terms of pixel neighborhoods.

Now the energy component $U_M(\boldsymbol{Y})$ and the related probability $P(\boldsymbol{Y})$ has been defined. Referring back to the probability $P(\boldsymbol{Y}|\boldsymbol{X})$ in equation 4.4 which we try to maximize, $P(\boldsymbol{X}|\boldsymbol{Y})$ component in the numerator and $P(\boldsymbol{X})$ component in the denominator should be analyzed afterwards. $P(\boldsymbol{X}|\boldsymbol{Y})$ is the probability measuring how well the labeling $\boldsymbol{Y}$ satisfies the observations $\boldsymbol{X}$. This probability is called as the data-fitting probability. As it is demonstrated in figure 4.1, for each pixel s, the data-fitting probability $P(\boldsymbol{x}_s|y_s)$ is independent. Therefore, $P(\boldsymbol{X}|\boldsymbol{Y})$ can be expressed as:

$$P(\boldsymbol{X}|\boldsymbol{Y}) = \prod_{s \in \boldsymbol{S}} P(\boldsymbol{x}_s|y_s). \tag{4.9}$$

$P(\boldsymbol{x}_s|y_s)$ can be retrieved from the ISG background model. Referring back to equations 2.16 and 2.17 in Chapter 2, for any pixel value $\boldsymbol{x}_s$, likelihood $P(\boldsymbol{x}_s|y_s = 0)$ or $P(\boldsymbol{x}_s|y_s = 1)$ can be easily found from the ISG model with the given formulas.

Similar to the fact that $P(\boldsymbol{Y})$ defines the MRF energy $U_M(\boldsymbol{Y})$; $P(\boldsymbol{X}|\boldsymbol{Y})$ defines the data fitting energy $U_D(\boldsymbol{X}|\boldsymbol{Y})$. Considering that the relation between the observations $\boldsymbol{x}_s$ and labels y_s is independent for any s, and using the probability relation in equation 4.9, we can assert that for pixel s, the data energy can be written as the negative logarithm of the likelihood value calculated from the ISG model as shown below:

$$u_D(\boldsymbol{x}_s|y_s) = -\log(P(\boldsymbol{x}_s|y_s)). \tag{4.10}$$

And then the overall data energy for the whole image becomes:

$$U_D(\boldsymbol{X}|\boldsymbol{Y}) = \sum_{s \in \boldsymbol{S}} u_D(\boldsymbol{x}_s|y_s). \tag{4.11}$$

We have now defined probabilities $P(\boldsymbol{X}|\boldsymbol{Y})$ and $P(\boldsymbol{Y})$ in equation 4.4, and also obtained the related energies, $U_D(\boldsymbol{X}|\boldsymbol{Y})$ and $U_M(\boldsymbol{Y})$.

Now the classification decision to find the optimum labeling $\hat{\boldsymbol{Y}}$ may be expressed as:

$$\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y}} \left[\frac{P(\boldsymbol{X}|\boldsymbol{Y})P(\boldsymbol{Y})}{P(\boldsymbol{X})} \right]. \tag{4.12}$$

Having the data-fitting energy and the MRF energy at hand, we can define the total energy as $U(\boldsymbol{Y}) = U_M(\boldsymbol{Y}) + U_D(\boldsymbol{Y})$. To rephrase once again: smoothness energy component U_M arises from $P(\boldsymbol{Y})$ and U_D is inherited from the probability $P(\boldsymbol{X}|\boldsymbol{Y})$.

Therefore, we can relate the probabilistic decision rule to an energy based decision rule as follows:

$$\hat{\boldsymbol{Y}} = \arg\min_{\boldsymbol{Y}} U(\boldsymbol{Y}). \tag{4.13}$$

To calculate the probabilities of all possible $\boldsymbol{X}$ and $\boldsymbol{Y}$ configurations over the whole image range is not feasible. Finding the labeling $\hat{\boldsymbol{Y}}$ by minimizing the energy is therefore computationally heavy. To solve these problems sub-optimally, we utilize 2 different MRF energy optimization techniques, which are explained in the next sections.

4.2 Iterated Conditional Modes (ICM) Optimization

Besag [16] proposed a deterministic algorithm called iterated conditional modes (ICM) which maximizes local conditional probabilities sequentially. It is a coordinate descent based greedy algorithm. If the energy function to minimize is convex, then practical methods like ICM can be used to find the optimal segmentation. If the energy function is not convex as in our case, it may iterate to a local minimum of the energy, or to a local maximum of the probability function.

ICM starts from an initial label image $\boldsymbol{Y}^0$. For the success of the method, this initial input is crucial. Steps in ICM energy minimization are summarized in algorithm 5.

Algorithm 5 ICM Iteration

```
$\boldsymbol{Y} = \boldsymbol{Y}^0$
repeat
  for $s \in \boldsymbol{S}$ do
    for $y'_s = \{0,1\}$ do
      Calculate $u_M(y'_s)$
      Calculate $u_D(\boldsymbol{x}_s|y'_s)$
      Calculate $u(y'_s) = u_M(y'_s) + u_D(\boldsymbol{x}_s|y'_s)$
    end for
    $y'_s = \{0,1\}$
    $\hat{y_s} = \arg\min_{y'_s}(u(y'_s))$
    Update $\boldsymbol{Y}$
  end for
until Num. of MAX Iterations is reached
```

ICM is not guaranteed to converge to the global minimum energy as mentioned before. It is a greedy algorithm to find a local optimum value for $\boldsymbol{Y}$. The maximum number of iterations should be adjusted according to the requirements mostly based on time and processing speed.

4.3 Belief Propagation (BP) Optimization

Belief propagation is another technique to find the optimal labeling in images where MRF assumptions are applied [18]. In this method, the decision criteria at any number of iterations is to compare ‘Beliefs” of different classes for a pixel s. The label which is “believed” by neighbors of s to be the most probable label for s is selected in this method.

In every iteration, pixels are assumed to send so-called messages to their neighbors. At each iteration, all pixels are assumed to send these messages simultaneously, at the same time to each other. Iterations are executed from the receiving pixel’s perspective for every pixel. A message, $\boldsymbol{m}$, is a vector with the same number of values available for y. A message sent from pixel s to pixel r at time t is $m^t_{s\rightarrow r}$. $m^t_{s\rightarrow r}$ implies which label is the best fitting label for pixel r, from the perspective of pixel s.

In our approach, each pixel s passes energy values to its neighbors. Pixel s first assumes that its neighbor r has label value $y_r = 0$. For the possible label values y_s for itself, pixel s calculates two energies. The energy of having $y_r = 0$ when $y_s = 0$ is the first one. The second one is the energy of having $y_r = 0$ when $y_s = 1$. The minimum of these two energies is a part of the message from pixel s to pixel r. This is the belief from pixel s’s perspective on pixel r having label value $y_r = 0$.

After this, pixel s calculates two more energy values; the energy of having $y_r = 1$ when $y_s = 0$ and the energy of having $y_r = 1$ when $y_s = 1$. The minimum of these two energies is also sent inside the message to pixel r. Hence, two energies are sent from pixel s to pixel r, showing the belief of pixel s on the two possible labels on pixel r.

This algorithm is explained in Algorithm 6.

In Algorithm 6, the number of iterations is configurable. When iterations converge, there are two Belief values for pixel r; Belief($y_r = 0$) and Belief($y_r = 1$). The label with the smallest belief is going to be selected as the optimal label. Notice that this algorithm is a minimum sum algorithm, which corresponds to a max-product algorithm

Algorithm 6 BP Iteration

```
INIT: $m_{s \to r} = 0 \forall (s \in \boldsymbol{S}, r \in \boldsymbol{N}_s)$
Pixel Processing Order: $(0,0)$ to $(0, \text{lastcolumn})$ to $(1,0)$ to $(1, \text{lastcolumn}) \ldots$
repeat
  for $s \in \boldsymbol{S}$ do
    for $r \in \boldsymbol{N}_s$ do
      for $y'_r = \{0,1\}$ do
        for $y'_s = \{0,1\}$ do
          $u(y'_s) = u_D(\boldsymbol{x}_s | y'_s)$
          $V(y'_s, y'_r) = V_c([y'_s, y'_r])$
          $\text{MSG}(y'_s) = \sum_{k \epsilon \boldsymbol{N}_{s/r}} m_{k \to s}(y'_s)$
        end for
        $m_{s \to r}(y'_r) = \min\ [u(y'_s) + V(y'_s, y'_r) + \text{MSG}(y'_s)]$
      end for
    end for
  end for
until Num. of MAX Iterations is reached
for $y'_r = \{0,1\}$ do
  $\text{Belief}(y'_r) = u(y'_r) + \sum_{k \epsilon \boldsymbol{N}_r} m_{k \to r}(y'_r)$
end for
Select $\hat{y_r} = \arg\min_{y'_r} [\text{Belief}(y'_r)]$
```

for the probabilities. The algorithm operates on $-\log$ domain. This is different than sum-product algorithms, which try to find a global optimum for the whole image labeling $\boldsymbol{Y}$.

Using the belief values as the decision criteria, we both satisfy the data constraints and the spatial smoothness constraints on the image. We also use the MRF segmentation to produce a new weight image $\boldsymbol{V}_M$. This new probability image will be used in tracking compensation for segmentation later in Chapter 5.

Each pixel value in $\boldsymbol{V}_M$ at pixel s is $\boldsymbol{V}_M(s)$, and these values are the probabilities calculated from the MRF model. We follow a trivial approach to obtain these probabilities. Simply, the ratio of the belief on foreground label for pixel s to the sum of the beliefs for both label values gives the probability of foreground for the pixel. The formula below displays this way to obtain $\boldsymbol{V}_M(s)$:

$$\boldsymbol{V}_M(s) = \frac{\exp -(\text{Belief}(y_s = 1))}{\exp -(\text{Belief}(y_s = 1)) + \exp -(\text{Belief}(y_s = 0))}. \tag{4.14}$$

This is a consistent approximation with MRF models. The overall marginal probability is not maximized as in max-product algorithms. However, using this approximation approach, a new segmentation for the foreground can be realized by thresholding MRF probabilities using an experimental probability threshold value τ_p^{mrf}. So the new label for pixel s then becomes $V_M(s) > \tau_p^{mrf}$. Later on, this probability image will also be used in tracking in this thesis.

Performance results for MRF based segmentation results are provided in Chapter 6. It is shown that MRF improves segmentation success, by creating a spatial smoothness penalty constraint on the image. The results are obtained using BP optimization. Different and more complex MRF optimization techniques like graph cut based optimization [20, 21] could also be used; since these methods are sometimes more likely to converge to the globally optimal energy values. Graph-cuts is guaranteed to converge for binary labeling problems. On the other hand, BP peak results are comparable to these methods, and graph cut algorithms require a graph data structure and heavy computational burden as indicated in [22]. For this reason, in the context of this thesis, it is not strictly necessary to use other optimization algorithms like graph cuts.

Smoothness parameter θ in equation 4.7 defines the balance between the ISG information and the smoothness penalty. When this parameter is high, the resulting probability image becomes very smooth. This might result with an over-smoothing. In the other case when the parameter is too low, then there is no effect of spatial correlations on the segmentation. The results will completely be dominated by the ISG model. We selected this parameter experimentally, in a way not to allow an over-smooth labeling while incorporating the smoothness information into the model.

Next chapter explain how to integrate tracking information using mean-shift tracking with the base model.

Chapter 5

Integrating Tracking Information

In this chapter, we explain how temporal continuity information can be integrated into the ISG model and the MRF model based foreground segmentation methods, using mean shift tracking. Mean shift algorithm tracks a target by estimating its location. The ISG model hence the MRF model does not utilize location information. Every pixel is modeled independently from others in the ISG model, and even though the MRF model considers local smoothness constraints on the image, it does not use the foreground location and movement information. We show that the integration of a tracker with the segmentation methods increases segmentation performance drastically. This provides a robust elimination of outliers detected as foreground objects in ISG/MRF models. As an additional argument which does not try to achieve better foreground segmentation, but tries to create a tracker; we also show that the probability images $\boldsymbol{V}_I$ and $\boldsymbol{V}_M$ obtained via the ISG and the MRF models can be used as weight images in robust object tracking with mean shift.

In the first section we will explain the mean shift algorithm and object tracking based on it. The next sections provide information on how tracking can be used to improve ISG/MRF foreground segmentation. Chapter 6 contains all the experimental results on tests that we have conducted with the tracker compensation on foreground segmentation.

5.1 Mean Shift Tracking

Mean shift algorithm was introduced in [23] to find the mean of a probability density. It was used to calculate the center of gravity for a given cluster of densities. In tracking context, densities correspond to target object probabilities for pixels. Densities are also referred as weights. Therefore, mean shift finds the most probable location for the target, which is the mean of the target probability distribution provided to the algorithm inside a frame. The use of mean-shift under tracking context is explained in detail in [24].

In mean shift tracking, an assumption based on target movement is necessary. At time t for a video, it is assumed that target object in the next frame, frame $t+1$, lies within the proximity of its current location in the current video frame, frame t. The region around the current location of the target is called as the proximity region. In short, target will be searched inside this proximity region in the next frame, instead of analyzing the whole frame to find it. Because of this assumption, it is enough to define target probabilities for pixels s which lie in this proximity region, as implemented in [27].

Target probabilities of pixels are necessary for mean shift. For each pixel, a probability of belonging to the target object should be available. This input is usually provided not in a mathematical probability distribution function format but in an image format. We will refer to this probability image as $\boldsymbol{V}$. An example for this kind of images is the $\boldsymbol{V}_I$ image we can obtain using the ISG model, or the $\boldsymbol{V}_M$ image we obtain using MRFs and Belief Propagation; as explained in previous chapters. Every v_s value on this image on pixel s are probability values. In the way we have been expressing foreground probabilities until now using ISG/MRF models, we can state that $v_s = P(y_s = 1|\boldsymbol{x}_s)$. It defines the weight to be assigned to pixel s during mean shift. For this reason, $\boldsymbol{V}$ is also referred as the weight image.

There are different ways to obtain the weight image $\boldsymbol{V}$. In traditional mean shift trackers, only target color features are used to obtain this image. As an example to such methods; color histogram of the target object being tracked is formed and backprojected to each pixel in the image to create foreground probabilities in [28]. In this work, if a pixel has similar colors to those of the target, due to the backprojection which creates a probability from the target histogram and the pixel color, this pixel will have a high weight assigned on it. In Avidan's work [27], a group of classifiers -the classifier ensemble- is trained on both target colors and non-target colors. This is a discriminative method of training the tracker. Then, the ensemble of classifiers assigns a weight to each pixel, showing how probable classifying each pixel as target object is. This discriminative analysis makes classifier based trackers perform better than traditional mean shift trackers.

We will now explain how mean shift operates on images, and how an object can be tracked in a video. To rephrase the first step, a cluster of weights should be provided to the algorithm. Mean shift will operate on this cluster and using moments of the weights, it will detect the center of gravity of the cluster. For tracking context, this cluster of weights is actually the target probability image, $\boldsymbol{V}$.

To remind, the probability image may also be referred as the "confidence map". v_s values should be in the interval $[0, 1]$ since these are target probabilities. Notice that if a pixel s is very likely to be a part of the target object, then the probability value v_s at this pixel location in the probability image is going to be higher. Therefore, when $\boldsymbol{V}$ is displayed as a grayscale image with intensities starting from 0 up to 1, pixels which are more likely to be target pixels will be displayed brighter than others. Mean shift is expected to find the mean location of this bright object.

During initialization, user must provide the location and size of the object that is going to be tracked. This is usually done by drawing a rectangle just around the object to be tracked on the first frame where the target is apparent. Initial location should be known to the algorithm so that the iterations can start from a correct location. Mean-shift takes this rectangle around the object as an object window $\boldsymbol{S}_O$ for the frame. Then, around this object window which precisely contains the object, a larger window is placed. This larger window is called the search region window $\boldsymbol{S}_R$, and it is a proximity region which is expected to contain the object in the next frame. Object will be searched within this window in the next frame. $\boldsymbol{S}_O$ will be automatically repositioned in the next frame inside $\boldsymbol{S}_R$, depending on the result of the mean shift algorithm and the direction of the shift in object location. Size of $\boldsymbol{S}_R$ is larger than the object window $\boldsymbol{S}_O$, but it does not occupy the whole video frame either. This brings some efficiency in calculating the mean. The probabilities only for pixels s which are inside this search $\boldsymbol{S}_R$ are enough to obtain, since mean shift operates only inside $\boldsymbol{S}_R$. This relies again on the assumption that in the next frame, target will still lie within the proximity of its current location.

A selection for the search region window size can be a static value like two times the size of the object frame.

Once the probabilities are assigned inside the search region window $\boldsymbol{S}_R$, then within the object window $\boldsymbol{S}_O$ on image $\boldsymbol{V}$ in the previous frame, mean location $s_m = (i_m, j_m)$ of the probability values is found by mean shift in a trivial way. After the new mean is found, $\boldsymbol{S}_O$ is updated according to the current frame. First, the moments of the weights inside $\boldsymbol{S}_O$ are found as shown below:

$$M_{00} = \sum_{\forall (i,j) \epsilon \boldsymbol{S}_O} \boldsymbol{V}(i,j), \tag{5.1}$$

$$M_{10} = \sum_{\forall (i,j) \epsilon \boldsymbol{S}_O} i\boldsymbol{V}(i,j), \tag{5.2}$$

$$M_{01} = \sum_{\forall (i,j) \epsilon \boldsymbol{S}_O} j\boldsymbol{V}(i,j). \tag{5.3}$$

Notice that value $\boldsymbol{V}(i,j)$ is the same as $\boldsymbol{V}(s)$ since $s = (i,j)$. This is the the weight value on pixel s. Moments calculated above are 0^{th} and 1^{st} moments of the probability values (weights) inside $\boldsymbol{S}$. Using these values, we can then find the coordinates of the mean of this distribution as:

$$i_m = \frac{M_{10}}{M_{00}}, \tag{5.4}$$

$$j_m = \frac{M_{01}}{M_{00}}, \tag{5.5}$$

where $s_m = (i_m, j_m)$ is the mean of the weight image inside the object window $\boldsymbol{S}_O$ on probability image $\boldsymbol{V}$. When the mean is found, $\boldsymbol{S}_O$ is centered around the new mean. These iterations continue until a convergence decision on the mean of the target is made. Convergence of mean shift is proved in [31]. In our implementation, two criteria are used together to decide when the calculation of the mean converges. One of them is the shift threshold. When the shift between a new mean and the previous one is less than a threshold ϵ, the new mean is provided as the ultimate result of the mean finding operation. As a second criteria, the maximum number of iterations is also configurable.

This ensures that algorithm loop does not continue for a long time to impact processing speed.

To state the rule to find the mean in a single iteration, we can rephrase equations 5.4 and 5.5 in a more generic way as:

$$s_m = \frac{\sum\limits_{s \epsilon \boldsymbol{S}_O} v_s s}{\sum\limits_{s \epsilon \boldsymbol{S}_O} v_s} \tag{5.6}$$

The question at this point is how to obtain a robust and reliable weight image that gives a correct estimation of the probability distribution of the pixels for a tracked object. As mentioned briefly above, there are different approaches which address this question. In practical mean shift implementations like [28], color histogram information of the tracked object is used to create this distribution using backprojection of the histogram. This is a simple and fast way of implementing mean shift method, and it can be robust where the tracked object is significantly different from the non-target pixels in terms of its color. Below are the steps to implement this approach:

Algorithm 7 Traditional Mean Shift Tracker

```
$\boldsymbol{X}_t$ = Video Frame at time $t$
Assume: $\boldsymbol{X}_1$ is the first frame with the target
Mark $\boldsymbol{S}_O$ in $\boldsymbol{X}_1$
**Hist** = A Color Histogram of $\boldsymbol{S}_O$
**for all** $\{\boldsymbol{X}_t | t = 1$ to Number of Frames$\}$ **do**
  Convert $\boldsymbol{X}_t$ into the related color space
  **repeat**
    Calculate $v_s$ $\forall s$ inside $\boldsymbol{S}_O$ with backprojecting **Hist**
    Calculate $s_m = (i_m, j_m)$ inside $\boldsymbol{S}_O$
    Center $\boldsymbol{S}_O$ around $s_m$
  **until** $s_m$ converges
**end for**
```

In Algorithm 7, the convergence rule "until s_m converges" actually has two conditions inside. Either the maximum number of iterations is reached, or the difference between the new mean and the previous mean is less than a predefined threshold, ϵ, as defined before.

In [27], instead of using color histograms, a group of classifiers is trained on the target and non-target pixels and used to form the distribution for the target. Here is how this method works:

Algorithm 8 Classifier Based Tracker

```
$\boldsymbol{X}_t$ = Video Frame at time $t$
Assume: $\boldsymbol{X}_1$ is the first frame containing the target
Mark $\boldsymbol{S}_O$ in $\boldsymbol{X}_t$
Calculate $\boldsymbol{S}_R$ around $\boldsymbol{S}_O$
Provide class labels for target pixels inside $\boldsymbol{S}_O$
Provide class labels for non-target pixels inside $\boldsymbol{S}_R$ - $\boldsymbol{S}_O$
Extract features $\boldsymbol{F}$ for target and non-target pixels inside $\boldsymbol{S}_R$
Train Classifiers using $\boldsymbol{F}$
for all $\{\boldsymbol{X}_t | t = 1$ to Number of Frames$\}$ do
  Calculate $v_s$ $\forall s$ inside $\boldsymbol{S}_R$
  repeat
    Calculate $s_m = (i_m, j_m)$ inside $\boldsymbol{S}_O$
    Center $\boldsymbol{S}_O$ around $s_m$
  until $s_m$ converges
  Calculate the new $\boldsymbol{S}_R$
  Extract features $\boldsymbol{F}$ for target and non-target pixels inside $\boldsymbol{S}_R$
  Update Classifiers using $\boldsymbol{F}$
end for
```

It is important that Avidan's classifier tracker is discriminative, unlike traditional mean shift trackers which do not consider non-target pixel colors at all. This is proven to outperform traditional trackers [27].

There is also a scale invariant version of the mean-shift algorithm, which has a dynamic size for the object window, instead of a static one. This method is named as Continuously Adaptive Mean Shift(CAMSHIFT) and introduced in [28]. In this method, the only difference is that size of the object window in a new frame is dynamic. It is a function of the 0^{th} moment of the older object window being repositioned on the new frame. This moment, M_{00}, gives a general idea on the scaling changes of the image, even the rotational changes. If the 0^{th} moment is much less then expected, this may point to a scale change, meaning that target object size has been reduced, and vice versa. Scaling factors is up to the implementation.

5.2 Leveraging Mean Shift Tracking for Segmentation

The ISG/MRF models we have used to segment the observed image $\boldsymbol{X}$ and obtain a label image $\boldsymbol{Y}$, do not use any location information on the target object. When combining tracking approaches with color modeling based foreground segmentation, words "target" and "foreground" gets the same meaning. Target for the tracker is the foreground for the ISG/MRF models.

In the previous section of this chapter, we explained how the location of the target can be estimated using the mean shift trackers. Intuitively, if we can estimate the location of the target using a manual input for the first frame that contains the target foreground object, and then utilizing a mean shift tracker; we can also use this information in segmentation with the ISG/MRF models. A very simple approach can be to eliminate pixels s labeled as foreground in label image $\boldsymbol{Y}$ if $s \notin \boldsymbol{S}_O$. We can remove all outliers that are misclassified as foreground away from the object location in this manner.

We implemented three different tracker based methods to obtain a labeling and to find a new $\boldsymbol{Y}$. These methods are:

- On label image $\boldsymbol{Y}$ retrieved via the ISG model if there are any foreground marked pixels s where $s \notin \boldsymbol{S}_R$ of the tracker for the current frame; then this is taken to be a misclassification. According to tracker's search window $\boldsymbol{S}_R$, there cannot be any foreground outside the search region. Then, for the new $\boldsymbol{Y}$, we re-label s as $y_s = 0$ in the previous $\boldsymbol{Y}$ of the ISG model, and mark s as background. The performance results of this outlier elimination on ISG model using the tracker are provided in the results chapter.

- We utilized a decision tree based classifier which returns foreground probabilities for the pixels inside $\boldsymbol{S}_R$ for every frame as a label. If we name this new probability image as $\boldsymbol{V}_T$, then we directly use a probability based thresholding on $\boldsymbol{V}_T$ based on a probability threshold τ_t. If the probability of foreground for a pixel is lower than this threshold, then the pixel is taken to be background. This method is similar in terms of eliminating all outliers outside the $\boldsymbol{S}_R$ of the tracker like the first method, but inside $\boldsymbol{S}_R$, this method relies on the probability of the tracker, instead of the ISG model.

- As the final method we experimented, we combine $\boldsymbol{V}_I$ and $\boldsymbol{V}_M$ of the ISG and MRF models with the probability calculated via the tracker, $\boldsymbol{V}_T$. We take a weighted average of ISG/MRF based probabilities and the tracker probabilities. This weight is dynamic and depends on how stationary the foreground object is.

Using ISG or MRF models in this combination result in similar performance, although MRF and the tracker combination results in a slightly better performance. The results are displayed in the next chapter.

This average between the tracker and the ISG/MRF is altered and dynamic, depending on the movement rate of the foreground object that can be learned via a history of object locations. We keep track of object window locations using the mean shift tracker, and we compare these locations between the frames to get an idea of the movement speed of the foreground object. If the movement rate is low, then, as it was explained before, the ISG/MRF foreground probabilities for the real foreground pixels drop significantly, since the object begins blending into the background. Therefore, according to the movement rate, we increase or decrease the effect of the tracker probability on the combination of ISG/MRF and the tracker probabilities. If the movement rate is high, then we rely more on the tracker instead of the ISG/MRF models. The details of this algorithm is given in Algorithm 9.

.

We used $\hat{\boldsymbol{V}}$ to save the new probability results of the combined model which averages the probabilities $\boldsymbol{V}_I/\boldsymbol{V}_M$ calculated via ISG/MRF models with the probabilities $\boldsymbol{V}_T$ calculated via the tracker. Each pixel s on this image has the probability value $\hat{v_s}$, which shows how likely that pixel is to belong to the target, from the combined model perspective.

Algorithm 9 describes the step of our implementation for this option to utilize target movement information in ISG/MRF segmentation:

In algorithm 9, $\hat{v_s}$ keeps an averaged target probability value for pixel s. $\tau_o(t)$ defines the weight of making the ISG/MRF model or the tracker more dominant in this weighted average. $\tau_o(t)$ also depends on the learning rate ρ for the ISG model, and how fast we want foreground to be accepted as background. The advantage of the combination is to compensate for a major disadvantage of using the ISG model. When the foreground is stationary, the ISG model hence the MRF model will classify it as background after a while, depending on the learning rate of the model. Therefore, the tracker can inform the ISG/MRF model on the location and the movement rate of the object. The

Algorithm 9 Using Motion for ISG/MRF Segmentation

```
Ensure: Mean-Shift runs in parallel With ISG/MRF
Ensure: Mean-Shift returns $\boldsymbol{S}_O$ for every frame
Ensure: Mean location vector $\boldsymbol{L}$ keeps last 10 object windows ($\boldsymbol{S}_O$)
  for all $\boldsymbol{X}_t | t = 1$ to Number of Frames do
    for all $s \in \boldsymbol{S}_R$ do
      $v_s$ = probability from the ISG or MRF
      $v'_s$ = probability from the tracker
      $\tau_o(t)$ = Incremental overlap rate between $\boldsymbol{L}(t)$ to $\boldsymbol{L}(t-10)$
      $\hat{v_s} = (1 - \tau_o(t))v_s + \tau_o(t)v'_s$
    end for
  end for
```

tracker can obtain a robust averaging weight $\tau_o(t)$, by measuring the distance between object windows in consequent frames, which is an indicator of the movement rate for the target. Then, decision-tree based foreground probabilities can be more dominant for the segmentation, since the ISG/MRF models blend the stationary foreground into the background. However, the tracker still knows where the foregorund is, hence to use the tracker probability is more reliable in such scenarios. Local color based classification will dominate the labeling decision where foreground remains stationary for a long time. But still, ISG or MRF models will have a minor effect on the decision as long as the $\tau_o(t)$ parameter does not increase extremely.

Optionally, when the foreground object should be considered as the new background in a scene, an additional configuraton on the implementation can surely be made; if the foreground must be blended into the background after some time. In that case, $\tau_o(t)$ can be reset to an initial value not to over-increase. The initial value for τ_o should favor the ISG/MRF model, and begin favoring the tracker in case of stationary foreground scenarios.

Performance results for the three different ways to use tracking for segmentation which are mentioned in this section will be shown in Chapter 6.

In the next section, we explain how the ISG/MRF models can be used to “track” the object. In short, next section is on a new tracker, that utilizes the probability image $\boldsymbol{V}_I$ of the ISG model and the probability image $\boldsymbol{V}_M$ obtained by the MRF model.

5.3 Using ISG and MRF Models for Tracking

From the previous chapters, we introduced how we obtain foreground probabilities for all pixels in $\boldsymbol{X}$. These probability images were $\boldsymbol{V}_I$ which was obtained from the ISG model, and $\boldsymbol{V}_M$ which was obtained from the MRF model.

Since mean shift trackers rely on a weight image, these probability images above can be used as a weight image for the tracker. The ISG model keeps track of observed values on each pixel by updating the related Gaussians inside the pixel mixture. This model takes temporal relations of the pixel values $\boldsymbol{x}_s$ into consideration. On the other hand, MRF model also uses the ISG model to obtain this temporal correlation between the different observations $\boldsymbol{x}_s$ on pixels s through time; and MRF model also incorporates spatial smoothness constraints. In belief propagation based MRF optimization which was explained in Chapter 3.2.3, every pixel in a sense warns their neighbors on which class they should belong to. As the ultimate result, an MRF based probability image $\boldsymbol{V}_M$ is formed by combining the belief of the neighbors together with the belief of the pixel itself, on being a foreground pixel. This gives the MRF based foreground probability for the pixel.

Traditional and classifier based mean shift trackers that were explained through this chapter rely on local features of the target object. They do not keep track of temporal or spatial relations between the pixels. The probability image that is formed in these trackers is local, mostly based on color features. Intuitively, using the ISG or the MRF models to assign target (foreground) probabilities to each pixel inside the search region window $\boldsymbol{S}_R$ of the mean shift tracker would be more robust than other probability assignments.

Consider a scenario where a traditional or a classifier based tracker is used. At a moment in time, the features of the target pixels might be very similar to the features of the non-target pixels. In these cases, the tracker would get false probabilities, false weights and the mean of the object window would not be estimated correctly. This would also make the tracker get stuck because the object window $\boldsymbol{S}_O$ would remain on the same position for some time due to the fact that non-target pixels with similar features to the target are stationary. This would affect the tracker drastically and make it impossible to recover. For this reason, ISG and MRF based weight images would be more robust.

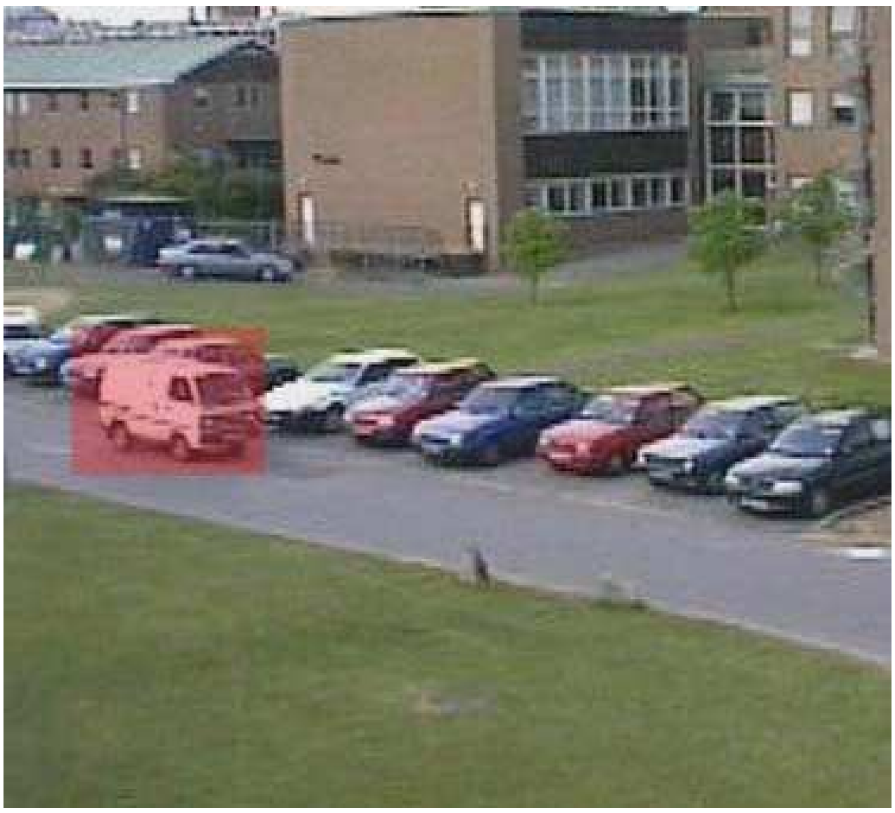
Even when the features of the background and the foreground are similar, the ISG model, if converged for some time, would be able to separate the target better since the Gaussians inside the ISG model would be very precise.

In a feature based tracker, if the search window involves some non-target pixels with target-like features, then the calculation of the object window fails. In ISG/MRF model case, the model would know that those non-target pixels have been containing similar values for a while, hence they are background but not foreground. A sample frame is displayed in Figure 5.1. It can be seen on that figure that color features of the background and foreground are similar for a part of the frame, and when the search window involves those non-target pixels, the object window shifts away from the real foreground object.

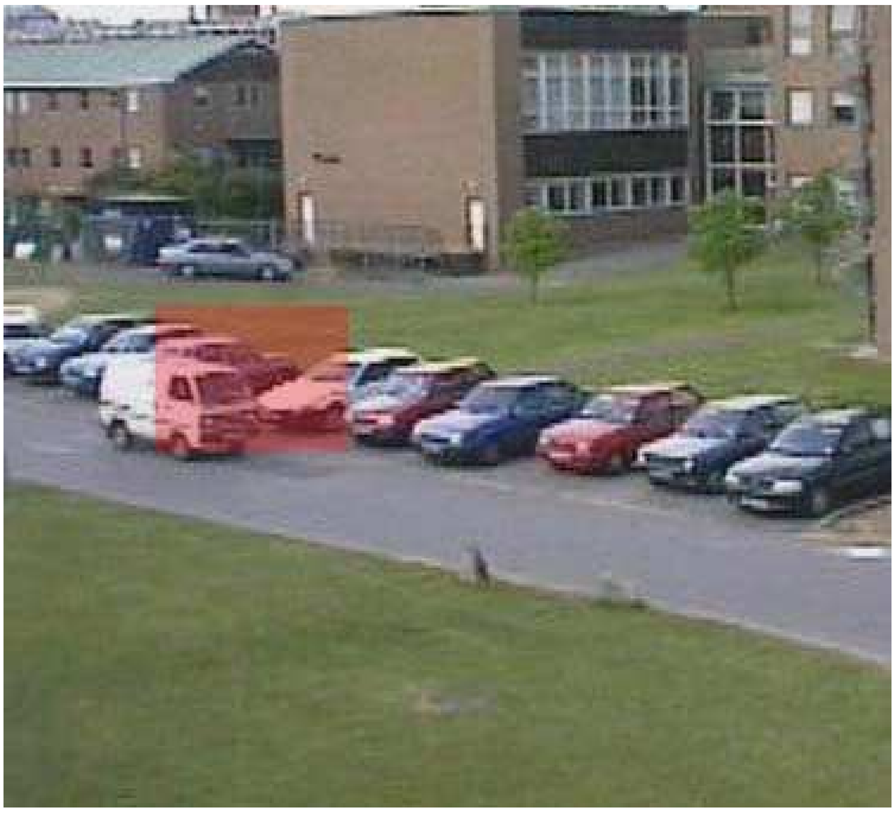
Considering another scenario where the foreground is very stationary, and the ISG/MRF model probabilities tend to favor the background; using the probabilities of the classifier based tracker can be advantageous. Keeping the movement rate of the foreground object and the location of the object; if the object moves very slowly, then the significance of the probabilities calculated by the classifier based tracker can compensate for obtaining correct weights. Otherwise, when the object movement rate is high, probabilities calculated by ISG/MRF can have higher significance in determining the weights for the mean-shift algorithm.

We combine two trackers as mentioned above. Calculation for the movement rate of the foreground object is no different than what is explained in Algorithm 9 in the previous section that is used for segmentation. The same parameter, $\tau_o(t)$ is used to quantify the movement rate and the probability $\hat{v_s} = (1 - \tau_o(t))v_s + \tau_o(t)v'_s$ is used as the mean-shift weight for pixel as, as in Algorithm 9.

Experimental results and comparison to other trackers of the ISG and MRF based tracking methods are provided in the next chapter, which contains all the results for all methods in this thesis.



(a) ISG/MRF based Tracker



(b) Classifier based Tracker

FIGURE 5.1: Tracker Comparison

Chapter 6

Experimental Results

6.1 Information on Test Data and Experiments

In this chapter, we present all experimental results of different methods which are explained throughout the thesis. In the first section, general information on the tests and the test data is presented. Then, the performance results and parameters for the following methods are given:

- Standard ISG model based foreground segmentation and probability thresholding based ISG segmentation introduced in Chapter 2,
- Hysteresis thresholding introduced in Chapter 3,
- MRF probability based segmentation introduced in Chapter 4,
- Purely decision tree classifier based tracker segmentation where a probability threshold is applied directly on the search window. In addition, the tracker compensated foreground segmentation methods using the ISG and the MRF models introduced in Chapter 5,
- Object tracker using the ISG and MRF probabilities introduced in Chapter 5.

All our tests were conducted in an Intel Processor based PC at 2.0Ghz, with 4.00 GB of RAM. Intel OpenCV image processing library for C language was used as the foundation library. 8 videos in total were analyzed for all different methods. All these videos have also their groundtruth information, either in the format of a black and white video, or text files pointing to foreground region windows for each frame in the video. 6 of these videos are taken from VSSN06 segmentation competition test data [29]. 2 videos were taken from the 2nd IEEE International Workshop on Performance Evaluation of Tracking and Surveillance [32] datasets. VSSN videos are artificially formed and their groundtruth is pixelwise matching with the real foreground. PETS2001 videos are observations of real scenes and the groundtruth for these videos are blockwise matching with the foreground. In other words, not the exact foreground object but a precise rectangle around the object is given as groundtruth in these videos. A fixed camera is used for both datasets. In terms of all experimental parameters, we tried to come up with the best fitting values for most of the scenarios, and then for a robust comparison, we kept those parameters constant and the same for all methods.

All parameters that were explained through the thesis are taken as constant for all the tests. The values we use are:

$\tau_p = 0.6$, $\tau_w = 0.75$, $\tau_r = 0.5$, $\tau_e = 120$pixels, $\kappa = 0.05$, $\tau_p^{mrf} = 0.6$, $theta = 0.3$, Initial $\tau_o = 0.2$, $\tau_t = 0.5$, $\alpha = \rho = 0.02$, $\beta = 0.1$, $\sigma_{init} = 25$, initial weight $= 0.05$ and $\lambda = 0.85$.

All videos below have 384x240 pixels, with different number of frames. In Table 6.1, information on the test videos is given. The duration of the each video and the learning period for all the videos are displayed in terms of seconds. During the learning period, performance parameters are not calculated. Subjective criteria like “background movement” and “BG-FG similarity”, “how stationary the foreground is” are represented in three levels: high, medium and low. These levels are assigned according to the observations and the averages within the test group.

	GMM Test Videos				
Video	Duration	BG Movement	BG-FG Similarity	Stationary FG	Learning Period
Video1	16	Low	Low	Medium	5
Video2	36	High	Medium	High	10
Video3	32	High	Low	Medium	10
Video4	32	High	Low	Low	10
Video5	30	Low	High	Low	10
Video6	30	Medium	Low	Low	10
Video7	90	Low	Low	Low	30
Video8	90	Medium	High	Low	30

TABLE 6.1: Characteristics of Different Test Videos

(a) Test Video 1

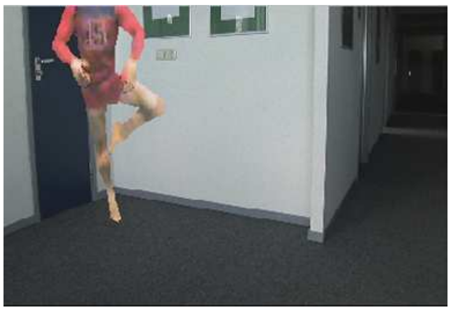
(b) Test Video 2

(c) Test Video 3

(d) Test Video 4

(e) Test Video 5

(f) Test Video 6


(g) Test Video 7

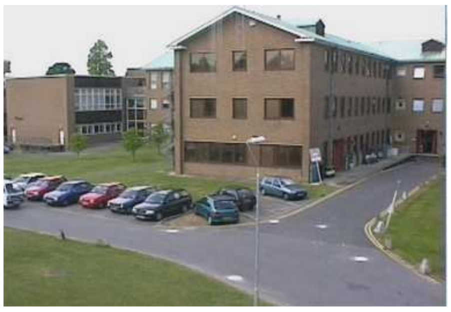
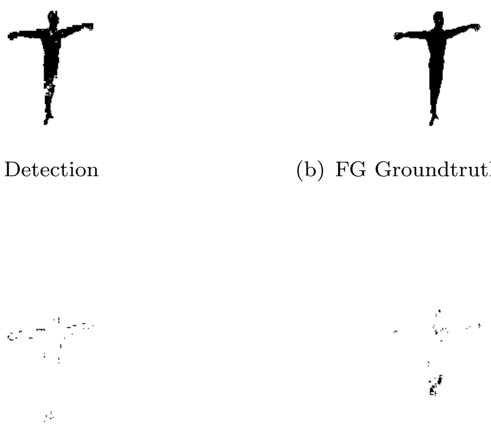

(h) Test Video 8

FIGURE 6.1: Test Video Screenshots

Screenshots from the videos given in Table 6.1 are provided in Figure 6.1 for a reference to the mentioned video databases.

Now, the experimental results and the discussion on these results are going to be presented for each method. Then, an overall comparison discussion will be given. In the



(a) FG Detection (b) FG Groundtruth

(c) False Alarms (d) Missed Pixels

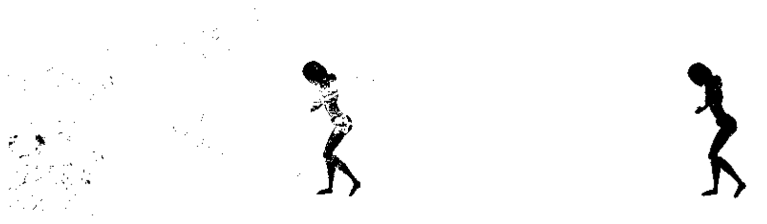
FIGURE 6.2: ISG Results - Test Video 1

last section of the chapter, results on the tracking method which utilizes the ISG/MRF probabilities will be provided.

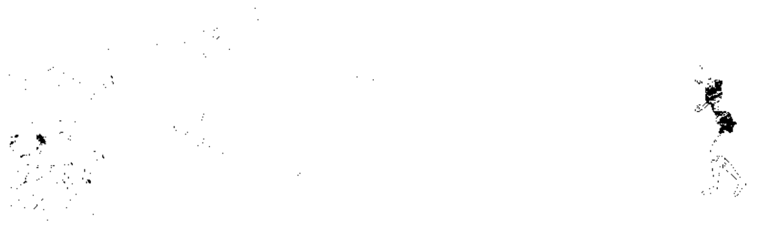
6.2 Experimental Results for ISG Model

We will first visualize the results of the standard ISG method, addressing the major problems again. Specific screenshots from random test videos are displayed for discussion purposes. In Figure 6.2, ISG detected foreground can be seen next to the groundtruth. False detections (false alarms) and misses can also be seen. This video does not contain extreme lighting changes and background is very stable. This makes the detection almost flawless. Some pixels that coincidentally have similar foreground colors with the background colors are missed. False alarms are usually around the object. This is due to pixel-wise value misreadings of the image from the groundtruth video. Even so, false alarms are also neglected.

In Figure 6.3, a video with lots of movement in the background (tree leaves in the wind) is shown. There are also very sharp lighting changes since this is an outdoor scene. This is one of the hardest videos in terms of modeling the background colors. As it can be seen, false alarms are quite a lot in this segmentation, since some tree leaves moving in the wind are detected as foreground. In the long run, for pixels where leaves are moving, number of false alarms decreases by time, since the model adapts the



(a) FG Detection (b) FG Groundtruth



(c) False Alarms (d) Missed Pixels

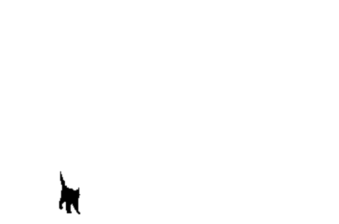
FIGURE 6.3: ISG Results - Test Video 2

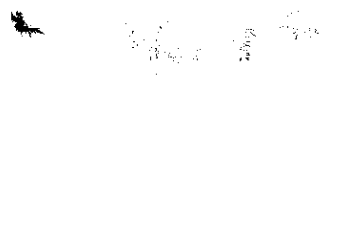
background to those movement changes. In this video, the foreground object (human shape) is turning around itself, hence some parts of the object remain stationary for some time. This makes the ISG model believe that some of the pixels on the foreground object are actually background now, since they become dominant in their Gaussian mixture based color distribution. In this case, the foreground partially blends into the background. Modifying the learning parameter for the ISG model according to each different scenario would be effective in such cases.

The video in Figure 6.4 is similar to the one in Figure 6.3, but here foreground objects pass through the scene very quickly. The problem of moving background still exists: there are moving tree leaves. But it can be seen that the number of missed pixels significantly decrease compared to the previous video, when foreground is not very stationary. The time foreground stands still is not enough for foreground colors to blend into background colors in the Gaussian mixtures, therefore resulting in a better segmentation of the foreground.

There is also another video with a car passing as a foreground object in 6.5. Here the issue is: parts at the back of the car have very similar colors with the background. ISG model takes these parts directly as background, since they fit well in the existing background distribution. At this point, one good solution could be keeping track of the size and location of the car, and feed this information to the background model

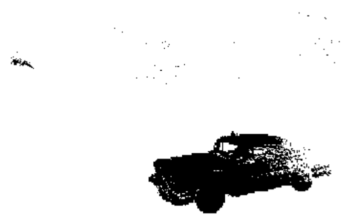
(a) FG Detection

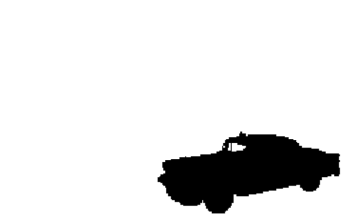

(b) FG Groundtruth


(c) False Alarms

(d) Missed Pixels

Figure 6.4: ISG Results - Test Video 3


(a) FG Detection


(b) FG Groundtruth

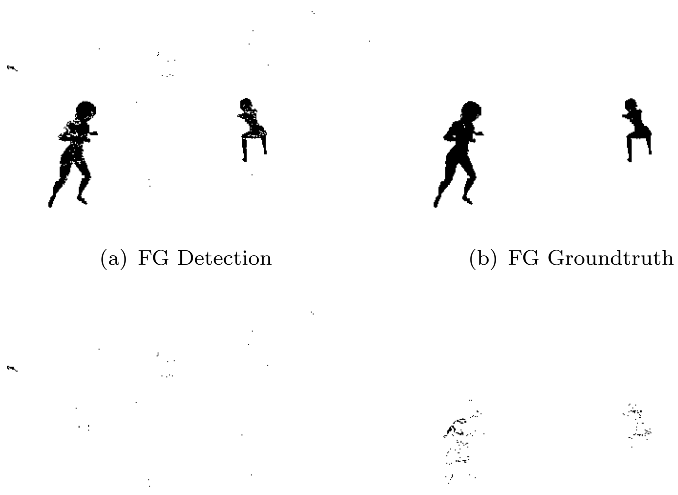
(c) False Alarms

(d) Missed Pixels

Figure 6.5: ISG Results - Test Video 4

with a weight. This scenario is a good example to show when tracking compensated segmentation can be useful.

Figure 6.6 shows a scenario where there are multiple foreground objects entering a scene. Foreground objects partially have similar colors with the background, hence there are some small clusters of missed pixels on foreground regions. Since the ISG model is completely pixelwise, any pixel that does not behave as expected will be classified as foreground; since the number of foreground objects is not very relevant in ISG. When


(a) FG Detection (b) FG Groundtruth

(c) False Alarms (d) Missed Pixels

FIGURE 6.6: ISG Results - Test Video 5

a smoothing approach like the hysteresis thresholding or the MRF model is used, this kind of problems are expected to be alleviated.

In most of the figures above, it can be noticed that foreground pixels are completely independent. There are some pixels detected as foreground, and some others very close but these are detected as background. Then, another small cluster of foreground pixels follow. This shows that the model may be improved if the spatial correlation of pixels is taken into consideration.

In some of the test videos where background is stationary for some time to allow the ISG model to learn the foreground colors as the background, then the detection performance decreases significantly. Almost all foreground objects will be detected as background; and when foreground starts to move again, background appearing out will be detected as foreground. This can be handled with an additional technique, tracker compensation, which warns the background model for stationary foreground objects.

Performance parameters used to measure the segmentation performance are taken from the VSSN06 competition. These performance parameters are calculated as:

$$\text{precision} = \frac{\#\text{detected foreground pixels} - \#\text{false alarm pixels}}{\#\text{detected foreground pixels}}, \tag{6.1}$$

$$\text{recall} = \frac{\#\text{real foreground pixels} - \#\text{misses}}{\#\text{real foreground pixels}}. \tag{6.2}$$

Precision parameter shows how much of the detected pixels really belong to the foreground. Recall parameter shows how much of the foreground was detected and how much was missed. Taking precision and recall values as percentages, a single parameter that represents the overall segmentation success can be also used and it is given by:

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{6.3}$$

Table 6.2 shows the results on 8 different videos using the ISG model and Gaussian weight based segmentation. These are the standard results for the ISG model without any incorporation of direct probability based thresholding.

	ISG Model Performance - Weight Based							
	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	96.7	45.1	55	63.1	93	60.6	93.9	65.5
Recall(%)	86.3	49.8	60.1	71	81.5	85.8	82.7	88.1
F1 Score	0.91	0.47	0.57	0.66	0.86	0.71	0.87	0.74

Table 6.2: Results - Standard ISG Performance

Table 6.3 shows the result of the probabilistic thresholding using the ISG model. In this method, ISG is used to extract class probabilities, and a direct foreground probability thresholding per pixel is applied with a threshold value, τ_p. It can be seen that the results are usually similar to the weight based thresholding, since the base model for both options is the same. The only difference appears when a foreground object has different colors than the previous foreground objects observed through the video. In this case, foreground based thresholding is slightly better than weight based thresholding. In general, probabilistic thresholding using ISG results at least as good as the regular ISG method, even better. In addition, calculation of these probabilities enables many enhancements like MRFs and trackers. The probability image calculated using the ISG model for direct thresholding is shown in figure 6.7.

These tests are made by keeping all experimental parameters constant. These parameters can be altered according to the specifics of the video to be segmented. For instance, if we change the probability threshold value τ_p to different values ranging from 0 to 1,

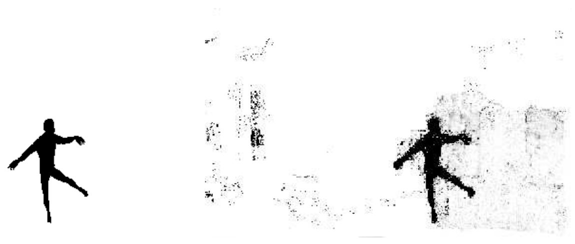


FIGURE 6.7: Probabilistic ISG Output

	ISG Model Performance - Probability Thresholding							
	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	96	43.7	57.6	58.2	93.2	58.1	94.2	66.2
Recall(%)	90.3	51.1	61	72.8	80.4	86	82.5	88.3
F1 Score	0.93	0.47	0.59	0.64	0.86	0.69	0.88	0.73

TABLE 6.3: Results - ISG Performance with Probability Thresholding

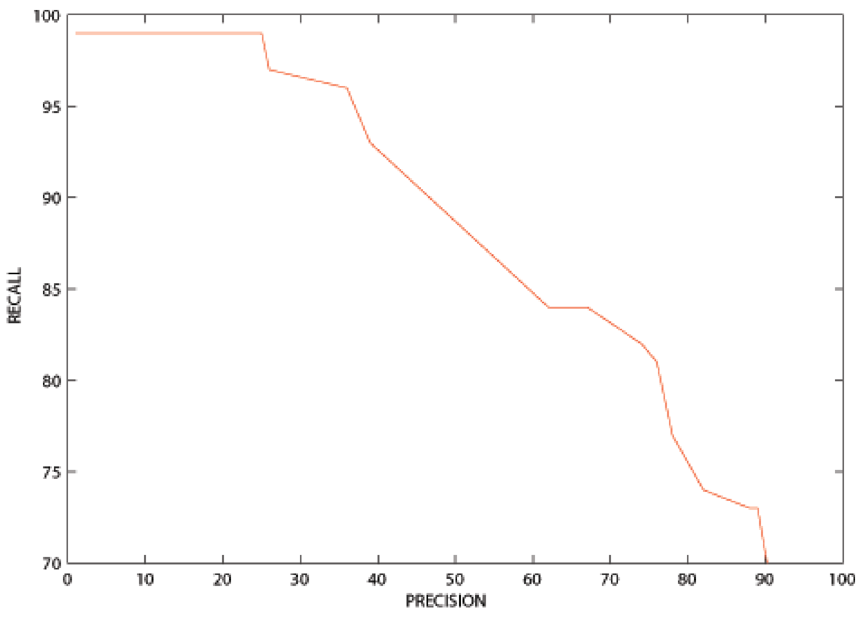
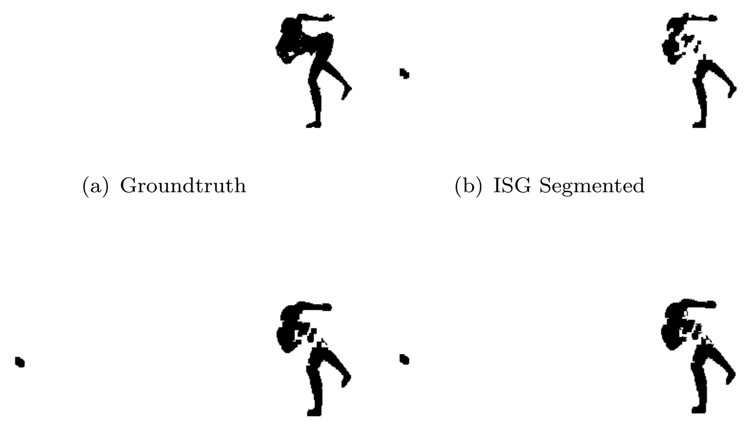


FIGURE 6.8: Precision-Recall Curve Example - Video 6

for each different value the results will be different. Each threshold value will result in a different pair of precision and recall parameters. This concept is depicted with the sample precision-recall curve for test video 6 in Figure 6.8. It is seen in this figure that increasing the threshold makes the segmentation more precise, however, decreases the recall rate. Less of the real foreground is detected with each increase in the probability threshold, as expected.


(a) Groundtruth (b) ISG Segmented

(c) Hysteresis Segmented (d) Edge Contours Incorporated

FIGURE 6.9: Hysteresis Segmentation Results

About the computation load of the algorithm, enough memory to allocate Gaussian mixture models for each pixel in the image is necessary. A single Gaussian mixture model has K number of Gaussians for a pixel at most; and a mean, a variance and a weight for each Gaussian. Every test video contains 384x240 pixels. So, $3 \times K$ variables for each of these pixels occupy the memory. With this computational load on our test PC, the ISG model operates at a frame rate of 38. When the additional probability calculations are made to extract foreground and background probabilities for all pixels, the frame rate drops down to 35 due to the additional computational load. Still, this rate is more than acceptable for real time applications.

6.3 Experimental Results for Hysteresis

Hysteresis thresholding combined with the edge contour information and some post-processing was done to address the spatial correlations between pixels, as in MRF smoothing. The resulting images obtained in hysteresis thresholding which overcomes the issue of the fragmented ISG foreground are displayed in Figure 6.9 together with the standard ISG segmentation result and the groundtruth. It can be seen that the foreground object is detected in fragments in the base model, but hysteresis thresholding combines the closer fragments into bigger single foreground objects.

Hysteresis thresholding increases the recall rate of the segmentation and keeps the precision stable. Since we apply hysteresis thresholding on background pixels inside the hysteresis region to detect missed foreground parts, it is expected that the recall will increase. The only conversion with the hysteresis thresholding is to relabel a pixel from background to foreground, when the pixel is between foreground objects and has a high probability of being foreground. Therefore, recall rate increases. Precision remains the same compared to the ISG model, since the number of pixels relabeled from background to foreground correctly is almost the same as those which are relabeled from foreground to background.

It is also seen that incorporating edge information slightly increases the performance. Since in terms of processing speed, using edge information does not cause any problems, this is acceptable. Hysteresis thresholding method combined with the edge contour enhancements operate at a processing rate of 24 frames per second. This method as well can be used for a real time application.

Precision and recall rates are given in Table 6.4 for the hysteresis thresholding method.

	Hysteresis Model Performance							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	95.4	45.1	57.9	60.7	91	56.8	93.7	65
Recall(%)	92.5	63	64.5	75.3	85	86	85.2	88.6
F1 Score	0.93	0.52	0.61	0.67	0.88	0.68	0.89	0.75

Table 6.4: Results - Hysteresis Segmentation

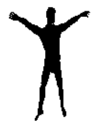
6.4 Experimental Results for MRF

MRF Belief Propagation optimization is applied to all test videos that are used for the base method. Table 6.5 displays precision and recall parameters obtained with introducing a smoothness penalty for the labels using the MRF approach and MRF probability thresholding.

It can be seen in the results for MRF smoothing that considering spatial correlations significantly increases both parameters in most cases as expected. False Alarms, which are usually pixels detected as foreground in small clusters surrounded by big clusters of background by the ISG model, will be smoothened, since MRF will penalize such a

–	MRF Model Performance							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	97.2	38	57.5	62.3	97.2	59.9	94.1	71
Recall(%)	94	61.4	60.9	79.6	80.7	89.1	87.6	92
F1 Score	0.95	0.47	0.59	0.69	0.87	0.71	0.90	0.80

Table 6.5: Results - MRF Segmentation


(a) Groundtruth

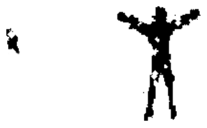

(b) MRF Segmented

Figure 6.10: MRF Foreground Detection

labeling. Thus, false alarms will be labeled as background to decrease the overall image energy. The spatial dependencies affect the foreground labeling decisions. MRF output increases the number of detected foreground pixels inside the foreground object where most of the pixels are foreground. It may also result in marking the pixels as background where there are already some number of pixels detected as background. Since we use 4-neighborhood system, the effect of this can easily be seen in smoothed pixels in a rectangular fashion. An example is shown in figure 6.10.

Pixels that would be detected as background inside foreground objects are also going to be classified as foreground, since MRF smoothness penalizes such a non-smooth labeling of pixels.

The only drawback of using MRF smoothing might be the significant decrease in frame rate. There are additional loops introduced due to the optimization algorithm. This causes a very significant decrease in the frame rate. Compared to the frame rate of the ISG model which is 38 FPS or the ISG probability thresholding that operates at 35 FPS, the MRF model based segmentation operates at a frame rate of 20 frames per second. However, this frame rate is still acceptable even for real time applications.

Another situation where MRF smoothing decreases the performance is when the data energies are very high and the smoothness is completely dominated by the incorrect data energies. When the data energy dictates that many pixels will be classified as foreground; but actually they are background; MRF tries to smooth the other background pixels around, thus, misclassifies them. In short, if the ISG model performs unacceptably bad, then MRF smoothing might make the segmentation even worse.

6.5 Experimental Results for Tracking Compensated Method

Within all proposed methods in this thesis, on the average, the best performing method is the tracking compensated segmentation where the decision-tree based tracker probabilities are combined with MRF probabilities. Tracking compensated method brings a very significant increase in performance.

In most cases in the videos, there are many outliers detected as foreground. Spatial smoothing like MRF and hysteresis can prevent this up to a point. That limit is defined by their smoothness energy weight or the probability thresholds. In tracking based method, ISG/MRF model is informed about stationary foreground objects. The probability of being foreground for pixels that lie outside the search window is 0, from the trackers perspective. This way, all outliers which are very far away from the foreground objects are eliminated directly. Inside the search window, the tracker is able to assign target probabilities to each pixel, based on the decision-tree classifier operating on RGB colors. Notice that for this tracker compensated method, the location of the foreground object must be provided manually on the first frame it appears. As an example, Figure 6.11 shows a video frame where tracker is used to track the object location and movement (red rectangle is the object window) and any pixel detected as foreground outside the search window (double size of the red object window) is eliminated and reclassified as background. In this example, a very low threshold is used in the ISG model to emphasize the effect of using the tracker compensation.

In cases where foreground is very stationary, ISG, MRF and hysteresis thresholding methods will definitely fail, since the data energy component will be very high and smoothing will cause the number or errors to increase. In tracking compensated method where the probabilities obtained by the tracker are averaged by the probabilities obtained by the ISG or MRF model, when the foreground object is stationary to blend into the background; then the significance of the tracker probability which is based only on color increases. This prevents the ISG or the MRF model to misclassify stationary foreground as background.

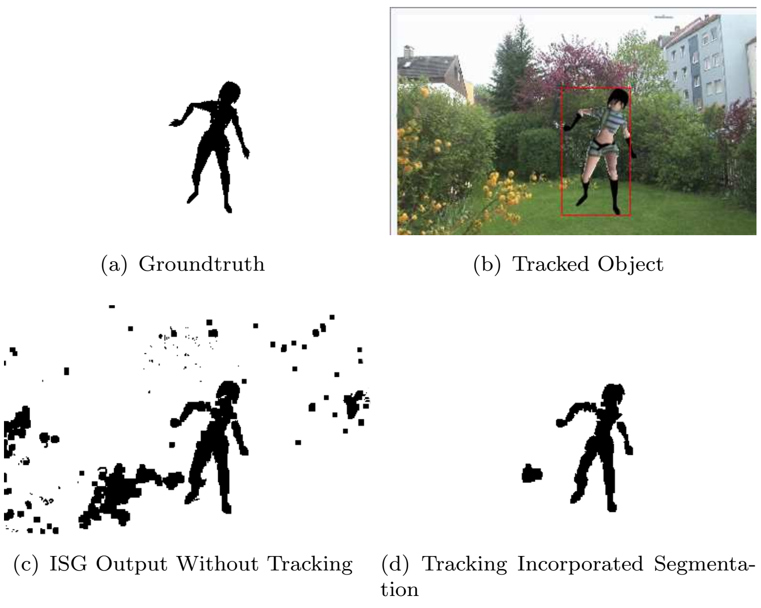

(a) Groundtruth (b) Tracked Object
(c) ISG Output Without Tracking (d) Tracking Incorporated Segmentation

FIGURE 6.11: Tracker Compensated Segmentation

Table 6.6 shows the results when we only apply outlier elimination using the search window of the tracker on the ISG segmentation result. In videos where there is a lot of background movement, there are also many outliers of misdetected pixels in the image, like in some of the test videos. Then, the outlier elimination method increases the performance significantly. The difference for this method is, inside the search window, it is only the ISG model classification that we rely on. There is no probability assumptions here, only a bare outlier elimination by the tracker is performed on top the ISG result.

–	Tracker Based Model Performance							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	97.6	63.3	68.3	59.1	93	60.5	91.1	66.5
Recall(%)	95.2	66.2	64.1	85.2	82.3	91	88.9	93.6
F1 Score	0.96	0.64	0.66	0.69	0.87	0.72	0.90	0.77

TABLE 6.6: Results - ISG and Tracker Outlier Elm.

In Table 6.7, the second option in tracker compensated method where only the foreground probabilities calculated by the decision-tree based tracker are used and directly thresholded. Since this tracker might be problematic when there are feature similarities between foreground and background, the precision decreases very significantly.

Upper half of Table 6.8 shows the performance results of combining the MRF probability image $\boldsymbol{V}_M$ and the decision tracker probability image $\boldsymbol{V}_T$. Since everything outside the search window is directly eliminated, this brings an expected increase in precision. It

–	Tracker Based Model Performance							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	86.7	51	57.9	59	71.6	58	91.7	53.3
Recall(%)	81.8	56.1	59.5	71.1	70.8	88.5	77.7	74.2
F1 Score	0.84	0.53	0.58	0.64	0.71	0.70	0.84	0.62

Table 6.7: Results - Only Tracker Probabilities Used

will not affect the recall rate since the search window is much larger than the object window. This guarantees that no foreground pixels are going to remain outside the search window. The lower half of Table 6.8 shows the results of the combined model, but not using the MRF probabilities, instead using the ISG probability image $\boldsymbol{V}_I$. There are no significant differences between these two options and they both result in very good performance when combined with tracking. However, MRF and the tracker combination performs slightly better.

–	MRF and Tracker							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	98.5	69.1	76.3	75.9	96.4	65.7	95.2	78.2
Recall(%)	96	73.7	72.4	88.1	89.9	95.1	94.6	97.7
F1 Score	0.97	0.71	0.75	0.81	0.93	0.77	0.94	0.86
-	ISG and Tracker							
Parameters	Video1	Video2	Video3	Video4	Video5	Video6	Video7	Video8
Precision(%)	97.6	66.2	70	62.1	96.2	61.3	93.8	70.5
Recall(%)	95.2	69.9	65.5	89.2	89.3	94.7	94	96.6
F1 Score	0.96	0.68	0.67	0.73	0.92	0.74	0.93	0.81

Table 6.8: Results - MRF/ISG and Tracker Combined

6.6 Overall Comparison of Segmentation Methods

Average performance parameter results for each method explained in the section above can be seen in Table 6.9. For this comparison table, simply an average for each method's results on every test video is calculated.

Avg.Parameters	ISG	Hysteresis	MRF	Outlier Elim. on ISG	Tracking+MRF
Precision	71.6	70.7	72.3	75.3	82.2
Recall	75.6	80	80.6	83.4	88.4
F1 Score	73.5	75	.76.2	79.1	85.2

Table 6.9: Results - Overall Comparison Table

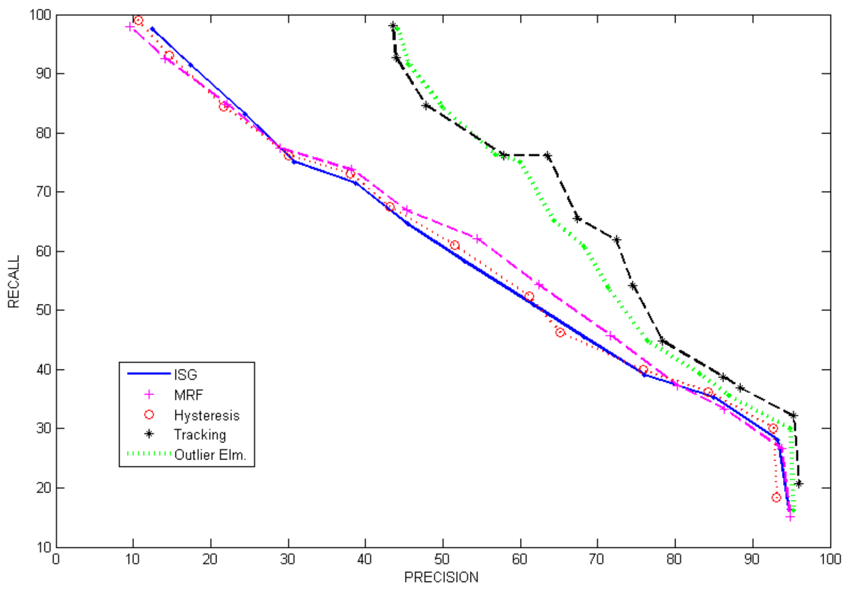


FIGURE 6.12: Precision-Recall Curve Comparison

Figure 6.12 shows the group of precision-recall curves of 5 different methods operating on the same video, video 6. To display a clear precision-recall curve, implementation parameters except probability and weight thresholds τ, that are given in the first section are fixed to contant values which are appropriate for a smooth precision-recall curve for almost all the videos. Each curve belongs to a specific method. These curves are formed by only modifying different threshold parameters, τ. Each point represents a precision-recall pair that corresponds to a specific threshold value. In each of these methods, τ_p, τ_w and τ_r are fixed. τ_r is always taken to be 3/7 of the parameters τ_w and τ_p, which are both taken to be the same for these curves. The area under a curve actually is directly proportional to how successful a segmentation method is.

On the figure itself, the blue curve shows the precision and recall relation for the regular ISG model.

Circle marked red points connected by red dashed lines form the precision-recall curve for the hysteresis thresholding. It is seen that for the same precision values, hysteresis thresholding increases the recall rate. In a general view, hysteresis line stays very close to the ISG line as expected, since it relies on the ISG detection and tries to improve the recall on that result. When the ISG model precision performance is very low, almost every pixel is detected as foreground. Therefore, filling the gaps between pixels make hysteresis thresholdind perform slightly worse in these cases.

Magenta colored plus signs and the magenta colored dashed line connecting them shows the curve for the MRF model. It can be observed that when the threshold values is very low, like in hysteresis thresholding, many of the pixels are assumed to be foreground and marked as foreground. MRF smoothing in this case favors the foreground too, and the model is not able to bring visible efficiency. The model performs even worse than hysteresis thresholding in these cases, since hysteresis thresholding uses a secondary criterion to fill in the gaps between foreground segments but the MRF model relies on the dominant data energies. When the threshold value begins to increase and the base ISG model is able to detect in an acceptable success, then the MRF increases the accuracy of the segmentation. This is the reason why for lower thresholds, sometimes the precision decreases compared to other methods. When ISG performance is acceptable, MRF performs better than the ISG model and the hysteresis thresholding method.

The green colored and dashed line shows the curve for tracker based outlier elimination with the ISG model. In this method, everything outside the search region window is neglected and directly considered as background. For this reason, even when the recall is very high, since all outliers will be eliminated, the precision value does not drop below 40 percent. An increased rate of misdetections in the overall image does not impact the method, since all these outliers are directly eliminated. For the pixels inside the search window, ISG decisions are used and this method completely relies on the ISG result. That is to say, inside the search window, precision and recall rates change as expected. When the recall rate of the ISG model drops to very low values, outlier elimination method approximates to the ISG model itself, since most of the pixels inside the search window will be detected as background due to the higher thresholding. For this reason, precision will begin to decrease.

The black colored stars connected via the black dashed line displays the best method according to our tests. In this method, the probability image obtained via MRF, which is $\boldsymbol{V}_M$, is averaged with the classifier based tracker's foreground probability image $\boldsymbol{V}_T$. Averaging parameter τ_o slightly favors the ISG model initially. On the other hand, unlike MRF smoothness penalty, this parameter is dynamic. When the foreground object remains stationary for some time, the weight of $\boldsymbol{V}_T$ increases significantly; hence correcting any possible misclassification based on $\boldsymbol{V}_M$. It can be seen that the area under the black curve is higher than those for all other curves. As expected, when the recall rate decreases down to extremely low values, all curves tend to approach each other, however, due to the tracking information and the effect of the tracker -which is not related to the ISG at all-, the black curve never goes under the other curves.

One disadvantage of the tracking compensated method is that at the same time, the ISG model, all MRF loops and the tracker operates and utilizes the memory. However, in our test PC, the tracker compensated MRF method operates at 16 FPS per second, which shows that it can be used for real-time applications anyway.

6.7 Results for Tracking with ISG/MRF

This section shows the results for the tracker, that uses the weight image obtained by the ISG/MRF model. Probability images $\boldsymbol{V}_I$ and $\boldsymbol{V}_M$ are used as the weight image input for a tracker, and mean shift algorithm is utilized to track the object within a frame as explained in previous chapter.

For tracking comparison, the main performance indicator we use is Frame Detection Accuracy (FDA), and it is defined by the VACE program [33] as

$$\frac{\text{Overlap\%}}{\#\text{Decisions}}, \tag{6.4}$$

where "Overlap%" is the overlapping area percentage between the tracker's bounding box and the groundtruth box. "#Decisions" is the number of total tracking decisions. Table 6.10 shows the FDA results of 4 trackers: ISG based tracker where $\boldsymbol{V}_I$ is the input weight image, MRF based tracker where $\boldsymbol{V}_M$ is used as the weight image, a decision-tree based classifier tracker that we implemented and an average of the probability images of MRF tracker and the decision tree tracker. In this table, it can be seen that ISG and MRF based trackers perform better than a decision tree based classifier which only operates by analyzing target features inside the search region window. ISG and MRF models keep track of the spatial and temporal correlations, thus they provide a more robust weight image. As mentioned before, traditional or classifier based trackers can easily get stuck in the video on a background object which "looks like" a foreground object in terms of its features. In addition to these, a dynamic combination of the ISG/MRF based tracking with the classifier based tracker according to the movement rate of the target object provides the best results as it can be seen in Table 6.10.

–	ISG	MRF	Classifier	Combined
Video1	88.2	88.3	84	88.6
Video2	94	94.4	87.9	94.7
Video3	77.4	79.6	90.1	88.5
Video4	96.8	98	93.2	98.3

Table 6.10: Results - Tracker with MRF and ISG

In terms of processing speed, ISG based tracker operates at 20 FPS, MRF based tracker operates at 15 FPS, and the combination of the MRF tracker and the decision-tree based tracker operates at 14 FPS. Real time tracking can still be provided by each option, considering that the implementation code is not even optimized perfectly for this thesis. In terms of tracking performance, combination mostly increases the accuracy of ISG and MRF trackers in conditions where the target stands stationary for a long time in the video. ISG model in this case decreases target probabilities since those pixels tend to be more “background”, but color based tracker still keeps a high foreground probability for those pixels.

The next chapter contains the conclusion and describes the future plans for this thesis.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this paper, we introduced a different approach on ISG model. We obtained class probabilities using the model. This probability image for the foreground at any time is $\boldsymbol{V}_I$. The approach to create a probability image $\boldsymbol{V}$ using each method, brings multiple options to enhance the segmentation performance. MRF segmentation can be used by getting the data-fitting energy components from $\boldsymbol{V}_I$ and it creates another probability image $\boldsymbol{V}_M$, which can be used in direct segmentation. This way, spatial correlation between pixels can be incorporated into the segmentation. The assumption of independent pixels in the ISG model is dealt with. It outperforms ISG based segmentation.

Spatial relations between pixels is also handled with the hysteresis thresholding mechanism. This method significantly removes fragmentation of the detected foreground in the ISG output, since it tries to fill in the gaps between foreground segments according to the foreground probabilities.

Besides spatial dependencies, the location and movement information about the foreground is alo utilized with the use of the mean-shift tracker. Decision-tree classifier based tracker runs in parallel with the ISG/MRF methods to provide location and movement information about the foreground object. Depending on this information, first of all, outliers are eliminated. In addition, the probability of having foreground inside the tracked object location is boosted especially in scenarios where foreground objects are very stationary. This completely prevents foreground blending into background undesirably in ISG and MRF models.

Besides all these enhancements for the segmentation, a side argument has been proven: $\boldsymbol{V}_i$ obtained via the ISG model or the $\boldsymbol{V}_M$ obtained via the MRF model can be well used as input weight images for a robust mean-shift tracker. Since mean-shift needs a probability image, and we introduce a way obtaining a robust probability image using the ISG and the MRF models; hence mean shift can be used in accordance with the ISG/MRF models for tracking purposes.

7.2 Future Work

For future improvements on our model, the main focuses are:

- The MRF optimization techniques ICM and BP are sub-optimal. Other techniques like Graph-cuts can be tested.
- Implementation can be optimized to achieve better real-time processing performance.
- Shadow removal can be applied especially on outdoor scenes. Shadow impacts especially the precision of the methods for outdoor scenes.
- Since we are using RGB color values, our models are intolerant to shadows or highlights. Modeling based not on RGB colors only but incorporating other features using longer feature vectors can be used and experimented with. These can be the features like texture based features, gradient based additional features, local binary pattern (LBP) and Gabor features, block features of the image and other color domain features like YCrCb or HSV, etc.

Bibliography

[1] C. Pornpanomchai, F. Stheitsthienchai, and S. Rattanachuen. Object detection and counting system. In *Image and Signal Processing, 2008. CISP '08. Congress on*, volume 2, pages 61–65, May 2008.

[2] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. In *Thirteenth Conf. on Uncertainty in Artificial Intelligence*, pages 175–181, 1997.

[3] P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection, September 2001.

[4] H. Han, Z. Wang, and J. et al. Liu. Adaptive background modeling with shadow suppression. In *Proc. of Intelligent Transportation Systems*, pages 720–724, 2003.

[5] D.M. Russell and S.G. Gong. A highly efficient block-based dynamic background model. In *AVSBS05*, pages 417–422, 2005.

[6] D. Turdu and H. Erdogan. Improved post-processing for gmm based adaptive background modeling. In *Int. Symposium on Computer and information sciences, ISCIS 2007*, pages 1–6, November 2007.

[7] O. Munkelt C. Ridder and H. Kirchner. Adaptive background estimation and foreground detection using kalman filtering. In *in Proc. ICAM*, pages 193–199, 1995.

[8] C. R. Wren, A. Azarbayejani, T. Darrel, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785, July 1997.

[9] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 2:252, August 2002.

[10] S. Calderara, R. Melli, A. Prati, and R. Cucchiara. Reliable background suppression for complex scenes. In *VSSN06: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 211–214. ACM, 2006.

[11] Patrick Perez. Markov random fields and images, 1998.

[12] Geman S. and Geman D. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *Journal of Applied Statistics*, 20(5):25–62, 1993.

[13] S. A. Barker and Peter J. W. Rayner. Unsupervised image segmentation using markov random field models. In *EMMCVPR '97: Proceedings of the First International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 165–178, London, UK, 1997. Springer-Verlag.

[14] C. Benedek and T. Sziranyi. Markovian framework for foreground-background-shadow separation of real world video scenes. In *ACCV06*, pages 898–907, 2006.

[15] Ross Kindermann and J. Laurie Snell. *Markov Random Fields and Their Applications.* AMS, 1980.

[16] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, B-48:259–302, 1986.

[17] Huawu Deng and David A. Clausi. Unsupervised image segmentation using a simple mrf model with a new implementation scheme. *Pattern Recognition, International Conference on*, 2:691–694, 2004.

[18] Judea Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *Association for the Advancement of Artificial Intelligence National Conference*, pages 133–136, 1982.

[19] Pedro Felzenszwalb, , and Daniel P. Huttenlocher. Efficient belief propagation for early vision. In *In CVPR*, pages 261–268, 2004.

[20] O. Veksler. Graph cut based optimization for mrfs with truncated convex priors. volume 2, pages 1–8, 2007.

[21] A. Shabou, F. Tupin, and J. Darbon. A graph-cut based algorithm for approximate MRF optimization. 2009.

[22] Shyjan Mahamud. Comparing belief propagation and graph cuts for novelty detection. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1154–1159. IEEE Computer Society, 2006.

[23] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40, January 2003.

[24] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, 2:142–149, 2000.

[25] Dorin Comaniciu and Peter Meer. Mean shift: a robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619, August 2002.

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

[27] S. Avidan. Ensemble tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(2):261–271, 2007.

[28] Gary R. Bradski. Computer vision face tracking for use in a perceptual user interface, 1998.

[29] http://imagelab.ing.unimore.it/vssn06/. Vssn2006 4th acm int. workshop on video surveillance and sensor networks, 2006.

[30] Rafael C. Gonzalez. *Digital image processing using MATLAB.* Prentice Hall, N.J., 2004.

[31] Y. Cheng. Mean shift, mode seeking, and clustering. *Pattern Anal. and Machine Intell., IEEE Transactions on*, (17):790–799, 1995.

[32] http://www.cvg.rdg.ac.uk/PETS2001/. 2nd ieee int. workshop on performance evaluation of tracking and surveillance, 2001.

[33] R. Kasturi, D. Goldgof, and P. et al Soundararajan. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):319–336, 2009.