

國立交通大學
資訊科學與工程研究所

博士論文

雙階層視訊分析 — 由靜態背景模型到動態前景切割

Bi-Layer Video Analysis – from Static
Background Modeling to Dynamic
Foreground Segmentation

研究生：林泓宏

指導教授：莊仁輝 博士

劉庭祿 博士

中華民國一百年三月



雙階層視訊分析 — 由靜態背景模型到動態前景切割

Bi-Layer Video Analysis – from Static Background
Modeling to Dynamic Foreground Segmentation

研究生：林泓宏

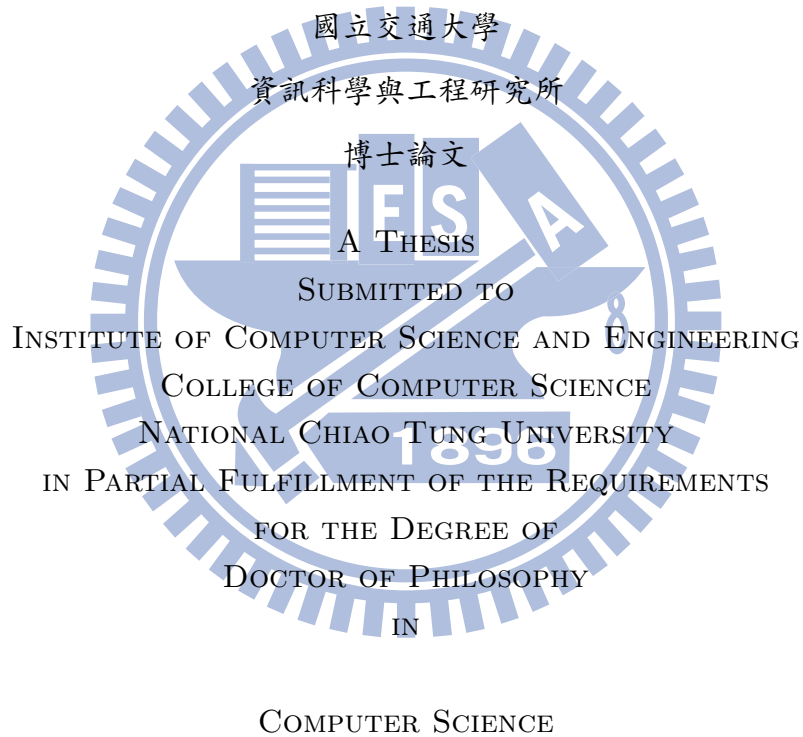
Student: Horng-Horng Lin

指導教授：莊仁輝 博士

Advisor: Dr. Jen-Hui Chuang

劉庭祿 博士

Dr. Tyng-Luh Liu



March 2011

HsinChu, Taiwan, Republic of China

中華民國一百年三月



謹獻給 我的父母 —
林良應先生與陳玉樺女士





雙階層視訊分析 — 由靜態背景模型到動態前景切割

學生：林泓宏

指導教授：莊仁輝 博士

劉庭祿 博士

國立交通大學 資訊科學與工程研究所

摘要

雙階層視訊分割 — 即對視訊影片作前景層與背景層的區域切割 — 是電腦視覺領域中一個極具挑戰性的問題，蓋因視訊內容的變化多樣，使得前景與背景的階層分割變得複雜。對於此一問題，我們在論文中分別以「背景模型初始化」、「背景模型維護」與「視訊階層遞移」三個研究主題來進行探討；其中，前兩個研究主題，是針對固定式攝影機所拍攝的影片，作靜態背景模型的建構與維護，使得前景階層，可透過與背景相減分割出來，而第三個研究主題，則是針對移動式攝影機所拍攝的動態影片，作前景與背景階層的遞移分割。

在背景模型初始化的研究主題探討中，我們開發一個以影像區塊為基礎的快速背景模型估計法，並提出新穎的背景模型完整度量測法則，使得一個完整的初始背景模型，可被快速建構出來。在背景模型維護的研究主題探討中，我們檢視了常用的高斯混合模型，發現在高斯混合模型中，需要兩種型態的學習速率控制，方可有效地平衡背景變化容忍度與前景偵測敏感度兩項拮抗因素，對此，我們提出一個基於高階資訊回饋的新式學習速率控制法，來改良高斯混合模型之背景模型維護方式。在視訊階層遞移的研究主題探討中，我們提出一

個基於半監督式頻譜叢集法的動態階層分割架構，來對於移動式攝影機所拍攝的視訊影像，逐張作前景與背景的階層分割；其中，我們進一步擴充了半監督式頻譜叢集法的數學模型，以便在視訊階層分割過程中，調控階層標籤估計的可靠度，以增進階層分割的準確性，實驗結果顯示，此一視訊階層遞移分割架構，對動態影片的切割具有良好的效果。

關鍵字：雙階層視訊分割、背景模型、高斯混合模型、半監督式頻譜叢集。



Bi-Layer Video Analysis – from Static Background Modeling to Dynamic Foreground Segmentation

Student: Horng-Horng Lin Advisor: Dr. Jen-Hui Chuang

Dr. Tyng-Luh Liu

Institute of Computer Science and Engineering

National Chiao Tung University

The watermark is a circular seal of National Chiao Tung University. It features a gear-like outer border, a central shield with a book and a scale, and the letters 'ES' and 'A' on either side. The year '1896' is at the bottom. The word 'ABSTRACT' is overlaid on the seal.

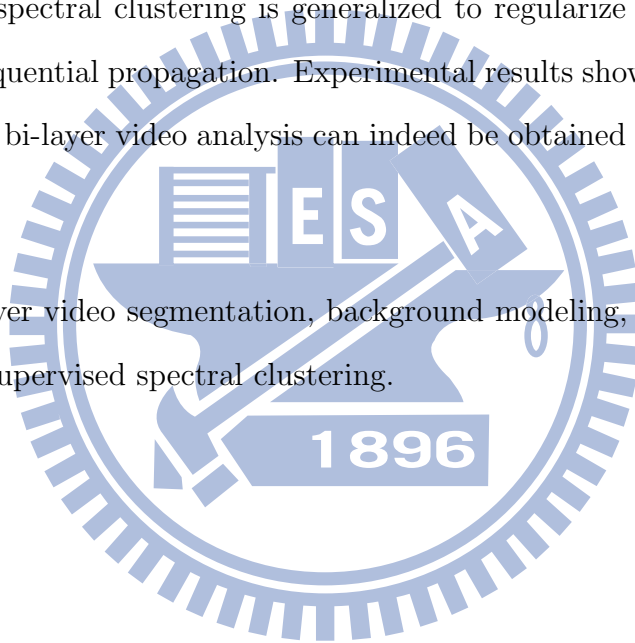
ABSTRACT

Bi-layer video segmentation, i.e., the extraction of foreground regions from background ones for a video sequence, is a challenging research field in computer vision due to large content variation among video frames. To better address this bi-layer video segmentation problem, three research topics are investigated in this thesis including background model initialization, background model maintenance, and video layer propagation. While the first two topics concern static background modeling for analyzing videos obtained from static cameras, the third one pertains to dynamic foreground segmentation for videos captured by moving cameras.

For the problem of background model initialization, we propose an efficient background model estimation scheme based on image block classification, and develop novel criteria for measuring the completeness of a background model. For the

problem of background model maintenance, we look into the formulations of Gaussian mixture modeling (GMM) and identify the needs of two types of learning rates for GMM to effectively deal with a trade-off between robustness to background changes and sensitivity to foreground abnormalities. A novel bivariate learning rate control scheme for GMM based on a feedback of high-level information is also proposed. For the problem of video layer propagation, a new framework based on semi-supervised spectral clustering is proposed for dynamic foreground segmentation of a video shot captured by a moving camera. The adopted formulation of semi-supervised spectral clustering is generalized to regularize the reliabilities of layer labels in sequential propagation. Experimental results show that satisfactory results of related bi-layer video analysis can indeed be obtained with the proposed approaches.

Keywords: Bi-layer video segmentation, background modeling, Gaussian mixture modeling, semi-supervised spectral clustering.



誌謝

非常幸運地，我能得到兩位指導教授的教導。在博士修業的前期，我同時也在中研院資訊所服國防役，是資訊所的劉庭祿老師，帶領我進入電腦視覺與機器學習的研究領域，教導我基礎知識，告訴我研究的目標與紀律，協助我建立研究常規，更對我在研究與工作上有許多包容；同時，在本科領域外，劉老師也帶領我初窺生物資訊領域的堂奧，回想起種種情境，無不讓我深深感念於心，劉老師的教導，對我的博士修業及日後的工作與研究，有著至為深遠的影響。國防役期滿後，回到交通大學成為全職學生，交大資工的莊仁輝老師，則是給予學生獨立研究的訓練，讓我在研究主題選擇上有很大的自由度，並逐步引導我作研究推展、論文撰改、投稿與答辯等，帶領我走向獨立研究之路；同時，莊老師自我碩士修業以來，逾十年的指導中，除了專業領域外，更涵蓋了我生活與家庭層面，提供我許多深具人生智慧的建議，使得我的研究與人生路程得以平順，長期以來，莊老師對我所付出的諸多心力，實無可計量。而今博士修業將告一段落，我深覺我是何等幸運，能夠同時得到兩位對學生至誠至性的師長，對我長年的指導，沒有他們，我不會走向研究之路，也無從得見研究之樂，更遑論學位的完成；對兩位老師的感激，遠非言語所能形容！

指導教授以外，我也特別感謝交大資工的蔡文祥老師，在我執行研究計畫時，所給予的指導與勉勵；以及中研院資訊所的陳祝嵩老師對我的關懷與幫助。此外，我也非常感激在中研院資訊所時陳煥宗、林彥宇、張天龍、趙盈

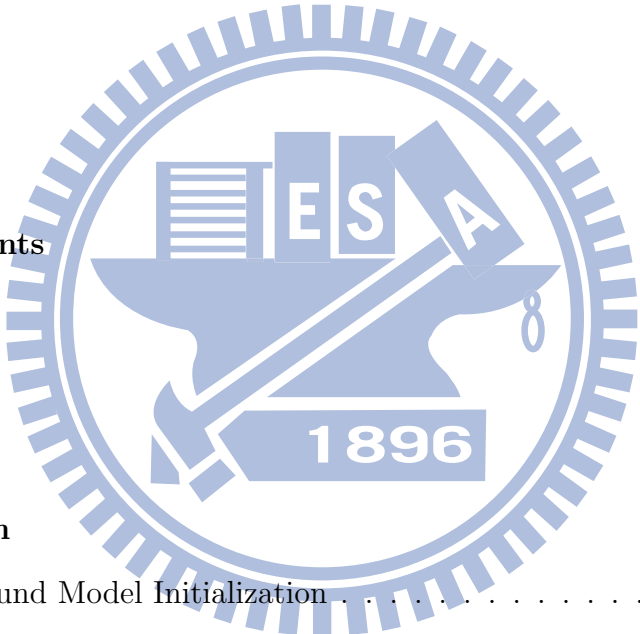
勝、陳俊宏、蔡玉寶、張文彥、葉士良等同事的研究討論、建議與鼓勵，以及在交大時高肇宏、吳至仁、羅國華、林哲寬、陳宇欣、吳思慧、邱郁婷、陳光兆、劉怡伶、李宗穎、蔡易達、吳佳昱等同學對我研究上的協助，是他們，豐富了我的研究歷程。同時，我也衷心感謝威聯通科技的黃哲文經理與張明智總經理，是他們的大力支持，我才有機會將公司工作與論文研究結合，進一步整理成研究成果。而在論文口試時，口試委員王才沛老師、王聖智老師、洪一平老師、范國清老師、廖弘源老師、賴尚宏老師、蔡文祥老師所給予的寶貴建議，也將作為我日後研究工作的指引。

在此同時，我也要對我的父母獻上最深的謝意與敬意，沒有他們無私的付出與栽培，我將無所憑、無所立。同時我也要謝謝內人意屏這些年來的支持，讓我可以投入研究工作、完成學業。最後，謝謝學習歷程中許多協助過我的朋友，謹致上我由衷地感激。



Table of Contents

摘要	i
Abstract	iii
誌謝	v
Table of Contents	vi
List of Figures	xi
List of Tables	xii
1 Introduction	1
1.1 Background Model Initialization	2
1.2 Background Model Maintenance	2
1.3 Video Layer Propagation	4
1.4 Thesis Organization	5
2 Background Model Initialization via Classification	7
2.1 Overview	8
2.1.1 Related Work	10



2.2	Background Model Estimation via Classification	14
2.2.1	Iterative Estimation Scheme	14
2.2.2	The Detailed Algorithm	18
2.3	Fast Classification with Soft Margins	21
2.3.1	Feature Selection	22
2.3.2	SVMs with Probability Outputs	23
2.3.3	CGBoost with Probability Outputs	24
2.4	Experimental Results	25
2.4.1	Classifier Training	26
2.4.2	Performance Evaluation	29
3	Background Model Maintenance via Density Estimation	41
3.1	Overview	42
3.1.1	Related Work	44
3.1.2	Model Accuracy, Robustness and Sensitivity	46
3.2	Bivariate Learning Rate Control via High-Level Feedback	48
3.2.1	Background Model Maintenance	49
3.2.2	Feedback Control	53
3.2.3	Heuristic for Adaptation of Double-Quick Lighting Change	60
3.3	Experimental Results	62
3.3.1	Regularized Background Adaptation	63
3.3.2	Parameter Tuning	67
3.3.3	Double-Quick Lighting Change	69
3.3.4	Quantitative Evaluations	71
3.3.5	Scene Change	73

3.3.6	Other Scenarios	74
4	Video Layer Propagation via Semi-Supervised Clustering	77
4.1	Overview	78
4.1.1	Related Work	80
4.2	Video Layer Propagation Framework	81
4.2.1	Block Label Inference	81
4.2.2	Semi-Supervised Spectral Clustering	84
4.2.3	Algorithm Interpretation	89
4.3	Algorithm Implementation	91
4.3.1	Similarity Measure	91
4.3.2	Local Clustering	92
4.3.3	Propagation Band	95
4.3.4	Regularization of Label Reliability	95
4.3.5	Optional User Intervention	99
4.4	Experimental Results	99
4.4.1	Video Layer Propagation in Static Background	101
4.4.2	Video Layer Propagation in Moving Background	103
5	Conclusion	107
5.1	Summary of Static Background Model Initialization	108
5.2	Summary of Static Background Model Maintenance	109
5.3	Summary of Dynamic Video Layer Propagation	110
	Appendix	113
	References	119



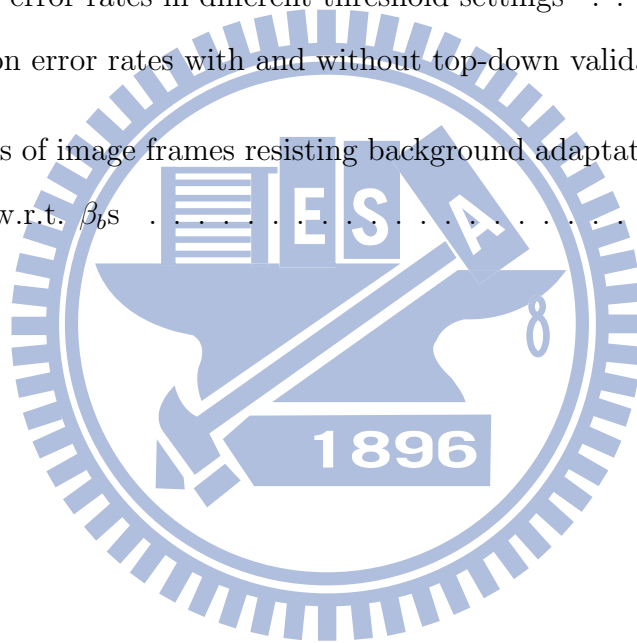
List of Figures

2.1	The general idea of background model initialization	8
2.2	Notations for background model initialization	13
2.3	Flowchart of the bottom-up and top-down processes	16
2.4	Training images for background model initialization	26
2.5	Distributions of training data and training results	28
2.6	Results of background model initialization	30
2.7	Results of background block detection	31
2.8	Results of different parameter settings	34
2.9	Comparisons of [22], [54], and our approach	36
2.10	Tests on lighting variations	38
2.11	Background model initialization and tracking	40
3.1	Flowchart of a general-purposed surveillance system	54
3.2	Simulated changes of the learning rate η_t	57
3.3	Example of motion blur	58
3.4	Examples of quick and double-quick lighting changes	60
3.5	Comparisons of background adaptation to quick lighting changes	64

3.6	Comparisons of background modeling for missing object and waving hand	66
3.7	Comparisons of background modeling without and with using the background-type rate control	67
3.8	Comparisons of background adaption to double-quick lighting change	70
3.9	Snapshots of the ground-truth images	71
3.10	Quantitative comparisons of [36], [54], and our approaches	72
3.11	Comparisons of scene change adaptation	75
3.12	Foreground detection and background modeling results for other scenarios	76
4.1	Illustration of the spatio-temporal neighbors of a block	82
4.2	Examples of kernel construction	89
4.3	Examples of sub-graph \mathcal{G}'_i in different representations	92
4.4	Simulated example on the regularization of label reliability	96
4.5	Quantitative evaluations of the proposed regularization of label reliability	98
4.6	Results of the <i>IU</i> experiment	100
4.7	Results of the <i>IU</i> experiment with user interventions	101
4.8	Quantitative evaluations for the <i>IU</i> experiment	102
4.9	Snapshots of the <i>Mobile</i> sequence and its ground-truth layer masks	104
4.10	Results of the <i>Mobile</i> experiment	105
4.11	Quantitative assessment on regularization of label reliability	106

List of Tables

2.1	Comparisons between SVMs and CGBoost	29
2.2	Average error rates in different threshold settings	29
2.3	Detection error rates with and without top-down validation.	32
3.1	Numbers of image frames resisting background adaptation to after- images w.r.t. β_6 s	68





Chapter 1

Introduction

Bi-layer video segmentation, which involves the extractions of foreground regions from background ones for a video sequence, is a challenging research field in computer vision, due to large content variation among video frames. Understanding of video contents via computer-assisted analysis, which is one of the main goals of intelligent video analytics for surveillance applications and multimedia search, can be greatly benefited by stable and accurate video layer segmentation. In this thesis, the research problem of bi-layer video analysis for segmenting videos captured by static and moving cameras are investigated along three research directions: background model initialization, background model maintenance and video layer propagation. While the first two directions concern static background layer modeling for analyzing video sequences obtained from static cameras, the third one addresses dynamic foreground/background layer extraction for video sequences captured by moving cameras.

1.1 Background Model Initialization

For a video sequence captured by a static camera, its foreground objects can often be efficiently extracted via background subtraction if a background model for properly describing a static background scene is given. Despite the large amount of previous research works on background modeling, the initialization of a stable background model for a busy scene, such as a road junction with heavy traffic, has been less discussed. In our investigation of the problem of background model initialization, a new estimation scheme that combines bottom-up and top-down information to construct a stable and complete background model is presented in Chapter 2, wherein efficient image block classification for background model construction is proposed and novel criteria for the measurement of background model completeness is developed. Experimental results show that the efficient block-based processing, together with the effective model completeness measure, can derive stable background models for busy scenes and outperforms the compared approaches.

1.2 Background Model Maintenance

Once a proper background model for a scene of interest has been initialized, this model needs to be maintained thereafter to catch background changes, such as environmental lighting variations, so that foreground regions can be accurately differentiated. For background model maintenance, Gaussian mixture modeling (GMM) is a popular choice due to its capability of adaptation to periodic background variations. However, the effectiveness of GMM is often limited by a trade-off be-

tween statistical robustness to background changes and sensitivity to foreground abnormalities, and is inefficient in managing the trade-off for various surveillance scenarios. To solve this problem, a novel bivariate learning rate control scheme for GMM based on a feedback of high-level information is proposed in Chapter 3. Experimental results show that the proposed GMM approach is superior to the compared GMM-based methods in delivering better background model adaptation results for challenging scenarios with the aforementioned trade-off.

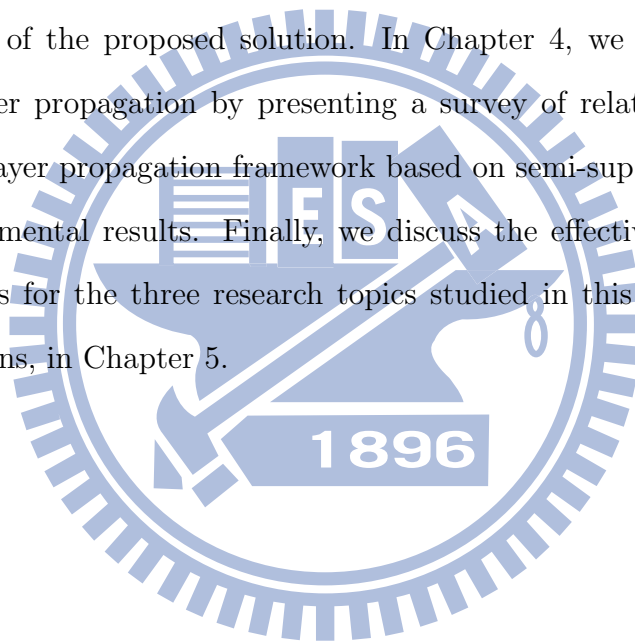
It is worth noting that two distinct approaches are applied in this thesis to solve the problems of background model initialization and maintenance. In general, the proposed GMM approach for solving the problem of background model maintenance can almost always give promising performance in background adaptation and foreground detection. However, according to our study, such an approach is not suitable for the problem of background model initialization, mainly due to its deficiency in evaluating whether a derived background model by GMM is stable and/or complete, as will be discussed in Chapter 2. On the other hand, while the proposed approach for background model initialization delivers a more stable background model than the GMM approach, it is less capable of capturing dynamic background changes, like waving trees, in long-term model maintenance, and may result in a stable but slightly blur background model. Therefore, we present a two-stage treatment to the general problem of static background modeling, with each stage adopting a different approach, which will best fit each specific need mentioned above.

1.3 Video Layer Propagation

For the case of moving camera, we investigate the problem of video layer propagation in Chapter 4, wherein foreground and background video layers of a video shot are segmented in a sequential manner for consecutive image frames. Assume that the bi-layer image segmentation for the first video frame of a video shot is given in advance. The goal of video layer propagation is to extract the corresponding video layer segments in subsequent video frames. Except for the initial layer information, no prior assumptions or restrictions, e.g., on foreground shapes, on background models, or with respect to camera motions, are made. This general setting brings a big challenge in problem solving because, for example, a background layer may be cluttered and may undergo large changes in a video shot due to camera movements. To extract time-varying video layer segments, a new framework based on semi-supervised clustering is developed. Under this framework, image blocks are used as layer propagation units to avoid costly pre-segmentation of images into super-pixels. By modeling video layer propagation between consecutive image frames as a label inference problem wherein new block labels are inferred from previously known ones, and by solving this problem via semi-supervised spectral clustering, video layers are progressively propagated. Experimental results show that the proposed video layer propagation method can effectively extract dynamic video layers, even in large, non-rigid motions.

1.4 Thesis Organization

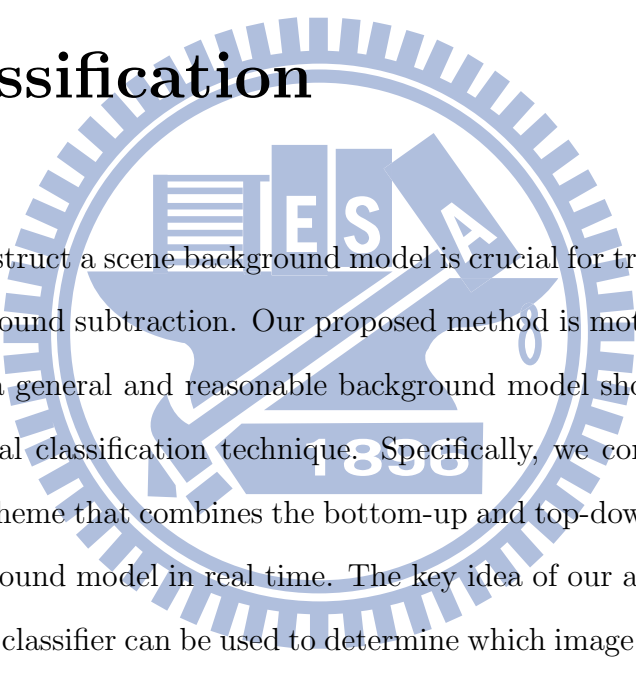
The rest of this thesis is organized as follows. In Chapter 2, we investigate the problem of background model initialization by presenting an overview of background modeling techniques, the proposed background model initialization approach and experimental results. Assuming that an initial background model is obtained, we discuss the problem of background model maintenance in Chapter 3 by addressing the trade-off between model robustness and sensitivity, giving a GMM-based solution for balancing the trade-off, and presenting experimental results to support the effectiveness of the proposed solution. In Chapter 4, we explore the problem of video layer propagation by presenting a survey of related literature, the proposed video layer propagation framework based on semi-supervised clustering, and some experimental results. Finally, we discuss the effectiveness of the proposed approaches for the three research topics studied in this thesis, as well as future explorations, in Chapter 5.





Chapter 2

Background Model Initialization via Classification



To efficiently construct a scene background model is crucial for tracking techniques relying on background subtraction. Our proposed method is motivated by criteria leading to what a general and reasonable background model should be, and realized by a practical classification technique. Specifically, we consider a two-level approximation scheme that combines the bottom-up and top-down information for deriving a background model in real time. The key idea of our approach is simple but effective: If a classifier can be used to determine which image blocks are part of the background, its outcomes can help to carry out appropriate block-wise updates in learning such a model. The quality of the solution is further improved by global validations of the local updates to maintain the inter-block consistency. A complete background model can then be obtained based on a measurement of model completion. To demonstrate the effectiveness of our method, various experimental results and comparisons are included.

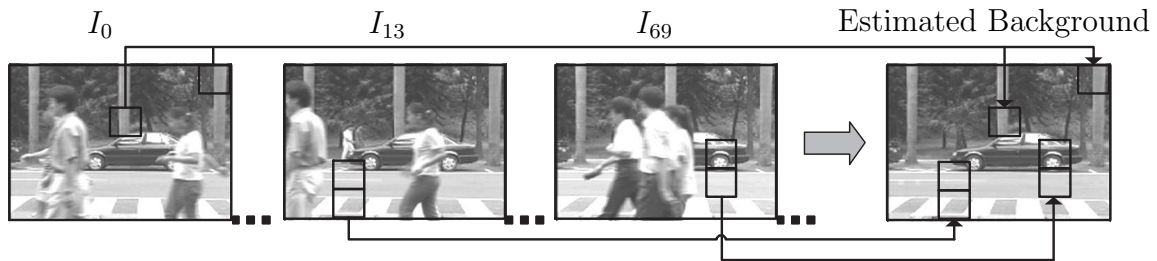


Figure 2.1: The general idea of background model initialization. Through performing on-line classifications and by iteratively integrating the frame-wise detected background blocks of images captured with a static monocular camera, the scene background can be reliably estimated in real time.

2.1 Overview

Visual tracking systems using background subtraction often work by comparing the upcoming image frame with an estimated *background model* to differentiate moving foreground objects from the scene background. Hence the performance of such systems depends heavily on how the background information is modeled initially, and maintained thereafter. In this work, we aim to establish a learning approach to reliably estimate a background model even when substantial object movements are present during the initialization stage. As illustrated in Fig. 2.1, the overall idea is to efficiently *identify* background blocks from each image frame through on-line classifications, and to iteratively *integrate* these background blocks into a complete model so that a tracking process can be automatically initiated in real time. In developing such a progressive processing scheme for initializing a background model, some criteria are considered.

- *Stationary scene adaptation*: It is commonly agreed that stationary scenes are considered as background. Thus, in our design, when a moving object becomes stationary over a certain period of time, it will be incorporated

into a background model. This would yield an initial background model accommodating the most recent statistics about the background scene, e.g., a parking car or an occluded area.

- *Gradual variation adaptation:* The computation of a background model should take account of small variations caused by, for example, gradual illumination changes, waving trees, and faint shadows. It allows a system to reduce the false detection rate of foreground objects.
- *Model completion:* Depending on object movements, the number of image frames needed in estimating an initial background could vary significantly. Hence, a measurement for the availability of a background model has to be defined so that the system can immediately begin to track objects upon the completion of model initialization.
- *Efficiency:* A background model must give rise to efficient on-line derivations to guarantee real-time tracking performance.

The first two criteria listed above manifest what kind of scene contents are considered as background. The last two ones illustrate the design requirements of a background model initialization system: it should be capable of deriving a complete background model in a progressive manner and in real-time.

In the proposed approach, two features will be observed. First, we utilize learning methods to identify background blocks. Rather than developing discrimination rules or models, we adopt learning approaches to construct a background block classifier. This strategy not only provides a convenient way of defining some preferred background types from image examples, but also avoids complicated issues

of manually setting discriminating parameters, because they can be resolved by learning from the chosen data. Second, the derived background model fulfills the four criteria. To achieve efficiency, a progressive estimation scheme is developed and a fast classifier adopted. For the model completion criterion, an effective definition is given to indicate that a complete background model is obtained, and the subsequent tracking procedures can be started. Regarding the adaptation criteria, we implement a bottom-up block updating, in either a gradual or an abrupt fashion, for capturing the background variations and scene changes, respectively.

2.1.1 Related Work

Background modeling for tracking typically involves three issues: *representation*, *initialization*, and *maintenance*. For example, one could *represent* a scene background by assuming a single Gaussian distribution for each pixel, *initialize* the model by estimating from an image sequence, and *maintain* it during tracking by updating Gaussian parameters of the background pixels. While the emphases of most previous works, including those to be described later, are mainly on representation and maintenance, the task to compute an initial background model has been somewhat neglected or otherwise simplified by not allowing large object movements throughout the initialization process, e.g., [14], [41], [64].

Background representation and maintenance

Gaussian models are perhaps the most popular representation for modeling a scene background, e.g., [6], [36], [42], [54], [64]. Their maintenance is usually carried out in the form of temporal blending to update intensity means and variances. Thus

related researches often differ in the number of Gaussian distributions used for each pixel, and the update formulas for the Gaussian parameters. In [20], Gao et al. further investigate possible errors caused by Gaussian mixture models, and then apply statistical analysis to estimate related parameters.

Apart from Gaussian assumptions, Elgammal et al. [13] consider *kernel smoothing* for a non-parametric estimate of pixel intensity over time. In [58], Toyama et al. propose a *wallflower* algorithm to address the problem of background representation and maintenance in three levels: pixel, region, and frame levels. Ridder et al. [47] use a *Kalman-filter* estimator to identify the respective pixel intensities of foreground and background from an image sequence, and to suppress false foreground pixels caused by shadow borders. In [27], a mixture of local histograms is proposed to construct a texture-based background model that is more robust to background variations, e.g., illumination changes.

Prior assumptions about the foreground, background, and shadows can be used to simplify the modeling complexity. For vehicle tracking, Friedman and Russell [19] propose three kinds of color models to classify pixels into road, shadow, and vehicle. They employ an incremental EM to learn a *mixture-of-Gaussian* for distinguishing the foreground and background. In [48], [59], prior knowledge at pixel level is considered in learning the model parameters of the foreground and background. Then, a high-level process based on *Markov random field* is performed to integrate the information from all pixels.

Background model initialization

The most straightforward way to estimate a background model is to calculate the intensity mean of each pixel through an image sequence. Apparently, this is rarely

appropriate for practical uses. Haritaoglu et al. [23], instead, compute intensity medians over time. Yet a more general framework by Stauffer and Grimson [54] is to use pixel-wise *Gaussian mixtures* to model a scene background. Mittal and Huttenlocher [42] later extend the Gaussian mixture idea to construct a mosaic background model from images captured using a non-stationary camera. In [58], *bootstrapping* for background initialization is proposed, and implemented with a pixel-level *Wiener filtering*.

Among the above-mentioned approaches, initializing a background model is viewed more or less as part of the process for background maintenance. They do not have a systematic way to measure the quality, and determine the degree of completion for such a model. Consequently, these methods often require *simple* initializations, or otherwise start tracking activities with unreliable background models.

For computing an explicit background model, Gutches et al. [22] use optical flow information to choose the most likely time interval of stable intensity at each pixel. However, the quality of their derived background model depends critically on the accuracy of the pixel-wise optical flow estimations. Cucchiara et al. [9] represent a background model by pixel medians of image samples, and specifically identify moving objects, shadows and ghosts¹ for different model updates using color and motion cues. Based on the Gaussian mixture model, Hayman and Eklundh [26] formulate a statistical scheme to derive a mosaic background model with an active camera. They consider a *mixel distribution* to correct the errors in background registration. In [11], De la Torre and Black apply *principal component*

¹Ghosts are false foreground objects detected by subtracting an inaccurate background model from image frames.

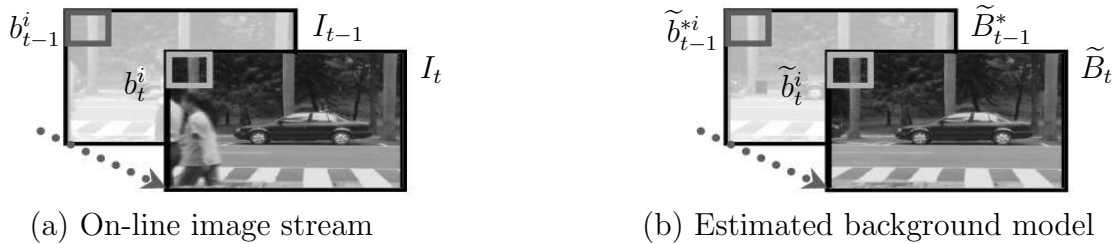


Figure 2.2: Notations for background model initialization. (a) I_t and I_{t-1} are the image frames at time t and $t-1$, and their i th blocks are denoted as b_t^i and b_{t-1}^i , respectively. (b) For the background models, \tilde{B}_t is a possible estimation at time t , while \tilde{B}_{t-1}^* is the best estimation up to time $t-1$. Accordingly, their i th blocks are represented by \tilde{b}_t^i and \tilde{b}_{t-1}^{*i} .

analysis (PCA) to construct the scene background from an image sequence. More recently, Monnet et al. [43] propose an *incremental PCA* to progressively estimate a background model and detect foreground changes. Still, these systems all lack an explicit criterion for determining whether a background initialization is completed or not—a crucial and practical element for a real-time tracking system.

Other techniques that explore layer decompositions of a video sequence can also be used to estimate a background model. Irani and Peleg [28] explore the decompositions of dominant motions and apply them to the construction of an unoccluded background image. In [29], [17], sprite layers are derived from probabilistic mixture models, in which cues of layer appearances and motions are encoded. In [1], [2], Aguiar and Moura consider rigid motions, intensity differences, and the region rigidity for figure-ground separation, and formulate them as a penalized likelihood model that can be optimized in efficient ways. In [7], [32], [63], graph-cut-based techniques, such as [5], are applied to decompose video layers via pixel labeling, with various objective functions being optimized. Though all the layer-based approaches are capable of deriving a background model even for dynamic scenes,

they often need to process a video sequence in batch, which is different from the proposed progressive scheme.

2.2 Background Model Estimation via Classification

Due to the restriction of limited memory space and the requirement of real-time performance, only a small number of recent image frames are stored and referred during the construction of a background model. Thus, an iterative estimation scheme is proposed in the following to progressively identify background blocks in image frames and to incorporate their information into a background model.

2.2.1 Iterative Estimation Scheme

To illustrate the idea of the proposed iterative estimation scheme, we begin by summarizing the notations and definitions adopted in our discussion.

- We denote the test image sequence up to time instant t as $\mathbf{I}_t = \{I_1, I_2, \dots, I_t\}$, and the most recent ℓ image frames as $\mathbf{I}_{t,\ell} = \{I_{t-\ell+1}, \dots, I_{t-1}, I_t\}$. We also use b_t^i to stand for the i th block of I_t , and $\mathbf{b}_{t,\ell}^i = \{b_{t-\ell+1}^i, \dots, b_{t-1}^i, b_t^i\}$ for the set of i th blocks from $\mathbf{I}_{t,\ell}$ (see Fig. 2.2).
- Let \tilde{B}_t be any possible background model estimation at time t , and \tilde{B}_{t-1}^* be the estimated background model at time $t-1$. Then, the i th blocks of \tilde{B}_t and \tilde{B}_{t-1}^* are denoted as \tilde{b}_t^i and \tilde{b}_{t-1}^{*i} , respectively.
- A training set of m samples, $\mathbf{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_m, y_m)\}$, is used to

build a binary classifier, where each \mathbf{x}_i is a fixed-size image block (or simply the extracted feature vector), and $y_i \in \{-1 \text{ (foreground)}, +1 \text{ (background)}\}$ is its label.

- With training data \mathbf{D} , an *optimal* classifier f^* can be defined as

$$f^* = \arg \max_f p(f | \mathbf{D}). \quad (2.1)$$

Equation (2.1) manifests that a classifier f^* can be derived from a probabilistic *maximum a posteriori* (MAP) treatment [49]. It is thus more desirable to have not only classification labels/scores but also probabilistic outputs of f^* . In Sec. 2.3 we will explain that either an SVM or a boosting-with-soft-margins classifier is appropriate for delivering such probabilities. With probabilistic outputs, a threshold can then be set to adjust the classification boundary, which is useful for our background estimation. We will demonstrate this usage in Sec. 2.4.1.

The proposed iterative estimation scheme for deriving a background model consists of a bottom-up block updating and a top-down model validation process. As shown in Fig. 2.3, a flowchart is given to illustrate the interactions between the two processes. The aim of the bottom-up process is to block-wise integrate identified background blocks into a model and to form a model candidate \tilde{B}_t . Then, in the top-down process, the inter-block consistency for all the updated background blocks are validated. By assuming that significant background updates often occur in groups, isolated updates that mainly result from noises will be eliminated by restoring their block statistics back to the previous estimates \tilde{b}_{t-1}^{*i} .

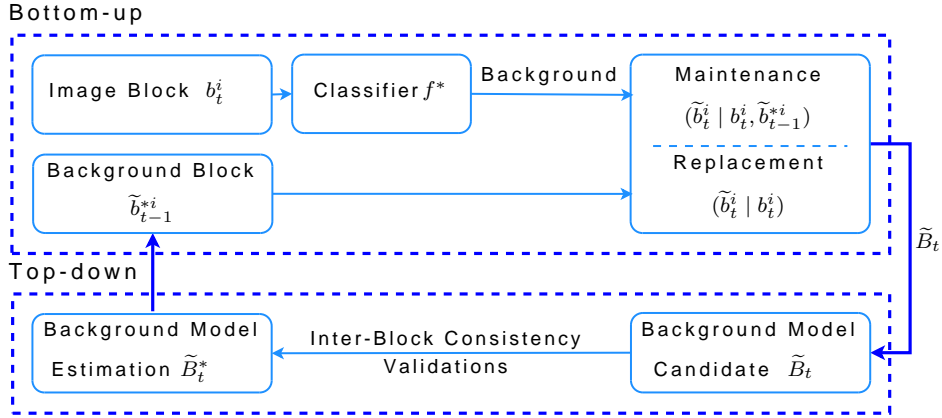


Figure 2.3: Flowchart of the bottom-up and top-down processes. The flowchart depicts the interactions between the bottom-up block updating and the top-down model validation processes. While the bottom-up process handles block-wise updates of the background, the top-down one deals with inter-block consistency validations. The coupling of the two processes forms an efficient scheme for deriving a background model.

More specifically, in the bottom-up process, the image block b_t^i classified as background and the previously estimated background block \tilde{b}_{t-1}^{*i} act as two inputs to the background adaptation. Based on a dissimilarity measure between the current image block b_t^i and the previous background block \tilde{b}_{t-1}^{*i} , either a maintenance step or a replacement step is invoked for a block update. In the maintenance step, the case of the small block difference is handled, assuming it is mostly caused by gradual lighting variations or small vibrations. A new background block estimate \tilde{b}_t^i can thus be computed by a weighted average of the two blocks b_t^i and \tilde{b}_{t-1}^{*i} . On the other hand, when b_t^i and \tilde{b}_t^i are dissimilar, implying an occurrence of an abrupt scene change, a replacement step is employed to calculate a renewed background block estimate \tilde{b}_t^i which is consistent with the image block b_t^i .

After the above bottom-up updates, a background model candidate \tilde{B}_t is obtained. To turn this model candidate into a final estimate, a top-down process

is introduced to assure the model consistency between the current candidate \tilde{B}_t and the previous estimate \tilde{B}_{t-1}^* , by assuming a smooth changing in background models. Though the checking of model consistency can be realized in various ways, we choose to implement it in a simple manner by finding the updates of isolated blocks and undoing them. Thus large and grouped background block updates are preserved in this design, since they most likely belong to significant and stable background changes, such as newly uncovered scenes or stationary objects. Through the validation process, a final background model estimation \tilde{B}_t^* is derived.

It is worth mentioning that the entire approach is linked to a MAP formulation, i.e.,

$$\tilde{B}_t^* = \arg \max_{\tilde{B}_t} \left\{ \underbrace{\left(\prod_{i^+} P(\tilde{b}_t^i | b_t^i, \tilde{b}_{t-1}^{*i}) \prod_{i^-} P(\tilde{b}_t^i | \tilde{b}_{t-1}^{*i}) \right)}_{\text{Likelihood}} \underbrace{P(\tilde{B}_t | \tilde{B}_{t-1}^*)}_{\text{Prior}} \right\}, \quad (2.2)$$

where $i^+ = \{i | b_t^i \text{ is classified as a background block by } f^*\}$ and $i^- = \{1, \dots, n\} - i^+$. (Assume there are n blocks in an image frame.) Interested readers can find the derivation of (2.2) in Appendix. The connections between (2.2) and our approach are elaborated as follows. Regarding the *likelihood* part, the two products can be viewed as block-wise updates after the background classification. For the image block classified as background, maximizing the probability $P(\tilde{b}_t^i | b_t^i, \tilde{b}_{t-1}^{*i})$ implies that similarities among the background estimate \tilde{b}_t^i , the image block b_t^i and the previous estimate \tilde{b}_{t-1}^{*i} should be retained. Likewise, for a foreground block, the corresponding probability $P(\tilde{b}_t^i | \tilde{b}_{t-1}^{*i})$ is maximized by setting the current block estimate \tilde{b}_t^i equal to the previous one \tilde{b}_{t-1}^{*i} . This is what we do in the bottom-up pro-

cess. Referring to the *prior* term, it indicates that model level consistency between \tilde{B}_t and \tilde{B}_{t-1}^* needs to be maintained for maximizing the probability $P(\tilde{B}_t | \tilde{B}_{t-1}^*)$. This, as well, corresponds to the top-down model validation. However, we note that the background model \tilde{B}_t^* derived by our approach is only a rough *approximation* to the MAP solution, since (2.2) is not exactly solved. In fact, to optimize (2.2), the underlying distributions of the probability terms should be further specified, and complicated optimization techniques, e.g., EM-based estimations, may need to be employed. Hence, instead of pursuing the MAP solution, our focus is on the design of a practical and efficient algorithm for background model estimation.

2.2.2 The Detailed Algorithm

Bottom-up process

We start by applying f^* to each b_t^i to determine its probability of being a background block. A simplified notation $P(b_t^i | f^*)$ will be hereafter adopted to denote such a probability, with the understanding that the most recent ℓ i th-blocks b^i s (i.e., $\mathbf{b}_{t,\ell}^i$) are available for calculating useful features, e.g., optical flow values, for classification. Observe that only for those image blocks classified as background at each time t , their corresponding block-wise updatings would modify the background model. It is therefore preferable to have as few false positives by f^* as possible. Hence we use a strict thresholding τ^* , i.e., the decision boundary of f^* , on $P(b_t^i | f^*)$ such that image blocks with $P(b_t^i | f^*) > \tau^* \geq 0.5$ are considered background. Given this setting, there are two possible cases for a block updating.

- If b_t^i is not a background block, then $\tilde{b}_t^{*i} = \tilde{b}_{t-1}^{*i}$, i.e., the pixel means and variances of \tilde{b}_{t-1}^{*i} are assigned to \tilde{b}_t^{*i} .

Algorithm 1: Background model estimation via classification

Data: Process I_t using f^* , \tilde{B}_{t-1}^* and an auxiliary image $\bar{B} = \{\bar{b}^i\}$. When $t = 0$, we have $\tilde{B}_0^* = \emptyset$, $\bar{B} = \emptyset$, and $\forall i$, $age(i) = 0$, $counter(i) = 0$, and $replace(i) = false$.

Result: Obtain a MAP estimate \tilde{B}_t^* .

begin

$\tilde{B}_t^* \leftarrow \tilde{B}_{t-1}^*$

for image block $b_t^i \in I_t$ **do**

if b_t^i is a valid background block **then**

if $diss(b_t^i, \tilde{b}_{t-1}^{*i}) \leq \delta (= 15^2 = 225)$ **then**

 /* Maintenance */

$\tilde{b}_t^{*i} \leftarrow \text{IterativeAverage}(b_t^i, \tilde{b}_{t-1}^{*i})$

$age(i) \leftarrow age(i) + 1$

$counter(i) \leftarrow 0$

$\bar{b}^i \leftarrow 0$

else

 /* Replacement */

$\bar{b}^i = \bar{b}^i + \frac{1}{N}b_t^i$

$counter(i) \leftarrow counter(i) + 1$

if $counter(i) = N$ **then**

$\tilde{b}_t^{*i} = \bar{b}^i$

$age(i) \leftarrow 0$

$counter(i) \leftarrow 0$

$replace(i) \leftarrow true$

else

$age(i) \leftarrow age(i) + 1$

$counter(i) \leftarrow 0$

$\bar{b}^i \leftarrow 0$

output \tilde{B}_t^*

- If b_t^i is classified as a background block, we measure the dissimilarity between b_t^i and \tilde{b}_{t-1}^{*i} by

$$diss(b_t^i, \tilde{b}_{t-1}^{*i}) = \frac{\|b_t^i - \tilde{b}_{t-1}^{*i}\|^2}{|b^i|},$$

where $\|b_t^i - \tilde{b}_{t-1}^{*i}\|^2$ is the sum of squared pixel intensity differences, and $|b^i|$ is the block size. Depending on the value of $diss(b_t^i, \tilde{b}_{t-1}^{*i})$, either a *maintenance step* or a *replacement step* is invoked (see Algorithm 1). We apply the iterative maintenance formulas proposed in [6] to update the latest small variations into \tilde{B}_t^* . Notice that a block replacement in evaluating \tilde{B}_t^* takes place only when the particular block has been classified as background for N consecutive frames. Indeed, the maintenance phase is designed to adapt the gradual variations, and the replacement phase is to accommodate new stationary objects.

Top-down process

A top-down process based on comparing \tilde{B}_t with \tilde{B}_{t-1}^* is employed to detect isolated block updates in the bottom-up evaluation of \tilde{B}_t , and undo these updates with the statistical data from \tilde{B}_{t-1}^* .² Conveniently, in implementing the algorithm, the top-down process can be carried out right after the background block classifications. This would yield a set of *valid* background blocks; all of them are not isolated. Hence, the bottom-up updating over these valid blocks would directly lead to the final estimate \tilde{B}_t^* .

²An isolated block updating (either for maintenance or for replacement) has less than three of its 4-connected neighboring blocks being updated in the bottom-up process.

The background model

Having described our two-phase scheme to iteratively improve \tilde{B}_t^* , we are now in a position to define a meaningful and steady initial background model \tilde{B}^* .

Definition 1. *The initial background model \tilde{B}^* is said to be $\tilde{B}_{t^*}^*$, if t^* is the earliest time instant satisfying the following three conditions:*

- (i) there is no block replacement occurred for the last N image frames, i.e., in calculating $\tilde{B}_{t^*-N+1}^*, \tilde{B}_{t^*-N+2}^*, \dots, \tilde{B}_{t^*}^*$;*
- (ii) all image blocks in $\tilde{B}_{t^*}^*$ have been replaced at least one time since $t = 0$; and*
- (iii) they are of ages at least L . (In all our experiments, we have $N = 45$ and $L = N + 15 = 60$.)*

2.3 Fast Classification with Soft Margins

In this section, issues related to the feature selection and the classifier formulation are addressed for the construction of an efficient background block classifier. In the feature selection, we have chosen to use features as general as possible so that the resulting classifier can handle a broad range of image sequences. Regarding the classifier formulation, two learning methods, *support vector machines* (SVMs) and *column generation boost* (CGBoost), are explored by investigating the following two issues. First, rather than binary-value classifiers, a classifier with probability outputs is required for our application. Second, the efficiency of the resulting classifier should fulfill the demand of real-time performance.

2.3.1 Feature Selection

For our purpose, the task of training is to learn a binary classifier for identifying background blocks from a video sequence captured by a static camera. We use a two-dimensional feature vector to characterize an image block b^i . The first component is the average *optical flow value*, where we apply the Lucas-Kanade’s algorithm [39] to compute the flow magnitude of each pixel in b^i . In our implementation, it takes three image frames, I_{t-2} , I_{t-1} , and I_t , to calculate the flow values properly. However, we note that if one-frame delay is allowed, a slightly better results in evaluating the values of optical flow can be achieved by referencing I_{t-1} , I_t , and I_{t+1} . The second component of a feature vector is derived from the (mean) *inter-frame image difference* by $|b^i|^{-1} \sum_{(x,y) \in b^i} |I_{t-1}^{x,y} - I_t^{x,y}|$. To ensure good classification results, the feature values of both dimensions are normalized into $[0, 1]$ for training and for testing.

The two feature components are discriminant enough for our application owing to their generality and consistency in classifying background blocks of varied image sequences. We should also point out that since the optical flow values are computed using just three consecutive image frames, it may occur that a few pixels would have erratic/large flow values. Hence, an estimated upper-bound threshold is enforced to eliminate such errors. On the other hand, the additional cue using temporal differencing is more stable and easier to calculate, but it may fail to detect all the relevant cases. For example, the inter-frame difference may not be small in evaluating a background block that consists of slightly waving trees. Instead, an optical flow value is more informative to capture such a background block with small motions.

2.3.2 SVMs with Probability Outputs

For binary classifications, SVMs determine a separating hyperplane $f_S(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x})$, $\mathbf{x} \in \mathbf{D}$ by transforming \mathbf{D} from the input space to a high dimensional feature space, through a mapping function ϕ . The optimal hyperplane f_S^* can be obtained by solving the following soft-margin optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi_i} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C_S^+ \sum_{i^+} \xi_i + C_S^- \sum_{i^-} \xi_i \\ \text{subject to} \quad & y_i f_S(\mathbf{x}_i) \geq 1 - \xi_i, \quad i = 1, \dots, m, \end{aligned} \quad (2.3)$$

where $\xi_i \geq 0$ are slack variables for tolerating sample noises and outliers. The two parameters C_S^+ and C_S^- are useful when dealing with unbalanced training data. (Recall that “+” is for background image blocks and “-” for foreground image blocks.) For the sake of reducing false positives, which may lead to more serious flaws in the estimated background model than false negatives would cause, C_S^- is given a value four times larger than the one for C_S^+ to penalize more the misclassifications of foreground blocks. In solving (2.3), we use a degree 2 polynomial kernel to yield satisfactory classification outcomes efficiently.

Probability output

We use a *sigmoid model* to map an SVM score into the probability of being a background block by

$$P(\mathbf{x}|f_S^*) = \frac{1}{1 + \exp(A f_S^*(\mathbf{x}) + B)}, \quad (2.4)$$

where the two parameters A and B can be fitted using *maximum likelihood estimation* from \mathbf{D} . Following [45], a *model-trust* algorithm is applied to solve the two-parameter optimization problem. In our experiments, 65% of the training blocks are used for deriving an SVM, and the other 35% are for calibrating probability outputs. The two fitted parameters are $A = -0.673724$ and $B = -2.359339$.

2.3.3 CGBoost with Probability Outputs

Among the many variants of boosting methods, the AdaBoost, introduced by Freund and Schapire [16], is the most popular one to derive an effective ensemble classifier iteratively. While AdaBoost has been proved to asymptotically achieve a maximum margin solution, recent studies also suggest the adoption of soft margin boosting to prevent the problem of overfitting [12], [46]. We thus employ the linear program boosting proposed by Demiriz et al. [12] for achieving soft-margin distribution over the training data \mathbf{D} and acquiring an ensemble classifier $f_B = \sum_{j=1}^T \alpha_j f_j$, which is comprised of T weak learners f_j s and weights α_j s. Actually, Demiriz et al. apply a column generation method to solve the linear program by part, and establish an iterative boosting process that is similar to AdaBoost. Note that in implementing the CGBoost, the weak learners are constructed from *radial basis function* (RBF) networks, denoted as h s [46]. And each h has three Gaussian hidden units where two of them are initialized for the background, and the remaining one is for the foreground training data. Let $f_j(\mathbf{x}) = \text{sign}(h_j(\mathbf{x}))$ be the weak learner selected at the j th iteration of CGBoost. Then, the RBF network h_j is derived by minimizing the following weighted error function

$$E_j = \frac{1}{2} \sum_{i=1}^m w_i (h_j(\mathbf{x}_i) - y_i)^2, \quad (2.5)$$

where $\{w_i\}$ is the weight distribution over training data \mathbf{D} at the j th iteration.

Probability output

Different from (2.4), it is more convenient to link boosting scores to probabilities. Friedman et al. [18] have proved that the AdaBoost algorithm can be viewed as a stage-wise estimation procedure for fitting an additive logistic regression model. Consequently, a logistic transfer function can be directly applied to map CGBoost scores to posterior probabilities by

$$P(\mathbf{x}|f_B^*) = \frac{1}{1 + \exp(-2f_B^*(\mathbf{x}))}, \quad (2.6)$$

where the mapping in (2.6) is valid when the training data \mathbf{D} do not contain a large portion of noisy samples or outliers. For the general case, it should still yield reasonable probability values with respect to the classification results by f_B^* s.

To summarize, both the two classifiers, f_S^* and f_B^* , seek a soft-margin solution when deciding a decision boundary for the training data \mathbf{D} . They indeed achieve similar classification performance in our experiments. However, SVMs are generally less efficient than boosting, as the number of support vectors increases rapidly with the size of \mathbf{D} . We thus prefer a CGBoost classifier for estimating an initial background model.

2.4 Experimental Results

To demonstrate the effectiveness of our approach, we first describe how the classifiers are learned for the specific problem. We then test the algorithm with a



Figure 2.4: Training images for background model initialization. Examples of collected images and their binary maps of the foreground (white) and the background regions (black) are plotted, top and bottom, respectively.

number of image sequences on a P4 1.8GHz PC. Through illustrating with the experimental results, we highlight the advantages of learning a background model by classification, and make comparisons with those related works. Finally, possible future extensions to the current system are also explored.

2.4.1 Classifier Training

Training data

We begin by collecting images that contain moving objects of different sizes and speeds from various indoor and outdoor image sequences captured by a static camera. These images are analyzed using a tracking algorithm (with known background models), decomposed into 8×8 image blocks, and then manually labeled as +1 for background blocks, or -1 for foreground ones. Examples of the collected images and the detected foreground and background regions are shown in Fig. 2.4. Since we prefer a resulting classifier to accommodate small variations, image blocks from regions of faint shadows or lighting changes are labeled as background. The feature vector of an image block can be computed straightforwardly by referencing

the related $\ell = 3$ blocks from the respective image sequence. Totally, there are 27,600 image blocks collected to form the training data \mathbf{D} . As shown in Fig. 2.5 (a), the features extracted from the background blocks are mostly of small values, while those extracted from the foreground blocks mostly have feature values corresponding to the regions of large motions.

Classifier evaluations

The training and the classification outcomes by implementing the classifier respectively with f_S^* and f_B^* are summarized in Table 2.1. Owing to the soft-margin property of the two classifiers, almost the same training errors have been obtained. However, the classification efficiency of f_B^* is more than 20 times faster than that of f_S^* . To visualize the distribution of a derived classifier, for example, f_B^* , its level curves of the decision scores are plotted in Fig. 2.5 (b). It can be observed that the area of positive scores is located near the lower-left corner, which is consistent with the distribution of feature values computed from the training data.

Probability thresholding

For the sake of reducing false positives, we adopt a stricter probability threshold $\tau^* = 0.6$ in setting the decision boundary of a CGBoost classifier. This value is determined through 10-fold cross validation. In Table 2.2, the average values of the false positive and false negative rates in cross validation with respect to different threshold settings are listed. While false negatives mainly affect the needed time in estimating an initial background model, the false positives, i.e. misclassifying foreground blocks as background, will have direct impacts on the quality of the background model. Thus, it is preferable to choose 0.6 as the probability threshold

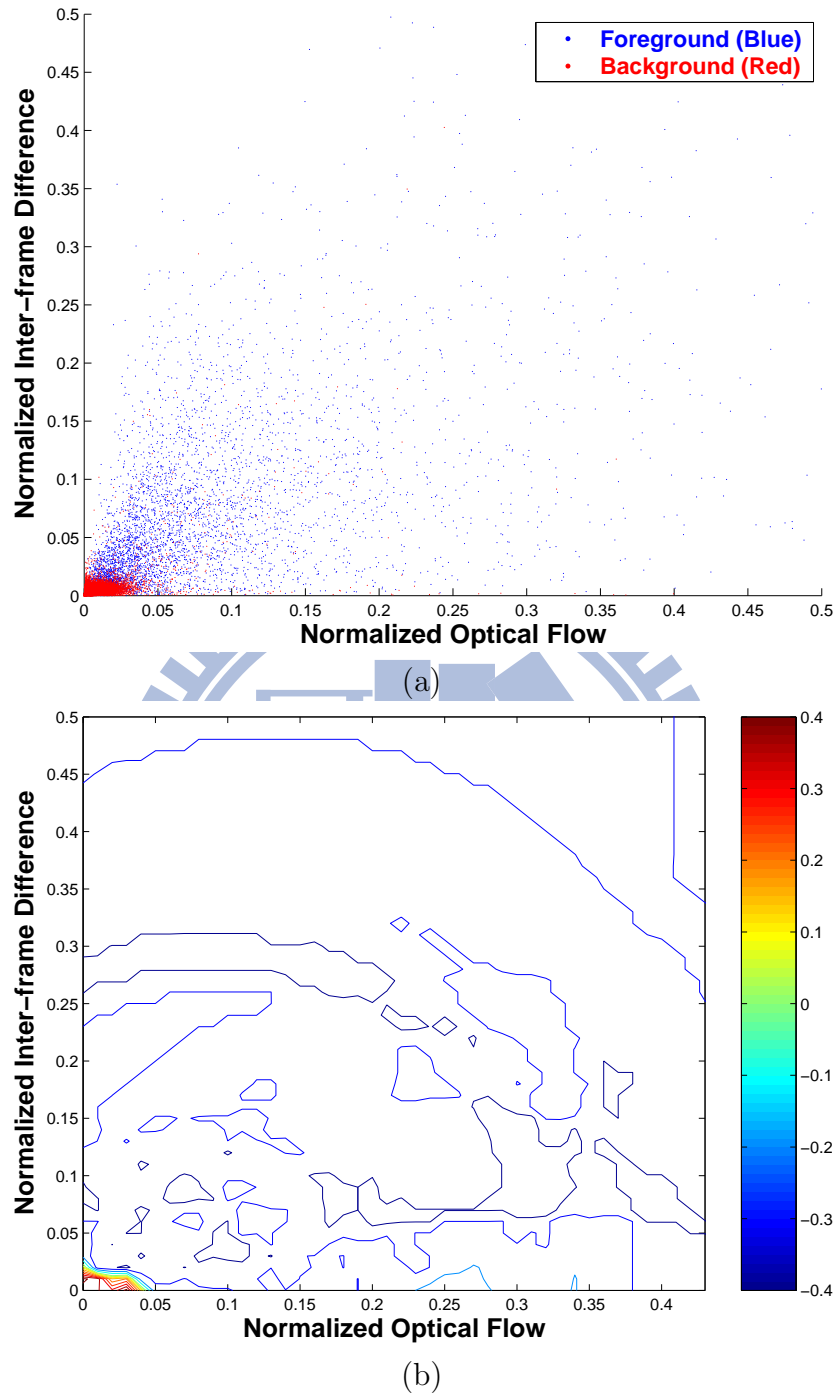


Figure 2.5: Distributions of training data and training results. (a) The distribution of the training data. The training features are normalized to the values between 0 and 1. In order to detail the distribution of background samples, only the part of 0 to 0.5 is plotted. (b) The level curve of f_B^* 's decision scores. The zero-score decision boundary is located on the lower-left corner of the plot.

Table 2.1: Comparisons between SVMs and CGBoost (using AdaBoost as a benchmark).

Classifier	SVM f_S^*	CGBoost f_B^*	AdaBoost f_A
Settings	Image Size: 320×240 , Platform: P4-1.8GHz PC		
Parameters	$C_S^+ = 20$, $C_S^- = 80$	$C_B = \frac{10}{27600}$	(None)
Components	4185 SVs	33 f_{js}	33 f_{js}
Error Rate*	0.0466	0.0466	0.0507
Test Speed	0.4fps	9.5fps	9.5fps

* Error Rate = # of Misclassified Blocks / # of Training Blocks

Table 2.2: Average error rates of 10-Fold cross validation in different threshold settings

τ^*	0.5	0.6	0.7
False Positive	0.02912	0.02768	0.01196
False Negative	0.01877	0.02062	0.49652

in that it causes fewer false positives without introducing too many false negatives.

2.4.2 Performance Evaluation

Since the classification efficiency of CGBoost is more than 20 times faster than that of an SVM implementation (see Table 2.1), we describe below only the experimental results yielded by using the CGBoost classifier f_B^* . For testing the generality of the proposed scheme, all the to-be-estimated scenes of the testing sequences are completely different from those of the training data. The testing sequences also contain complex motions, e.g., substantial object interactions, and varied lighting conditions, like cloudiness.

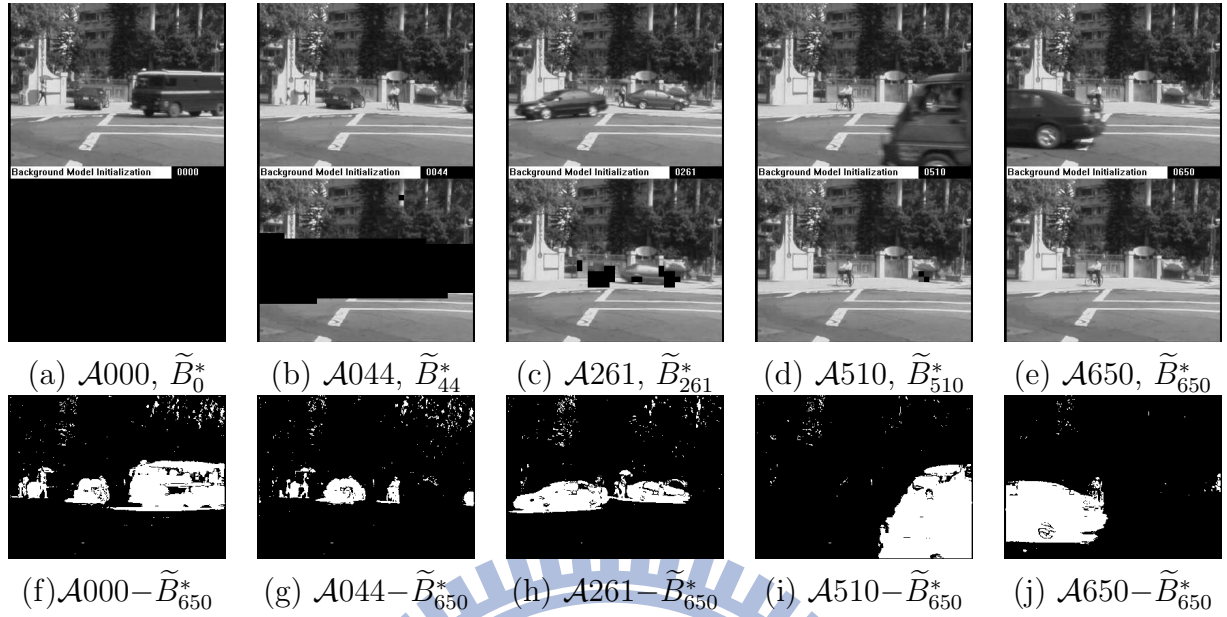


Figure 2.6: Results of background model initialization. (a)–(e) The upper row shows image frames from sequence \mathcal{A} , and the lower row depicts the progressive estimation results. The initial background model is completed at $t^* = 650$. (f)–(j) The frame subtraction results by referencing the derived background model \tilde{B}_{650}^* .

Background model initialization

We first demonstrate the efficiency of our method for an outdoor environment. The sequence \mathcal{A} contains different types of objects, including slightly waving trees, walking people, slow and fast moving vehicles, and even a stationary bike rider. We shall use this example as a benchmark to analyze the quality of our results, detection rates, and comparisons to other existing algorithms. As illustrated in Figs. 2.6 (a) and (b), the background model is initialized into an empty set at $t = 0$, and it is until the 44th frame that stationary regions of the scene are started to be incorporated into the model (due to $N = 45$ in our setting). Fig. 2.6 (c) shows a very slow moving car is falsely adapted into the background in transient (and is eventually removed after its leaving the scene). More interesting is the scenario

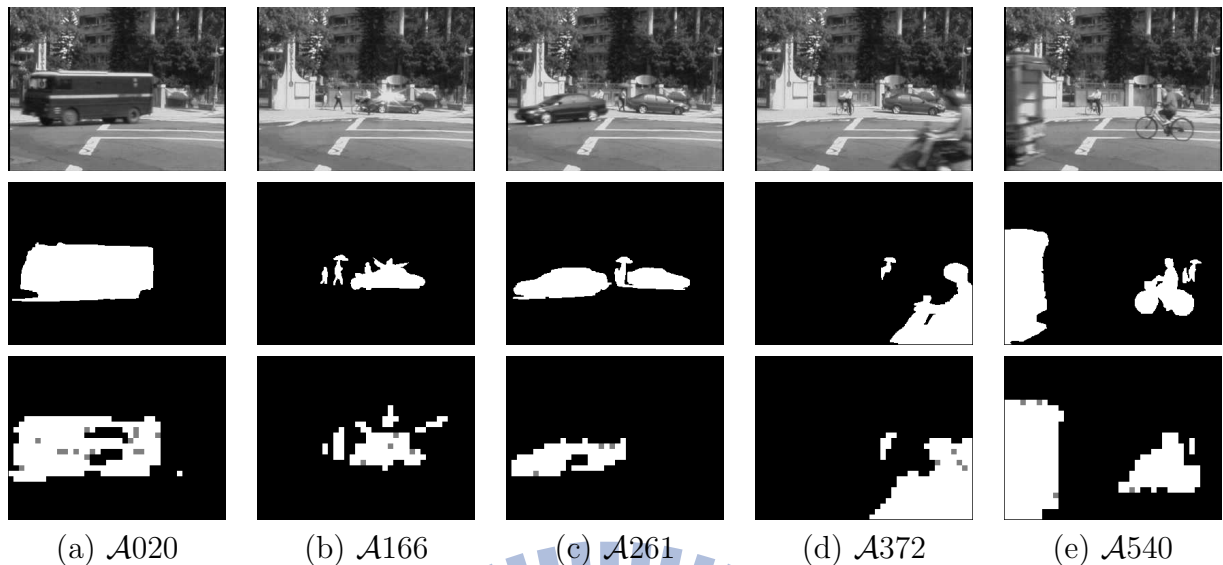


Figure 2.7: Results of background block detection. Row one: Image frames from sequence \mathcal{A} . Row two: The manually labeled foreground (white) and background (black) maps. Note that the very slow-moving car in (c) that later becomes fully stationary in (d) is labeled as foreground and background, respectively. Row three: Our background block detection results. The foreground blocks in gray are identified by the top-down validation process.

depicted in Figs. 2.6 (d) and (e) that a bike rider waiting for a green traffic light has remained still long enough to become a part of the derived back-ground model at $t^* = 650$. Then the system can start to track objects via frame differencing and proper model updating. On the other hand, if we subtract the model from the first t^* frames, it gives the *complexity* of how the background model is initialized. Factors such as dark shadows and waving trees can now be easily identified from those shown in Figs. 2.6 (f)-(j).

Background block detection

To quantitatively evaluate the accuracy of the bottom-up block classifications and the improvement with the top-down validations, we select twenty image frames

Table 2.3: Detection error rates with and without top-down validation.

BG/FG Block Detection	Without top-down	With top-down	% Improvement
Detection Error Rate*	0.04142	0.03779	8.764 %
False Positive Rate	0.02246	0.01825	18.744 %
False Negative Rate	0.01896	0.01954	-3.059 %

* Detection Error Rate = # of Misclassified Blocks / # of Testing Blocks

from sequence \mathcal{A} that contain moving objects of different sizes and speeds, specular light, and shadows. We then manually label each image block of the twenty frames to result in a set of 20061 background and 3939 foreground blocks, where we shall use them to examine the accuracy of our scheme for background block detection. In Fig. 2.7, we show results for five selected frames. Note that those gray blocks are detected as foreground through the top-down validation process. To further justify the need of a local and global approach, a comparison of the detection error rates with or without the top-down validation step is given in Table 2.3. Though the values of detection rates could vary from testing our system in different environments, it is clear that the improvement of reducing the errors by applying the top-down validation is significant. As in this example, the reduction rate of false positives is about 18.744% while the increase rate of false negatives is only 3.059%. Two observations could arise from the foregoing verification for the accuracy of our scheme in detecting background blocks.

- For the classifier to accommodate small variations like waving trees, it may mistakenly classify very slow-moving objects into background (see Fig. 2.6 (c) and Fig. 2.7 (c)). This is indeed a trade-off, and we resolve the issue by learning a proper decision boundary from the training data.

- Our classification scheme may suffer from the aperture problem in detecting large objects in that we use motion features to construct a general classifier (see Figs. 2.7 (a) and (c)). With the top-down validation, this problem can be alleviated to some degree. Still a number of false positives caused by the aperture problem exist frame-wise. However, since only the same false positive occurring for N consecutive frames would be adapted into a background model, such an event rarely happens in practice (with a very low probability, e.g., around 0.01825^N for the example in Table 2.3).

Feature selection

In our design, two general motion cues, the inter-frame difference and the optical flow value, are adopted to discriminate background scenes. While the inter-frame difference is effective in detecting static background blocks, the optical flow value, on the other hand, provides discriminability in classifying image blocks in small motions into gradually-varying background or moving foreground. To further justify the use of the optical flow cue, additional evaluations using the inter-frame difference alone are provided. With the best setting of the difference threshold at 0.013, the training error is raised from 0.0466 to 0.0528 (or a 13.3% increase), and the testing error for the 20 evaluation image frames increases from 0.0378 to 0.0436 (or a 15.3% increase). Hence, the benefit of incorporating the optical flow value is obvious.

Parameter settings

We next investigate the sensitivity of our method with respect to different values of the two parameters N and L . (Since $L = N + 15$, it is indeed a one-parameter

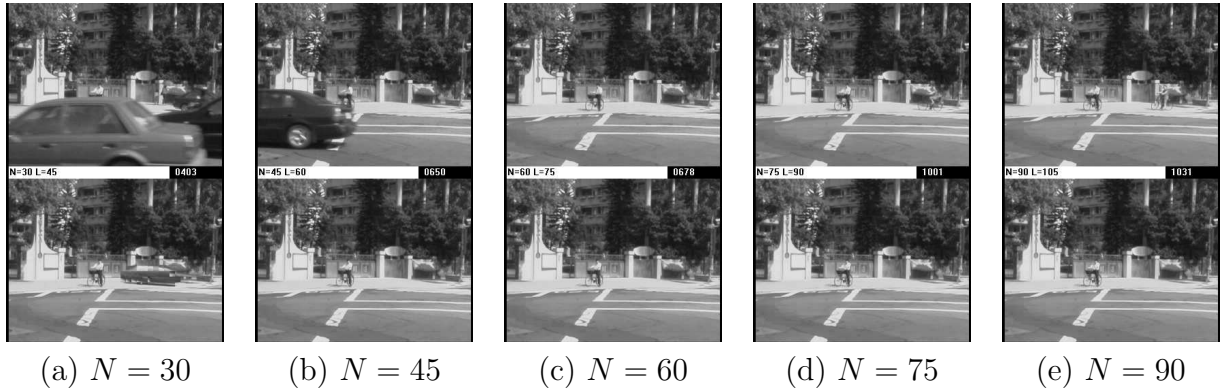


Figure 2.8: Results of different parameter settings. In each case, we show the image frame I_{t^*} (above) that our system completes its estimation for a MAP initial background model (below). Only for $N = 30$, it would produce an unstable estimation due to the violation of stationary criterion. Different values of N and L (given in Definition 1) mainly affect the needed time to derive a stable background model. Respectively, it takes 403, 650, 678, 1001, and 1031 frames to compute the initial background models.

scheme.) Specifically, we have experimented with $N = 30, 45, 60, 75$, and 90 . We show in Fig. 2.8 that, with different values of N and L , it mainly affects the needed time to compute a stable initial background model. The larger the value of N is, the longer period of time it takes to complete the estimation. Except for $N = 30$, which is too short a time period for yielding a stationary adaptation, all other settings of N lead to stable background models.

Comparisons of Background Model Completeness

A clear advantage of our formulation is the ability to know when a well-defined initial background model is ready to be used for tracking. We demonstrate this point by making comparisons with the popular mixture of Gaussians model [54] and the local image flow approach [22]. While the two methods are also effective for background initialization, they both lack a *clear* definition of what an underlying

background scene is at any time instant of the estimation processes. For systems based on the mixture of Gaussians, they work by memorizing a certain number of modes for each pixel, and then by pixel-wise integrating the most probable modes to form a background model. This is in essence a local scheme that the overall quality of a background model is difficult to evaluate. On the other hand, the method described in [22] is designed to process a whole image sequence to output a background model. We thus need to modify the algorithm into a sequential one so that the comparisons can be done by frame-wise examining the respectively derived background models.

The first experiment is carried out with image sequence \mathcal{A} where the three algorithms are alternately run till the image frame $t^* = 650$ that our method completes its estimation for an initial background model. For the mixture model, we use three Gaussian distributions and a blending rate of 0.01, and initialize the background model at $t = 0$ to the first image frame. For the local image flow implementation, the values of w and δ_{max} are set to 30 and 15, and the background model is an empty set at $t = 0$. In Fig. 2.9, we show some intermediate results of ours and the corresponding background models produced by the other two methods. Due to the batch nature of the local image flow scheme, its three background models shown in Fig. 2.9 are obtained by running the algorithm three times, using the respective periods of image frames as the inputs. Overall, the results produced by ours and the mixture of Gaussians are more reliable than those of the local image flow, largely because the local flow scheme relies heavily on the estimations of optical flow directions and their accuracy. While the outcomes by the mixture of Gaussians seem to be satisfactory and similar to ours, the absence of a good measurement to guarantee the quality of the resulting background models

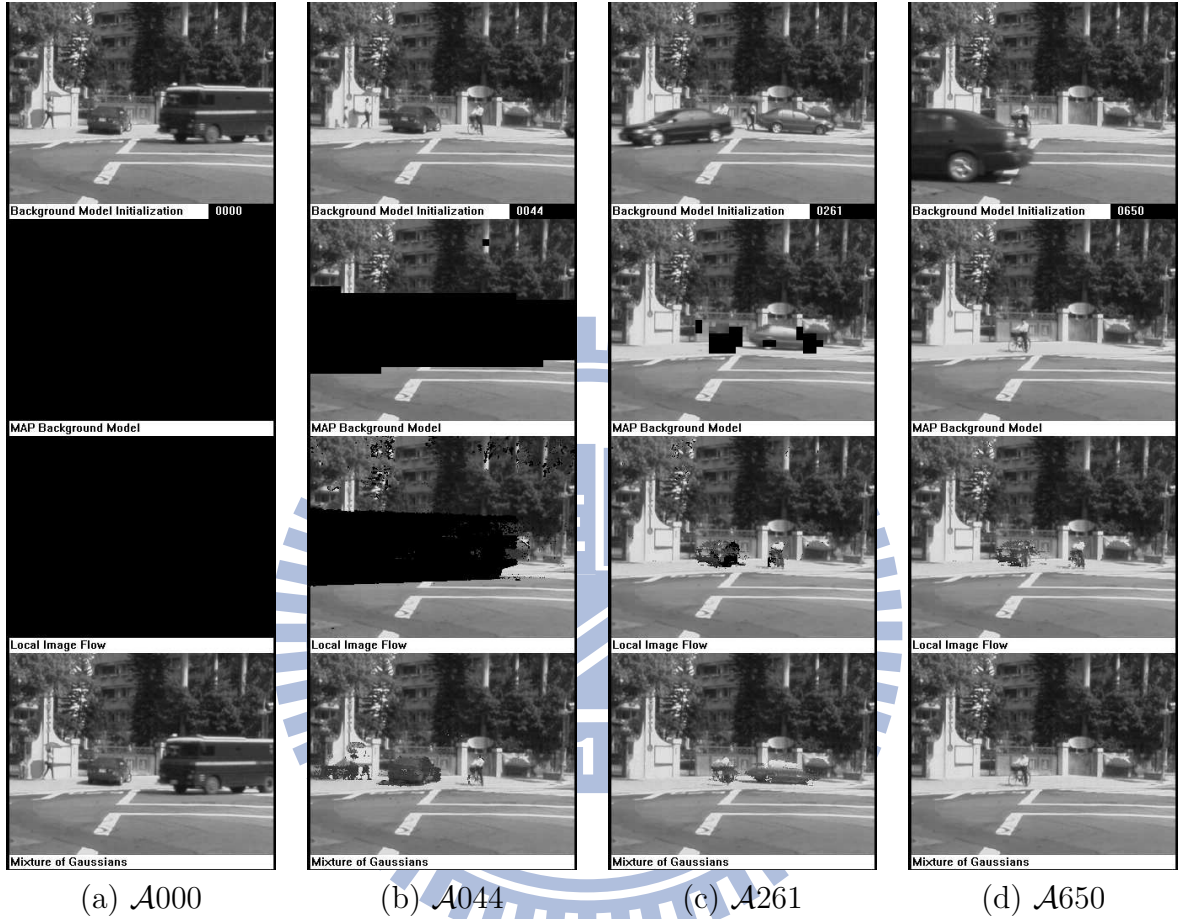


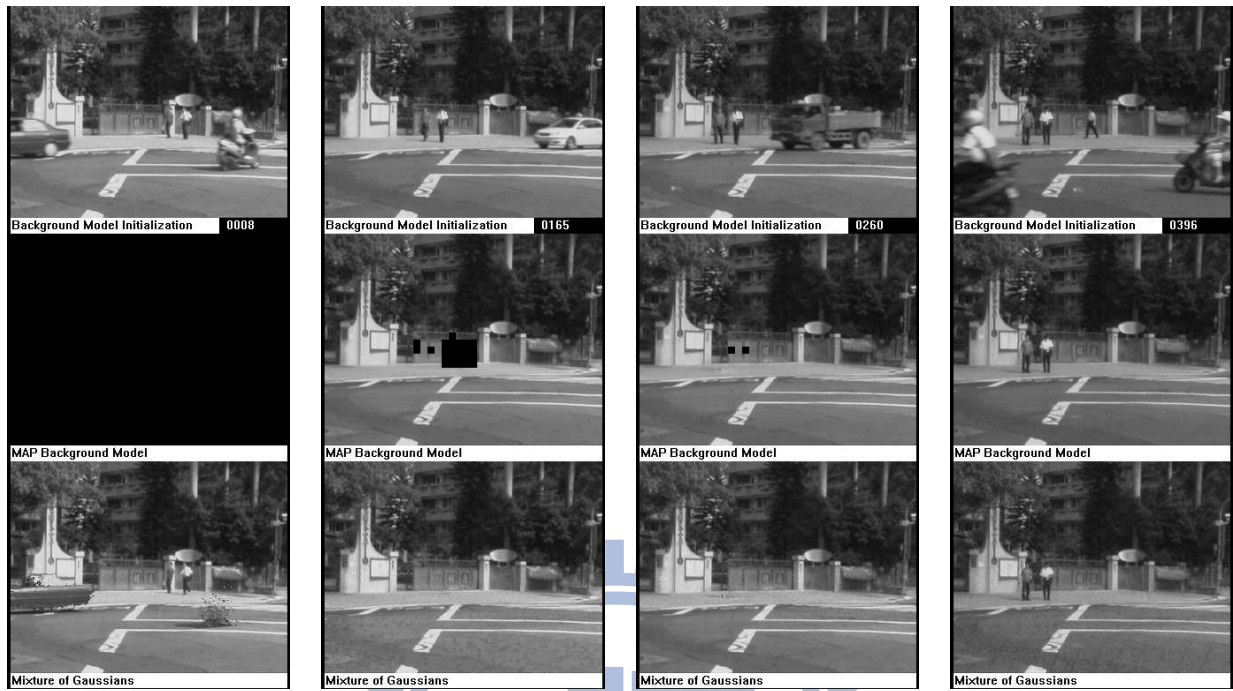
Figure 2.9: Comparisons of [22], [54], and our approach. Row one: Images frames from image sequence \mathcal{A} . Row two: The intermediate results of background estimation by our method that completes at $t^* = 650$. Row three and four: The results respectively derived by the local image flow approach [22] and the mixture of Gaussians method [54] at each corresponding time instant.

remains a disadvantage of the approach. Furthermore, as one would expect that a mixture of Gaussians method should be sensitive to lighting variations in that it is done by locally combining pixel intensities. We shall further elaborate on this issue with the next experiment.

Our second comparison focuses on the effects of lighting changes. For the outdoor sequence \mathcal{B} (see Fig. 2.10), the lighting condition varies rapidly due to overcast clouds. And the experimental results show that our method is less sensitive to variations of this kind. Specifically, in Figs. 2.10 (e) and (f), we enlarge the sizes and enhance the contrasts of the two derived background models for a clearer view. Note that especially in the road area our background model estimation is clearly of better quality than the one yielded by the mixture of Gaussians. This is mostly because of our uses of motion cues for identifying background blocks and the properties of the MAP background model for integrating local and global consistency. On the other hand, the mixture of Gaussians approach uses only the pixel-wise intensity information so that its performance depends critically on the variations of intensity distribution about the background scene.

Initialization and tracking

To further illustrate the efficiency of using our proposed algorithm to estimate a background model for tracking, we show the estimations of initial background models of test sequences \mathcal{C} and \mathcal{D} , and some subsequent tracking results in Fig. 2.11. Below each depicted image frame I_t , the corresponding background model \tilde{B}_t^* is plotted. In the two experiments, the estimations of the initial background model $\tilde{B}_{t^*}^*$ are completed at frame number $t^* = 470$, and 243, respectively. Once the $\tilde{B}_{t^*}^*$ is available, the system can start to track objects immediately, using the scheme



(a) $\mathcal{B}008$

(b) $\mathcal{B}165$

(c) $\mathcal{B}260$

(d) $\mathcal{B}396$



(e) Mixture of Gaussians



(f) MAP

Figure 2.10: Tests on lighting variations with the sequence \mathcal{B} . (a)–(d) Due to overcast clouds, the outdoor lightings over the road change significantly throughout the sequence. As a result, the quality of background models yielded by the mixture of Gaussians is considerably affected. However, our formulation is more robust to such lighting perturbations. (e)–(f) The two derived background models at $t^* = 475$ are enlarged and enhanced in contrast. False textures and extra noises can be observed in the road areas of (e).

described in [6]. We also note that, as demonstrated in Figs. 2.11 (j)-(l), the background model can be updated appropriately during tracking, even when significant changes in the scene background have occurred.



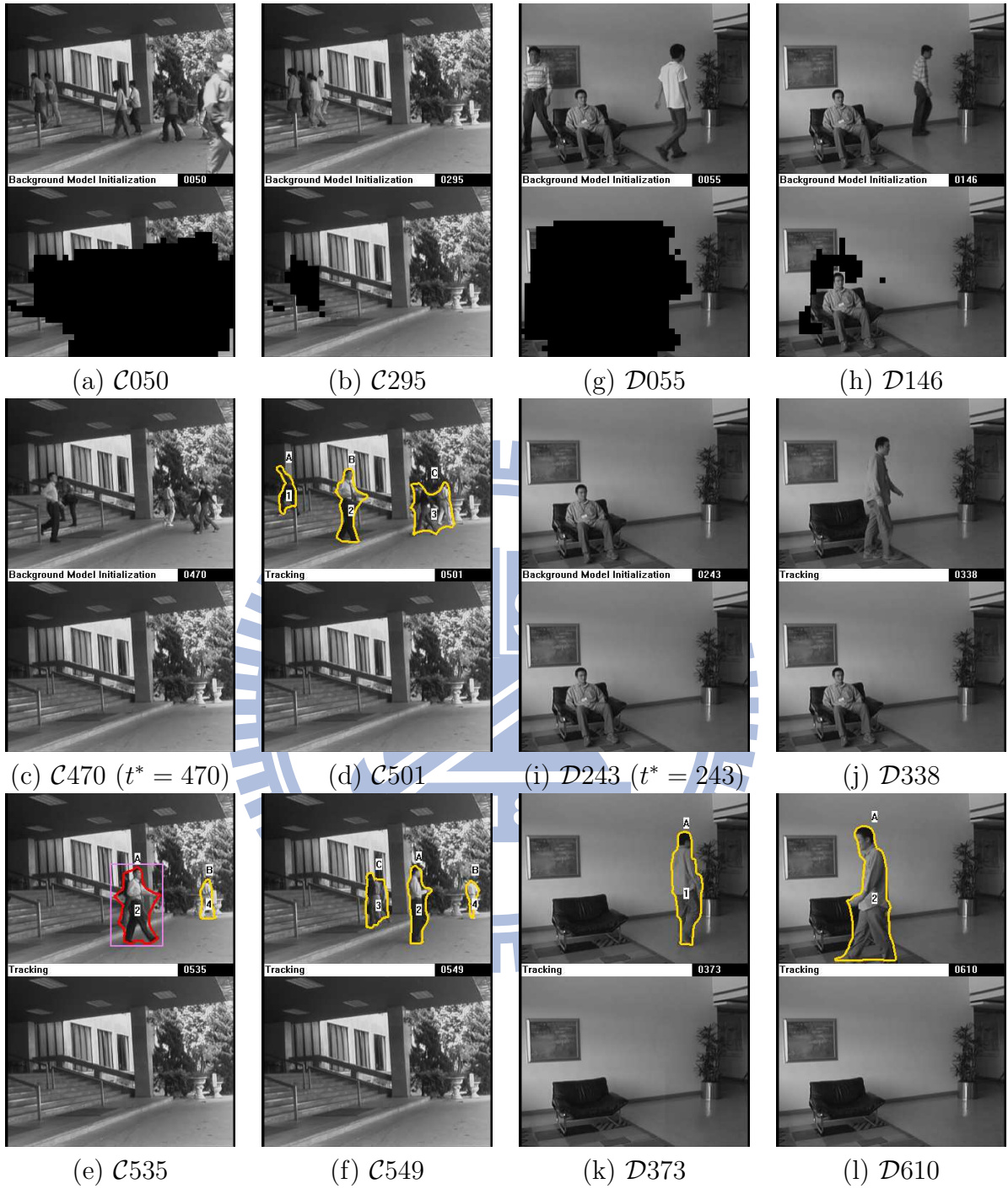


Figure 2.11: Background model initialization and tracking. The current image frame I_t and the derived background model \tilde{B}_t^* are plotted together, top and bottom, respectively. Some Tracking results are also shown in the I_t s.

Chapter 3

Background Model Maintenance via Density Estimation

To model a scene for background subtraction, Gaussian mixture modeling (GMM) is a popular choice for its capability of adaptation to background variations. However, GMM often suffers from a trade-off between robustness to background changes and sensitivity to foreground abnormalities, and is inefficient in managing the trade-off for various surveillance scenarios. By reviewing the formulations of GMM, we identify that such a trade-off can be easily controlled by adaptive adjustments of the GMM's *learning rates* for image pixels at different locations and of distinct properties. A new bivariate rate control scheme based on a feedback of high-level information is then developed to provide better regularization of background adaptation for GMM and to help resolving the trade-off. Additionally, to handle lighting variations that change too fast to be caught by GMM, a heuristic rooting in frame difference is proposed to assist the proposed rate control scheme for reducing false foreground alarms. Experiments show the proposed

bivariate learning rate control scheme, together with the heuristic for adaptation of *double-quick* lighting change, gives better performance than conventional GMM approaches.

3.1 Overview

For video surveillance using a static camera, background subtraction is often regarded as an effective and efficient method for differentiating foreground objects from a background scene. The performance of background subtraction highly depends on how a background scene is modeled. Ideally, a perfect design of background modeling should be able to tolerate various background variations without losing the sensitivity in detecting abnormal foreground objects. However, the trade-off between statistical robustness and sensitivity in background modeling is commonly encountered in practice and is hard to be balanced within a single computational framework.

Among various background modeling approaches, e.g., [4], [9], [13], [19], [21], [24], [33], [40], [41], [43], [47], [48], [50], [51], [54], [58], [64], [68], [70], the Gaussian mixture modeling (GMM) [19], [21], [54] is known to be effective in sustaining background variations, e.g., waving trees, due to its use of multiple buffers to memorize scene states. It is hence widely adopted as a base framework in many later developments [25], [26], [27], [36], [42], [55], [71]. However, the GMM often suffers from the trade-off between statistical robustness to background changes and sensitivity to foreground abnormalities, abbreviated as *R-S trade-off* in later discussions. For instance, a Gaussian mixture model being tuned to tolerate quick changes in background may also adapt itself to stationary objects, e.g., unattended

bags left by passengers, too quickly to issue reliable alarms. The lack of a simple and flexible way to manage the R-S trade-off for various scenarios motivates this research to re-examine the formulations of the GMM.

In the original formulations of the GMM, every image pixel, regardless of its intensity being changing or not, is given the same setting of *learning rates* in background model estimation, which is inefficient in managing the R-S trade-off. Considering a pixel of background that was just uncovered from occlusion of a moving object, the corresponding Gaussian mixture model for this pixel should be updated in a slower pace than that for a stable background pixel, to prevent false inclusion of moving shadows or motion blurs into background. Nonetheless, in the original GMM formulations, an identical learning rate setting is applied to all image pixels, leaving no space for tuning the background adaptation speeds for this case. We therefore highlight the importance of adaptive learning rate control in space and in time, and develop a new bivariate rate control scheme based on the high-level feedback of pixel properties for GMM.

Features of the proposed scheme of bivariate learning rate control for GMM are in several folds. Firstly, two types of learning needs are identified for a Gaussian mixture model (for an image pixel), one for controlling the model estimation accuracy and the other for regularizing the R-S trade-off. Different from previous works, e.g., [25], [54], that use a single learning rate setting for both learning needs, the proposed bivariate rate control scheme distinguishes two different types of learning rates and manipulates them independently. Secondly, the background adaptation rates for image pixels are set individually in space. Image pixels at different locations may thus exhibit distinct behaviors in background adaptation for accommodating local scene changes. Thirdly, for every image pixel, its learning

rate for regularizing the R-S trade-off is computed based on the high-level feedback of its latest pixel type, i.e., as background, stationary foreground, moving foreground, etc. Under this feedback control, the learning rate setting for an image pixel can be dynamically adjusted in time, according to its type, and with respect to different application scenarios¹. The more pixel types are allowed, the higher flexibility in background adaptation can be attained. Fourthly, a heuristic for *adaptation of double-quick lighting change* is suggested to assist the learning rate control to adapt very rapid lighting changes in background. This heuristic enhances the model robustness to speedy lighting variations without sacrificing the sensitivity in detection of significant foreground motions. To sum up, we maintain that via a careful design of learning rate control for the GMM, the R-S trade-off can be effectively and efficiently regularized in fulfilling various needs in video surveillance.

3.1.1 Related Work

Balancing the R-S trade-off has long been an important task in background modeling. In [58], Toyama *et al.* explore several scenarios that are hard to be handled by background modeling, and propose a hybrid approach to maintain background models at different spatial scales. In [4], Boulton *et al.* apply different learning rates to foreground and background pixels to increase the model sensitivity for single Gaussian formulation and develop cleaning algorithms to reduce false alarms. In [20], Gao *et al.* use statistical analysis to tune parameters in background modeling, including the number of Gaussian components and the learning rate, for controlling

¹For example, while pixels of stationary objects may need to be quickly adapted into background for the application of moving object detection, they should be stably identified as foreground for the application of unattended object detection.

the trade-off. In [37], Li *et al.* utilize spatio-temporal features to model complex backgrounds and develop a criterion to select the learning rate for the adaptation of *once-off* background change. In [25], Harville discusses some trade-offs frequently encountered by the GMM and adopts high-level feedback as a remedy. Also based on the GMM, Tian *et al.* propose a weight exchange scheme based on object-level feedback to prevent foreground fragmentation in the detection of static object [55]. In [71], Zivkovic analyzes the appropriate number of mixture components for the GMM and dynamically removes some mixture components for computational efficiency. In [36], Lee proposes a new rate control formulation for the learning of Gaussian parameters to enhance the accuracy and *convergence speed* of background model estimation. Model robustness to background changes is improved by Lee's learning rate control without obvious side-effects on model sensitivity. In [66], a two-layer GMM is proposed by Yang *et al.* to learn foreground and background models at different learning rates and to achieve better foreground segmentation results. Beyond Gaussian-based formulations, Elgammal *et al.* adopt kernel density estimation to compute background models, and combine short-term and long-term models to balance the R-S trade-off [13]. Despite the effectiveness in background modeling for all the approaches mentioned above, no comprehensive investigation into the relationship between model learning rates and the trade-off control for different surveillance scenarios within a single background modeling framework has been conducted.

Note also that the idea of adopting high-level feedbacks, e.g., using foreground pixel type, in background modeling is not new [4], [25], [55], [66]. Yet the proposed feedback control over learning rates has several novel features. Firstly, to the best of our knowledge, the proposed work is the first to apply independent controls

over two types of learning rates for simultaneously enhancing the model estimation accuracy and regularizing the R-S trade-off. High-level feedbacks are applied only to the learning rate control related to the R-S trade-off. Based on our study, this independent control of two-type learning rates is a key to derive a robust background modeling system. Secondly, a new rate control framework capable of managing multiple pixel types as feedbacks is demonstrated to be practical and feasible. Thirdly, the need of dynamically adjusting the learning rates for pixels of background type is firstly identified in this study. This particular learning rate control for background pixels can increase model sensitivity to hovering objects with little side-effect to model robustness.

3.1.2 Model Accuracy, Robustness and Sensitivity

To estimate a density distribution from a sequence of intensities $I_{0,\mathbf{x}}, \dots, I_{t,\mathbf{x}}$ ² for a pixel at a position \mathbf{x} via the GMM, three issues regarding model accuracy, robustness and sensitivity need to be addressed. Specifically, a mixture model consisting of N Gaussian distributions at time instance t can be denoted by

$$P(I_{t,\mathbf{x}}) = \sum_{n=1}^N w_{t-1,\mathbf{x},n} \mathcal{N}(I_{t,\mathbf{x}}; \mu_{t-1,\mathbf{x},n}, \sigma_{t-1,\mathbf{x},n}^2),$$

where \mathcal{N} symbolizes a Gaussian probability density function

$$\mathcal{N}(I; \mu, \sigma^2) \doteq \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(I - \mu)^2}{2\sigma^2}\right),$$

²Here $I_{t,\mathbf{x}} \in \mathbb{R}$ denotes the 1-D pixel intensity only. Yet, all our formulations can be easily extended to multi-dimensional color image processing, e.g., $\mathbf{I}_{t,\mathbf{x}} \in \mathbb{R}^3$.

$\mu_{t-1,\mathbf{x},n}$ and $\sigma_{t-1,\mathbf{x},n}^2$ are the Gaussian parameters of the n th model, and $w_{t-1,\mathbf{x},n}$ is the respective mixture weight. For maintaining this mixture model, the parameters μ_{t-1} , σ_{t-1}^2 and w_{t-1} need to be updated based on a new observation $I_{t,\mathbf{x}}$. In the GMM, the update rule for μ , for the case that $I_{t,\mathbf{x}}$ matches the n th Gaussian model, is

$$\mu_{t,\mathbf{x},n} = (1 - \rho)\mu_{t-1,\mathbf{x},n} + \rho I_{t,\mathbf{x}},$$

where $\rho \in [0, 1]$ is a *learning rate*³ that controls how fast the estimate μ converges to new observations. Likewise, similar update rules can be applied to renewing σ^2 and w , given corresponding learning rates.

In updating the Gaussian parameters μ and σ^2 , their values should reflect the up-to-date statistics of a scene as accurately as possible. It is thus preferable to set their learning rates to large values to quickly derive Gaussian distributions that fit new observations. Also as noted in [36], setting higher learning rates for μ and σ^2 improves model convergency and accuracy, and brings few side-effect in model stability.

While the model estimation accuracy depends on the learning rates for μ and σ , one can see that the R-S trade-off is affected by the learning rate for the mixture weight w . In the original GMM for background model estimation, the classification of Gaussian models into foreground and background is done by evaluating their mixture weights through thresholding. The Gaussian models that appear more often will receive larger weights in the model updating process, and will possibly be labeled as background [54]. However, the frequency of model occurrence should not be the only factor that guides the changes of mixture weights. For example,

³The definition of learning rate is inherited from [54].

one may prefer to give large weights to the Gaussian models of tree shadows (for background adaptation) while to keep small weights to those of parked cars (for foreground detection), despite the similar frequencies of occurrence of these two objects. By incorporating the high-level information of pixel types, e.g., of shadow or car, into the weight updating process, flexible background modeling can then be carried out. As more pixel types are designated by a surveillance system, more appropriate controls on weight changes can be advised accordingly, which will help resolving the R-S trade-off in background modeling. Based on this observation, we propose a new bivariate learning rate control scheme based on a feedback of pixel type for GMM.

3.2 Bivariate Learning Rate Control via High-Level Feedback

Our presentations of the proposed bivariate learning rate control via high-level feedback is divided into three parts. Firstly, an algorithm of *background model maintenance* using the GMM is proposed, wherein two types of learning rates are formally defined. We highlight the importance of the learning rate control for mixture weights and elaborate its relationship to foreground pixel labeling. Secondly, a feedback scheme that controls the learning rates for mixture weights is detailed. Under this feedback control, different learning rates can be applied to different image locations and scene types, which makes dynamic background adaptation possible. Thirdly, a heuristic based on frame difference is introduced to assist the learning rate control for the adaptation of double-quick lighting changes.

False alarms caused by, for example, sudden sunshine changes in the background can hence be suppressed by this heuristic while significant, object motions can still be captured.

3.2.1 Background Model Maintenance

Given a new observation of pixel intensity $I_{t,\mathbf{x}}$, the task of background model maintenance is to match this new observation to existing Gaussian distributions, if possible, and to renew all the parameters of the Gaussian mixture model for this pixel. The detailed steps of the proposed background model maintenance using the GMM is shown in Algorithm 2.

For the model matching in Algorithm 2, $l(t, \mathbf{x})$ is utilized to index the best matched Gaussian model of $I_{t,\mathbf{x}}$, if existing. Otherwise, $l(t, \mathbf{x}) = 0$ will be set to indicate $I_{t,\mathbf{x}}$ is a brand-new observation and should be modeled by a new Gaussian distribution. The matching results of $I_{t,\mathbf{x}}$ can be recorded by model matching indicators, i.e.,

$$M_{t,\mathbf{x},n} = \begin{cases} 1, & \text{if } n = l(t, \mathbf{x}), \\ 0, & \text{otherwise,} \end{cases} \quad \text{for } n = 1, \dots, N,$$

and will be used in the later model update. Unlike [54] that adopts a more complex formulation in model matching, i.e.,

$$l(t, \mathbf{x}) = \arg \min_{n=1,\dots,N} \frac{|I_{t,\mathbf{x}} - \mu_{t-1,\mathbf{x},n}|}{\sigma_{t-1,\mathbf{x},n}}, \quad (3.1)$$

a simple rule that selects the model of higher weight as the best match is used in

Algorithm 2. The proposed *weight-based matching rule* prefers matching a pixel observation to the Gaussian model of background (with higher weight) other than those of foreground, if this observation falls in the scopes of multiple models. Using this rule not only saves computational costs but also fits the proposed rate control scheme better, as will be discussed in more detail later.

After model matching, we check if $M_{t,\mathbf{x},l(t,\mathbf{x})}$ is equal to 0, which implies no model matched. If so, a model replacement is performed to incorporate $I_{t,\mathbf{x}}$ into the GMM; otherwise, a model update is executed. In the replacement phase, the least weighted Gaussian model is replaced by the current intensity observation. In the update phase, the following three rules,

$$\mu_{t,\mathbf{x},l(t,\mathbf{x})} = (1 - \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha)) \mu_{t-1,\mathbf{x},l(t,\mathbf{x})} + \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) I_{t,\mathbf{x}}, \quad (3.2)$$

$$\sigma_{t,\mathbf{x},l(t,\mathbf{x})}^2 = (1 - \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha)) \sigma_{t-1,\mathbf{x},l(t,\mathbf{x})}^2 + \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) (I_{t,\mathbf{x}} - \mu_{t,\mathbf{x},l(t,\mathbf{x})})^2, \quad (3.3)$$

$$w_{t,\mathbf{x},n} = (1 - \eta_{t,\mathbf{x}}(\beta)) w_{t-1,\mathbf{x},n} + \eta_{t,\mathbf{x}}(\beta) M_{t,\mathbf{x},n}, \quad (3.4)$$

are applied, where $\rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) \in \mathbb{R}$ denotes the learning rate for the Gaussian parameters μ and σ^2 , and $\eta_{t,\mathbf{x}}(\beta) \in \mathbb{R}$ is a new learning rate introduced in this research for controlling the updating speed of the mixture weight w . Here, the two scalars α and β can be viewed as hyper-parameters over ρ and η for tuning their values. In [54], the learning rate ρ is defined as

$$\rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) \doteq \alpha \mathcal{N}(I_{t,\mathbf{x}}; \mu_{t-1,\mathbf{x},l(t,\mathbf{x})}, \sigma_{t-1,\mathbf{x},l(t,\mathbf{x})}^2), \quad (3.5)$$

while in [36] it is given by

$$\rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) \doteq \left(\frac{1 - \alpha}{c_{t,\mathbf{x},l(t,\mathbf{x})}} + \alpha \right), \quad (3.6)$$

where $c_{t,\mathbf{x},n} = c_{t-1,\mathbf{x},n} + M_{t,\mathbf{x},n}$ and $c_{t=0,\mathbf{x},n} = 0, \forall \mathbf{x}, n$.⁴ Although (3.6) may result in quicker convergence in Gaussian parameter learning [36], we still choose (3.5) in our implementation for experimental comparisons and put our emphasis on the control of the learning rate η for the mixture weight. In later experiments we will show that better performance can be achieved by controlling the learning rate η than by tuning the rate ρ . Also, as noted in [36], typical values of α are in $[0.1, 0.001]$ for both (3.5) and (3.6), yielding a wide range of convergence rates in Gaussian parameter estimation. Here we set $\alpha = 0.025$ as a default value for quick model learning.

In previous background modeling researches, e.g., [25], [36], [54], a naive setting for mixture weight update, i.e.,

$$w_{t,\mathbf{x},n} = (1 - \alpha) w_{t-1,\mathbf{x},n} + \alpha M_{t,\mathbf{x},n}, \quad (3.7)$$

is adopted. The rule (3.7) can be viewed as a special case of the proposed weight update of (3.4) with $\eta_{t,\mathbf{x}} = \alpha$. In (3.7), all image pixels are confined to having an identical rate setting in mixture weight learning, so that scene changes can not be properly handled with respect to space and time. Instead, with our generalization that assigns individual learning rates for mixture weights to image pixels and adapts them over time, higher flexibility in regularizing background adaptation

⁴Interested readers can find the details in [36].

Algorithm 2: Background model maintenance

```
1 Parameters:  $T_\sigma (= 2.5)$ ,  $\sigma_0^2 (= 10^2)$ ,  $w_0 (= 0.01)$ 
2 // Model matching
3  $M_{t,\mathbf{x},n} = 0$ ,  $\forall n = 1, \dots, N$ 
4  $d_{t,\mathbf{x},n} = \inf$ ,  $\forall n = 1, \dots, N$ 
5 for  $n = 1, \dots, N$  do
6   if  $|I_{t,\mathbf{x}} - \mu_{t-1,\mathbf{x},n}| \leq T_\sigma \sigma_{t-1,\mathbf{x},n}$  then  $d_{t,\mathbf{x},n} = -w_{t-1,\mathbf{x},n}$ 
7  $l(t, \mathbf{x}) = \arg \min_{n=1, \dots, N} d_{t,\mathbf{x},n}$ 
8 if  $d_{t,\mathbf{x},l(t,\mathbf{x})} \neq \inf$  then  $M_{t,\mathbf{x},l(t,\mathbf{x})} = 1$  else  $l(t, \mathbf{x}) = 0$ 
9 // Model renewing
10  $w_{t,\mathbf{x},n} = (1 - \eta_{t,\mathbf{x}}(\beta)) w_{t-1,\mathbf{x},n} + \eta_{t,\mathbf{x}}(\beta) M_{t,\mathbf{x},n}$ ,  $\forall n$ 
11 if  $M_{t,\mathbf{x},l(t,\mathbf{x})} = 1$  then
12   // Update phase
13    $\rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) = \alpha \mathcal{N}(I_{t,\mathbf{x}}; \mu_{t-1,\mathbf{x},l(t,\mathbf{x})}, \sigma_{t-1,\mathbf{x},l(t,\mathbf{x})}^2)$ 
14    $\mu_{t,\mathbf{x},l(t,\mathbf{x})} = (1 - \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha)) \mu_{t-1,\mathbf{x},l(t,\mathbf{x})} + \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) I_{t,\mathbf{x}}$ 
15    $\sigma_{t,\mathbf{x},l(t,\mathbf{x})}^2 = (1 - \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha)) \sigma_{t-1,\mathbf{x},l(t,\mathbf{x})}^2 + \rho_{t,\mathbf{x},l(t,\mathbf{x})}(\alpha) (I_{t,\mathbf{x}} - \mu_{t,\mathbf{x},l})^2$ 
16 else
17   // Replacement phase
18    $k = \arg \min_{n=1, \dots, N} w_{t-1,\mathbf{x},n}$ 
19    $\mu_{t,\mathbf{x},k} = I_{t,\mathbf{x}}$ 
20    $\sigma_{t,\mathbf{x},k}^2 = \sigma_0^2$ 
21    $w_{t,\mathbf{x},k} = w_0$ 
22  $w_{t,\mathbf{x},n} = w_{t,\mathbf{x},n} / \sum_{n=1}^N w_{t,\mathbf{x},n}$ ,  $\forall n$ 
```

can be obtained. Note that the index n is not attached to $\eta_{t,\mathbf{x}}$ because the changing rates for the weights $w_{t,\mathbf{x},n}$ s, $\forall n$, are designed to be consistent among the N Gaussian models of the same image pixel. Regarding the computation of $\eta_{t,\mathbf{x}}$, we link it to the high-level feedback of pixel types and describe the feedback control in Sec. 3.2.2.

In the GMM, all the scene changes, regardless of being foreground or background, are modeled by Gaussian distributions. To further distinguish these two classes, a foreground indicator $F_{t,\mathbf{x},n}$ for each Gaussian model is defined using the

corresponding mixture weight as

$$F_{t,\mathbf{x},n} = \begin{cases} 0 & \text{if } w_{t,\mathbf{x},n} \geq T_w, \\ 1 & \text{otherwise,} \end{cases} \quad (3.8)$$

where $T_w \in \mathbb{R}$ is a preset parameter.⁵ A binary foreground map can then be defined as a set $\mathcal{F}_t = \{F_{t,\mathbf{x},l(t,\mathbf{x})} | \forall \mathbf{x}\}$. In the original GMM formulations applying (3.7), more frequently matched Gaussian models will have larger weights and will be labeled as background. Nevertheless, stationary objects, e.g., abandoned packages or standing persons, that appear constantly within a restricted area should not always be absorbed into background for some applications. Rather, these objects may need to be stably highlighted as foreground and alarms should be triggered if necessary. By adaptively adjusting $\eta_{t,\mathbf{x}}$ in (3.4) based on object types, as will be discussed next, such demands may be fulfilled without resorting to complex versions of (3.8) for foreground and background separation.

3.2.2 Feedback Control

A flowchart of a general-purposed surveillance system is illustrated in Fig. 3.1, where five processing modules are presented in a sequential manner. To address the above issue associated with object types, the final results derived by the last module of *object type classification* is fed back to the first one of *background model maintenance* for further control of the learning rates. Rather than digging into the details of each module wherein different implementations can be accommodated, we place the focus on the learning rate control for mixture weights in the following

⁵The procedure of model sorting by the values of w/σ , as suggested in [54], is not applied here since it is more complex and may cause complications in foreground pixel labeling.

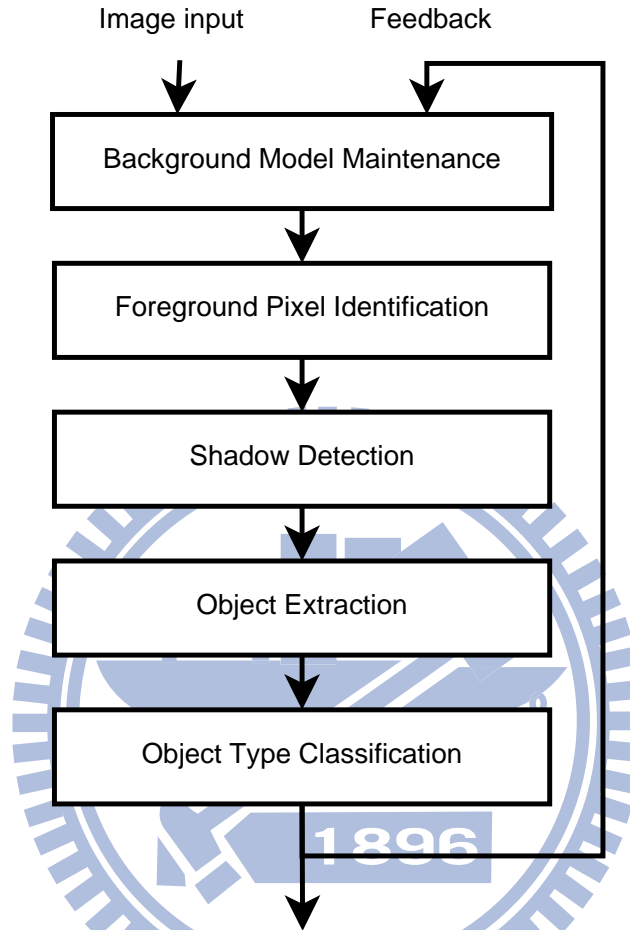


Figure 3.1: Flowchart of a general-purpose surveillance system. The first module of background model maintenance corresponds to the Algorithm 2. The second one of foreground pixel identification is implemented by using the mixture weight thresholding discussed in Sec. 3.2.1. The third module can be realized by using a shadow detection algorithm described in [6]. For object extraction, we mark small ($< 4 \times 4$ pixels), isolated foreground regions as noises via morphological processing and group the rest foreground pixels into objects by connected component analysis. Regarding the object type classification and the feedback control on learning rates, they are presented in Sec. 3.2.2.

discussions.

In the proposed approach, we adopt different learning rate settings for four object types of background, shadow, still foreground and moving foreground, respectively. Based on the processing flow of Fig. 3.1, the object types of background, shadow and foreground can be easily discriminated. To further classify the foreground type into still and moving ones, the object tracking algorithm presented in [6] is adopted to find the temporal associations among objects of time instances t and $t - 1$. Then, the position displacements of tracked objects are thresholded for discrimination of still and moving types. Thus, an object type indicator for every pixel at time instance t can be defined as

$$O_{t,\mathbf{x}} = \begin{cases} 0 & \text{if } F_{t,\mathbf{x},l(t,\mathbf{x})} = 0, \text{ (Background)} \\ 1 & \text{if } F_{t,\mathbf{x},l(t,\mathbf{x})} = 1 \text{ and Type}(I_{t,\mathbf{x}}) = \text{Shadow,} \\ 2 & \text{if } F_{t,\mathbf{x},l(t,\mathbf{x})} = 1 \text{ and Type}(I_{t,\mathbf{x}}) = \text{Still foreground,} \\ 3 & \text{Otherwise. (Moving foreground)} \end{cases}$$

and an object map can be denoted by $\mathcal{O}_t = \{O_{t,\mathbf{x}} | \forall \mathbf{x}\}$. Subsequently, the object map \mathcal{O}_t is sent back to the background model maintenance module for the learning rate control at the next time instance. This process can be regarded as a delayed feedback control since the current learning rates are calculated based on the previous estimations of pixel types. The above one-frame delay in feedback control works well in practice because the previous type estimations often provide reasonable guesses of the current pixel types if the frame rate is high with respect to foreground movements. In addition, since the feedback control is applied to the learning rate $\eta_{t,\mathbf{x}}$, but not to the mixture weights $w_{t,\mathbf{x},n}$ directly, dramatic changes

in mixture weights as pixel type varies can be avoided. Stable foreground and background separation (via weight thresholding) can thus be obtained.

With the above notations, the learning rate $\eta_{t,\mathbf{x}}$ can now be specified by

$$\eta_{t,\mathbf{x}}(\boldsymbol{\beta}) = \begin{cases} (1 - \beta_b) \eta_{t-1,\mathbf{x}} + \eta_b \beta_b & \text{if } O_{t-1,\mathbf{x}} = 0, \\ \beta_d \mathcal{N} \left(I_{t,\mathbf{x}}; \mu_{t-1,\mathbf{x},b(t,\mathbf{x})}, \sigma_{t-1,\mathbf{x},b(t,\mathbf{x})}^2 \right) & \text{if } O_{t-1,\mathbf{x}} = 1, \\ \beta_s & \text{if } O_{t-1,\mathbf{x}} = 2, \\ \beta_m & \text{if } O_{t-1,\mathbf{x}} = 3. \end{cases} \quad (3.9)$$

where η_b is a preset constant, the hyper-parameter $\boldsymbol{\beta} = [\beta_b \ \beta_d \ \beta_s \ \beta_m]^T \in \mathbb{R}^4$ is extended to a vector for controlling the learning rate with respect to different pixel types, and the index of the most probable background model, $b(t, \mathbf{x})$, is defined by

$$b(t, \mathbf{x}) = \arg \max_{n=1,\dots,N} w_{t,\mathbf{x},n}.$$

For a pixel of moving foreground ($O_{t-1,\mathbf{x}} = 3$), one may set $\beta_m \sim 0$ to suppress the adaptation of all moving objects into background, resulting in a very sensitive system to motions. In contrast, by setting β_m to a large value, which results in a quick increase of the weight of a Gaussian model for, say, a waving tree, a system will be more capable of tolerating background variations. On the other hand, for the type of still foreground, the larger the β_s is set, the quicker a stationary object will be merged into background. For the application of abandoned and missing object detection, a small β_s is preferred. Regarding the case of shadow type, we favor faster adaptation of fainter shadows into background, so $\mathcal{N}(\cdot)$ is used to estimate the similarity between the shadow intensity and the Gaussian model of

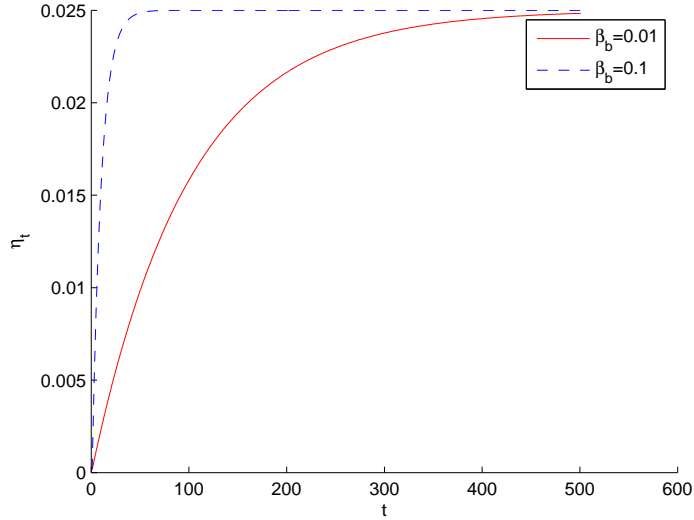


Figure 3.2: Simulated changes of the learning rate η_t for a pixel being persistent background, given $\beta_b = 0.01$ (solid line) and $\beta_b = 0.1$ (dotted line), respectively. The initial learning rate $\eta_{t=0}$ is set to $1/6000$ and η_b is set to 0.025 .

the most probable background (indexed by $b_{t,x}$). The corresponding learning rate is then set to the similarity measure multiplied by a regularization scalar β_d .

For a pixel of background type, i.e., $O_{t-1,x} = 0$, its learning rate is designed to be gradually increased at a rate regularized by β_b , as formulated in (3.9). The learning rate for an image pixel being persistently identified as background will asymptotically approach η_b , as shown in Fig. 3.2. However, once this pixel position being occluded by shadows or moving objects, the respective learning rate will be reset to other value, e.g., β_m , that is much smaller than it was used to be. This design helps preventing false inclusion of *afterimages* left by moving objects into background. When taking pictures of moving objects, their boundaries are often in blur. See Fig. 3.3 for an example. Some motion-blurred regions near object boundaries may be misclassified as background, resulting in afterimages. For an object hovering over a region, its afterimages appear frequently and will be quickly



Figure 3.3: Example of motion blur. The foreground and background boundaries of a moving hand may not be clearly distinguished, even by human visual inspection.

included into a background model. To alleviate such a problem, instead of setting the learning rate to a constant, i.e.,

$$\eta_{t,\mathbf{x}} = \eta_b, \quad \text{if } O_{t-1,\mathbf{x}} = 0, \quad (3.10)$$

it is increased gradually for a pixel of background in the proposed approach. In Sec. 3.3.1, benefits of adopting this *background-type rate control* will be demonstrated. Note that, in all our experiments, we set $\eta_b = \alpha$, $\beta_b = 0.01$, $\beta_d = 1/100$, $\beta_s = 1/900$ and $\beta_m = 1/6000$.

As discussed in [4] and [13], a major *problem with feedback control* for background modeling is that misclassifications of pixel type in the current frame will propagate to subsequent frames as the learning rates are determined by classification results. For instance, if a background pixel is misclassified as foreground, a false positive will persist at this pixel location for a long time due to the low learn-

ing rate setting for foreground pixel. Fortunately, this problem can be treated, if not cured, by the proposed framework of bivariate learning rate control.

Based on our observations, the *problem with feedback control* for background modeling can be effectively treated if the following two criteria fulfilled: (a) accurate estimation of a background model and (b) prevention of background adaptation to pixels of misclassified types. In the proposed approach, giving separate controls to the learning rates ρ and η meets the criterion (a). Up-to-date model estimations can hence be delivered by setting a large ρ , regardless of foreground classification results controlled by η . Even for the pixels of misclassified types, their Gaussian models can still be accurately estimated. Our experiments in Sec. 3.3.5 show that the accurate estimation of background models will help reducing persistent false positives of misclassified pixels.

Regarding the criterion (b), the background-type rate control in (3.9) is designed for it. With this control, false background adaptation to foreground motion blurs (a.k.a. afterimages) can be largely reduced, as will be shown in Sec. 3.3.1. In addition, the weight-based matching rule is utilized in our approach to eliminate false positives even more. Although the matching rule seems to prefer the most-weighted Gaussian models of background for new pixel observations, its matching results are still trustworthy owing to our capability of deriving accurate Gaussian models. Advantages of adopting this weight-based matching rule will be further demonstrated in Sec 3.3.5.

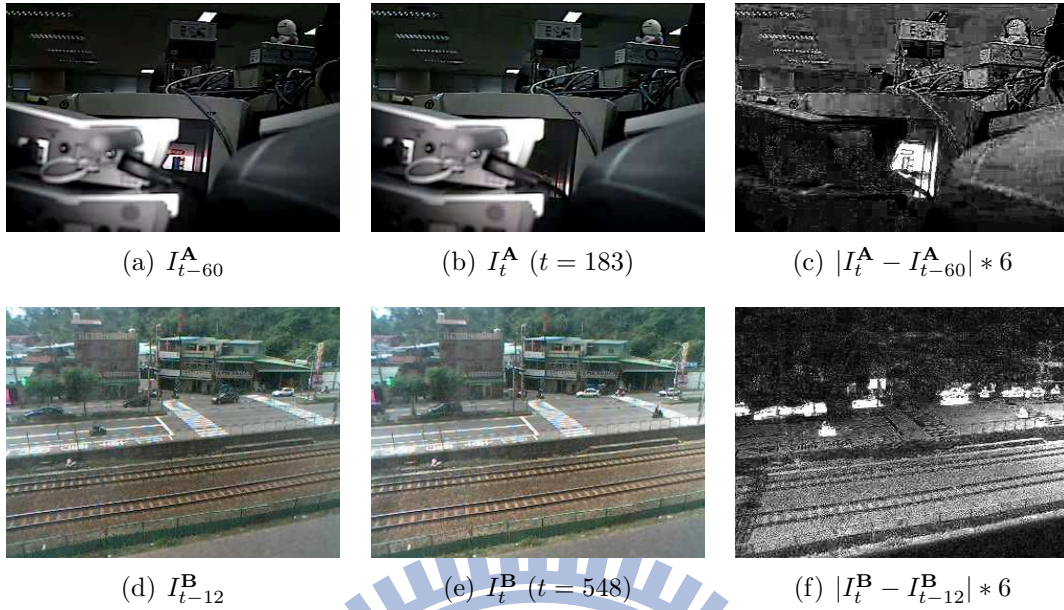


Figure 3.4: Examples of (i) quick and (ii) double-quick lighting changes. (a)-(c) Two images in Seq. **A** (recorded at 20 fps) and their difference for (i). (d)-(f) Two images in Seq. **B** (recorded at 15 fps) and their difference for (ii).

3.2.3 Heuristic for Adaptation of Double-Quick Lighting Change

Surveillance systems often encounter challenges from lighting changes, especially for systems used in outdoor environments. While gradual and quick lighting variations can often be adapted by the GMM, some *double-quick* changes can not be caught via background model learning at reasonable learning rates. For instance, two examples of quick and double-quick lighting changes are given in Fig. 3.4. The image sequence **A** shown in Figs. 3.4 (a)-(c) records a laboratory with a monitor displaying rolling interferences. In this indoor sequence, it takes about 3 seconds to increase the average intensity by 20%. This quick variation in image brightness can still be learned by the GMM, as will be demonstrated in Sec. 3.3.1. In contrast,

for a double-quick lighting change shown in Figs. 3.4 (d)-(f), similar increases of image intensity are observed in less than one second for an outdoor environment. As will be shown in Sec. 3.3.3, many false alarms in foreground detection are issued under such condition. Consequently, a heuristic based on frame difference is also developed to assist the GMM to cope with double-quick lighting changes.

The idea behind the heuristic is simple yet effective. While image intensity variation of double-quick lighting change may seem to be large among temporally distant image frames, it may be small between two consecutive frames if the frame rate of recording is high enough. The small and smooth change of image brightness between consecutive image frames provides a cue for eliminating false alarms in foreground detection for double-quick, but not *abrupt*⁶, lighting changes. For example, by thresholding the differences between corresponding pair of pixels, each from two consecutive frames, at a proper level, such false alarms can often be reduced.

Accordingly, the proposed heuristic consists the following formulations. First, the thresholding of intensity difference for every pixel pair is performed by

$$D_{t,\mathbf{x}} = \begin{cases} 1 & |I_{t,\mathbf{x}} - I_{t-1,\mathbf{x}}| > T_d, \\ 0 & \text{otherwise.} \end{cases}$$

where $T_d(= 10)$ is a given threshold. Thus, a frame difference map $\mathcal{D}_t = \{D_{t,\mathbf{x}}|\forall\mathbf{x}\}$ can be derived. By combining both the frame difference map \mathcal{D}_t and the foreground map \mathcal{F}_t via

$$\mathcal{F}'_t = \mathcal{F}_t \text{ AND } (\mathcal{F}'_{t-1} \text{ OR } \mathcal{D}_t), \quad (3.11)$$

⁶Abrupt changes in background are regarded as salient deviations between two consecutive image frames, due to, e.g., light on/off.

a new foreground map \mathcal{F}'_t being less affected by lighting changes can now be obtained. Note that the OR operation in (3.11) is utilized for temporal accumulation of foreground regions, which is useful for detecting objects in slow motion. The map \mathcal{F}'_t is then used to replace \mathcal{F}_t as a new output of the second module in Fig. 3.1. Regarding the lighting change areas where $\mathcal{F}'_t - \mathcal{F}_t \neq 0$, they are relabelled as background and will be quickly learned by the GMM via (3.9). False alarms caused by double-quick lighting changes will hence be reduced. Based on our experiments shown in Sec. 3.3.3, the system robustness to lighting changes will be increased without losing the sensitivity in detecting significant foreground motions.

Because this heuristic is developed to improve the tolerance of our model to speedy lighting changes without altering the background estimation results much, the threshold value is usually limited by $10 \leq T_d \leq 20$. Image differences larger than 20 between two consecutive image frames, which might be perceived by sensitive human eyes, are considered as abrupt changes. Owing to the accumulating formulation in (3.11), large lighting changes between two distant image frames can still be handled using small T_d for most cases.

3.3 Experimental Results

Several real videos are used to test the effectiveness of the proposed bivariate learning rate control scheme. In Sec. 3.3.1, comparisons among different learning rate controls proposed by the original GMM [54], its variant [36] and this research are presented⁷ using two image sequences with lighting changes, missing

⁷For experimental evaluations, we apply the conventional matching rule (3.1) to [36] and [54], and use the same labeling rule (3.8) with $T_w = 0.24$ to all the methods to segment foreground regions.

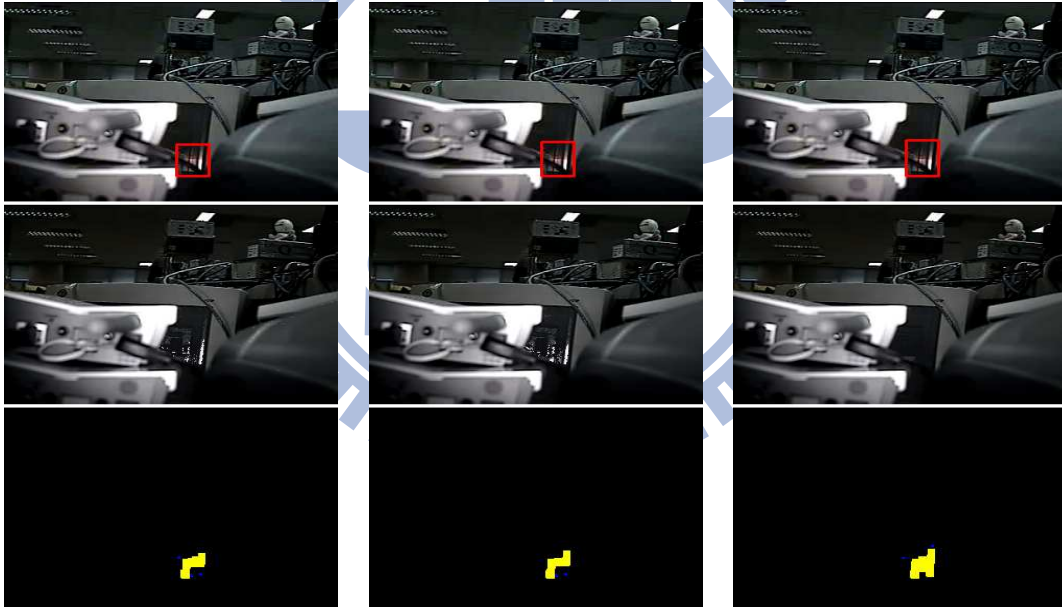
objects and waving hands. While the first scenario of lighting changes should be quickly adapted into background, the other two should not. All these scenarios can be properly handled by the proposed approach but not by those of [54] and [36]. In Sec 3.3.2, the effects of tuning the parameter β are discussed. Next in Sec 3.3.3, by using a third image sequence as a benchmark, the superiority of the proposed heuristic for adaptation of double-quick lighting change is demonstrated. In Sec. 3.3.4, quantitative evaluations of selected approaches with respect to different α values are presented. In Sec. 3.3.5, an example of fountain spurt is used to demonstrate our treatments of the *problem with feedback control* for background modeling. Finally, additional experimental results are given to show the effectiveness of the proposed approach for the scenes of *waving water* and *crowded entrance*.

3.3.1 Regularized Background Adaptation

In the first experiment for the adaptation of quick lighting changes, we use Seq. **A** previously illustrated in Fig. 3.4 as a benchmark. The foreground detection results and the learned background models, up to the image frame I_t^A , obtained from different approaches are shown in Fig. 3.5 for two different learning rates. For visual comparisons of the learned background models, a definition of background map $\mathcal{B}_t = \{\mu_{t,\mathbf{x},b(t,\mathbf{x})} | \forall \mathbf{x}\}$ is adopted, and the derived background maps are drawn in the middle row of Fig. 3.5. In Figs. 3.5 (a) and (b), false positives of foreground detection are observed by using $\alpha = 0.01$ for the approaches of [54] and [36]. As shown in Figs. 3.5 (d) and (e), all the false positives can be eliminated by giving a higher learning rate with $\alpha = 0.025$ while only the rolling interferences



(a) Results of [54] ($\alpha = 0.010$) (b) Results of [36] ($\alpha = 0.010$) (c) Our results ($\alpha = 0.010$)



(d) Results of [54] ($\alpha = 0.025$) (e) Results of [36] ($\alpha = 0.025$) (f) Our results ($\alpha = 0.025$)

Figure 3.5: Comparisons of background adaptation to quick lighting changes using Seq. A. Top row: foreground detection results for I_t^A ; middle row: computed background maps \mathcal{B}_t s; bottom row: derived foreground maps. In the foreground maps, the regions in blue denote shadows and noises. (a), (b), and (c) The results of [54], [36], and our approach, respectively, with $\alpha = 0.010$. (d), (e), and (f) The results of [54], [36], and our approach, respectively, with $\alpha = 0.025$.

on a monitor are marked as foreground. On the other hand, correct foreground detection results are obtained in Figs. 3.5 (c) and (f) by the proposed approach (with the heuristic of (3.11) applied) for both rate settings.

In the previous experiment, $\alpha = 0.025$ can be regarded as a proper setting for adaptation of quick lighting change. However, if the same setting is used for Seq. C, defects of foreground detection will appear for approaches of [54] and [36]. (Because the foreground detection results of [54] and [36] in this experiment are almost the same, only those of [36] are shown in Fig. 3.6 for brevity.) As shown in Fig. 3.6 (a), a cellular phone on a desk is taken away. Usually, a missing personal property should be marked as foreground and trigger an alarm. However, such an abnormal event can not be stably detected with $\alpha = 0.025$. The quick adaptation of the uncovered region into background happens in about one second, as shown in Fig. 3.6 (b), leaving no evidence of the missing cellular phone. Similarly, hand waving in front of the camera is soon adapted into background as well, as shown in Fig. 3.6 (c), causing the hand regions only partially detected. In contrast, the above two scenarios can be properly handled by the proposed approach with the same parameter setting ($\alpha = 0.025$), as shown in Figs. 3.6 (d)-(f). Thanks to the regularization of the learning rate η , quick lighting changes, missing objects and periodic motions can all be modeled decently in an unified framework.

Advantages of the proposed background-type rate control are also demonstrated, using Seq. C, in Fig. 3.7 wherein background modeling results are obtained with and without the background-type rate control. By replacing the gradual increase of background learning rate in (3.9) with a constant setting of (3.10), as can be seen in Fig. 3.7 (a), the afterimages induced by the waving hand are included into the background model (the second row) and the resulted segmentation of fore-

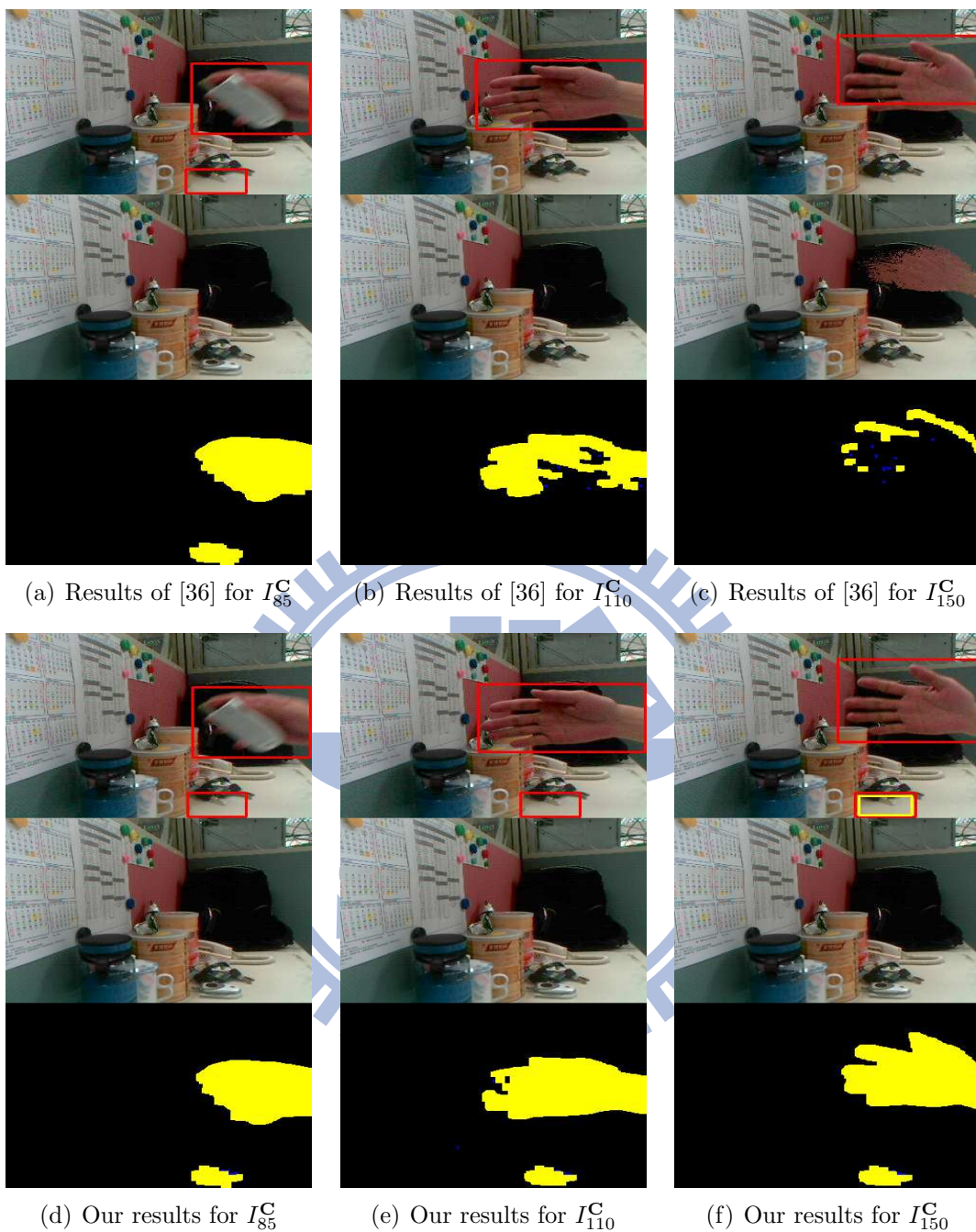
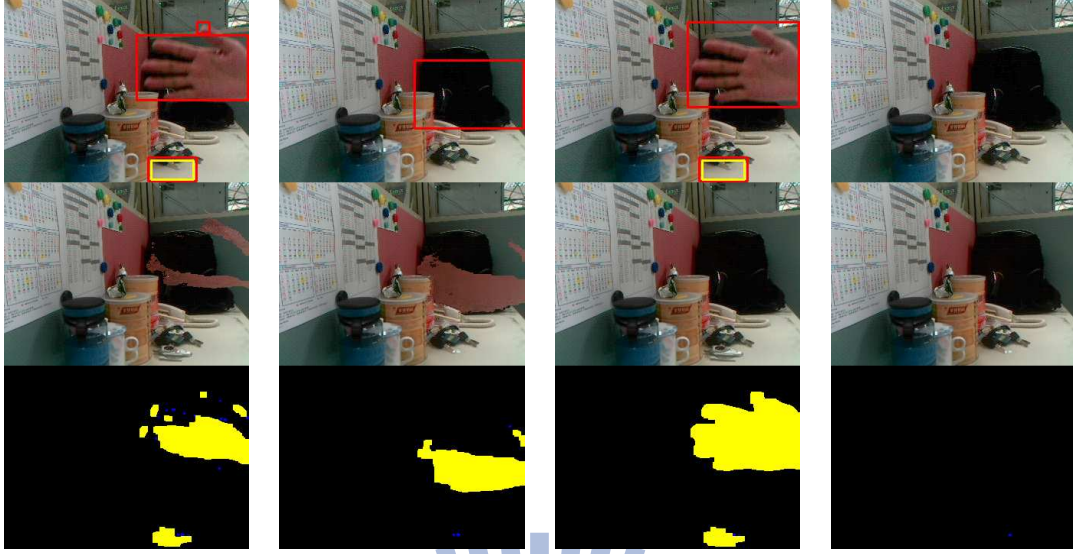


Figure 3.6: Comparisons of background modeling for missing object and waving hand using Seq. C. Top row: foreground detection results; middle row: computed background maps; bottom row: derived foreground maps. (a), (b), and (c) The results for I_{85}^C , I_{110}^C , and I_{150}^C , respectively, using [36] with $\alpha = 0.025$. (d), (e), and (f) The results for I_{85}^C , I_{110}^C , and I_{150}^C , respectively, using our approach with $\alpha = 0.025$. In (f), the cellular phone taken away is identified as a missing object and highlighted by a yellow box.



(a) I_{365}^C (w/o BTRC) (b) I_{810}^C (w/o BTRC) (c) I_{365}^C (w/ BTRC) (d) I_{810}^C (w/ BTRC)

Figure 3.7: Comparisons of background modeling results obtained without and with using the background-type rate control (BTRC). Top row: foreground detection results; middle row: computed background maps; bottom row: derived foreground maps. (a) and (b) The results derived by replacing the first equation of (3.9) with (3.10). (c) and (d) The results derived by (3.9).

ground regions is incomplete (the third row). In Fig. 3.7 (b), as the hand moving out of the scene, the incorrect background model continues to give false positives in foreground detection for a period of time. On the other hand, such defects can be effectively reduced by using the proposed rate control for background pixels, as shown in Figs. 3.7 (c) and (d).

3.3.2 Parameter Tuning

As tuning the hyper-parameter $\boldsymbol{\beta} = [\beta_b \ \beta_d \ \beta_s \ \beta_m]^T$, the effect is more on time span of background adaptation than on the accuracy of background modeling. Specifically, varying β_s changes the time span for a still object to be merged into a background model, if no interrupt occurs. The number of required image frames

Table 3.1: Numbers of image frames resisting background adaptation to afterimages w.r.t. β_b s.

β_b	0.01	0.05	0.1	0.5	1
# of Frames	> 700	> 700	~ 430	~ 175	~ 165

to adapt a pixel of still type into background can be estimated by

$$\arg \min_t w_t > T_w \quad \text{subject to} \quad w_t = (1 - \beta_s) w_{t-1} + \beta_s.$$

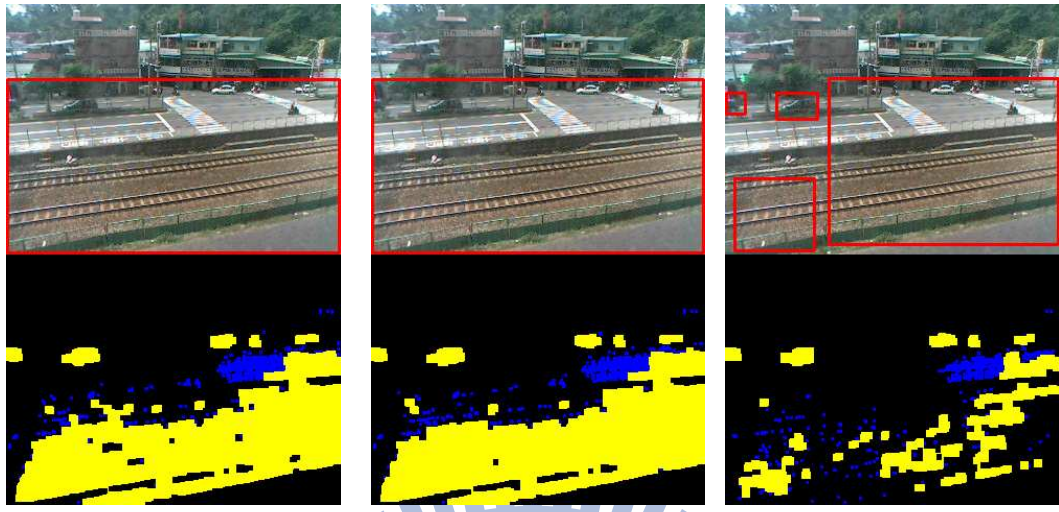
For example, given $\beta_s = 1/900$ and $w_0 = 0.01$ as an initial value, at least $t = 239$ image frames are required to complete the background adaptation of a still-type pixel. For Seq. **C** shown in Fig. 3.7, it takes about 288 frames to replace regions of the missing cellular phone with newly-revealed scenes in the background model, just a little longer than predicted. Regarding the default setting of $\beta_m = 1/6000$, at least $t = 1588$ image frames are needed for a pixel being continuously occupied by the same hovering object to be adapted into background. This number of image frames roughly matches the testing example shown in Sec. 3.3.5 where all the regions of a fountain spurt are adapted into background in about 2000 image frames.

Similarly, tuning β_b alters the time span of avoiding afterimages to be incorporated into a background model. Taking Seq. **C** as a benchmark, the numbers of image frames having no afterimage in background models under different β_b s are summarized in Table 3.1. Here setting β_b to 0.05 or less gives no obvious defects in the estimated background models throughout the sequence. On the other hand, increasing β_b and β_m may be needed for scenarios with large periodic motions, e.g., shaking tree branches and moving tides.

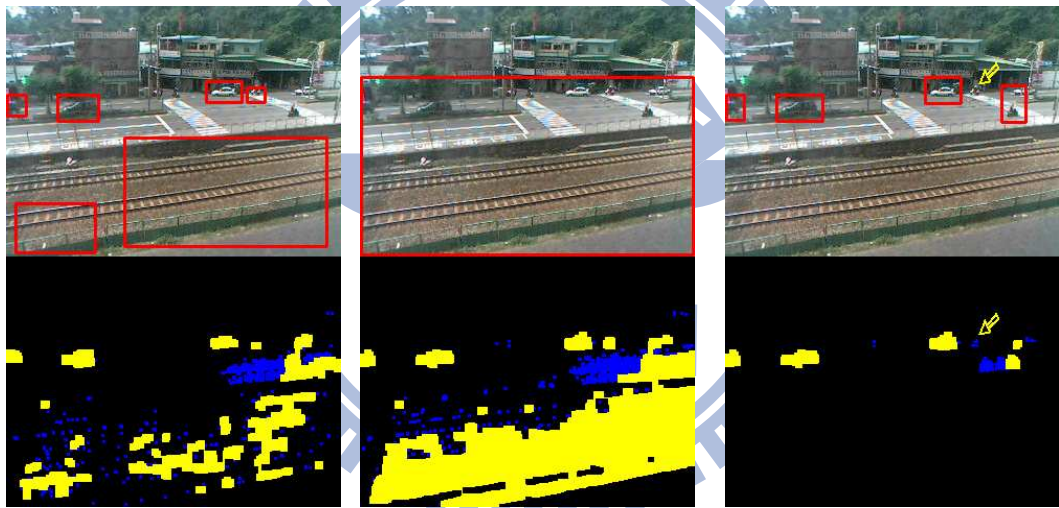
Regarding β_d , it is tuned to slightly defer the adaptation of shadows that are usually casted by foreground objects into a background model. Thus, the product $\beta_d \mathcal{N}\left(I_{t,\mathbf{x}}; \mu_{t-1,\mathbf{x},b(t,\mathbf{x})}, \sigma_{t-1,\mathbf{x},b(t,\mathbf{x})}^2\right)$ should be kept below β_b . In addition, if the product is less than β_s , it will instead be reset to β_s in our implementation, to adapt static and frequently-seen shadows into background. To sum up, via proper tuning of β , the required time spans for adapting pixel of different types into background can be easily and accurately controlled for various applications.

3.3.3 Double-Quick Lighting Change

Fig. 3.8 shows a scene experiencing double-quick sunshine changes. The resultant double-quick changes in background can not be adapted in time by the GMM framework, even by setting high learning rates, as shown in Fig. 3.8. By utilizing the proposed heuristic, with T_d set to 10, for adaptation of double-quick lighting change, almost all the false positives resulted from sunshine changes are eliminated in the entire testing sequence. Nevertheless, a few sides-effects are also observed. Fig. 3.8 (d) gives an example that a small motorcycle whose colors are similar to the background scene is misidentified as noises (marked in blue), for some parts of this object are deleted by frame difference. Through examination of these results, one can easily see that, overall, adopting such a heuristic actually brings in more benefits than drawbacks. Further quantitative evaluations, as will be presented later, also support this observation. Many false positives in foreground detection can thus be reduced while only limited false negatives are induced. Besides, large, significant motions will not be ignored by using this heuristic due to its design of foreground map accumulation via the OR operation in (3.11).



(a) Results of [54] ($\alpha = 0.025$) (b) Results of [36] ($\alpha = 0.025$) (c) Results of [54] ($\alpha = 0.050$)



(d) Results of [36] ($\alpha = 0.050$) (e) Our results ($\alpha = 0.025$, w/o heuristic) (f) Our results ($\alpha = 0.025$)

Figure 3.8: Comparisons of background adaption to double-quick lighting change using Seq. **B**. The foreground detection results for $I_t^{\mathbf{B}}$ are illustrated. (a) and (b) The results of [54] and [36], respectively, with $\alpha = 0.025$. (c) and (d) The results of [54] and [36], respectively, with $\alpha = 0.050$. (e) The results of the proposed approach without using the heuristic for adaptation of double-quick lighting change. (f) The results of the proposed approach. The yellow arrows mark the undetected foreground regions of a small motorcycle.

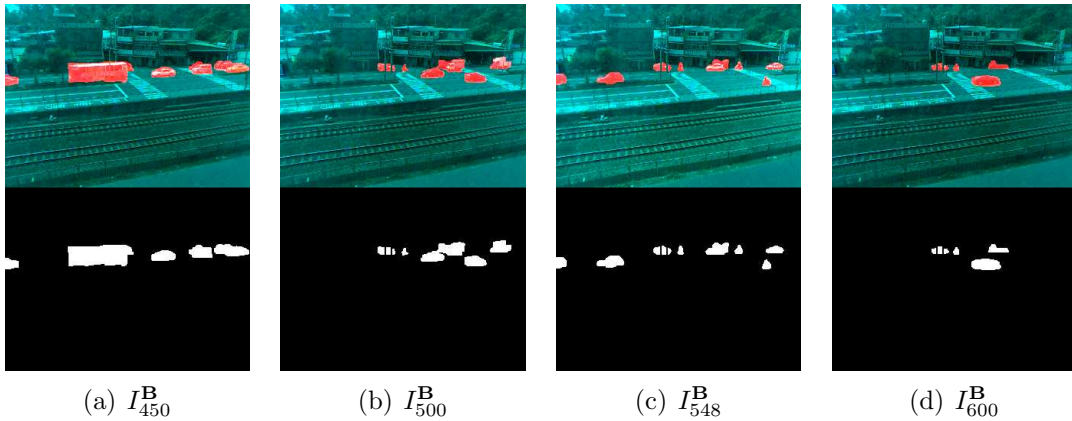
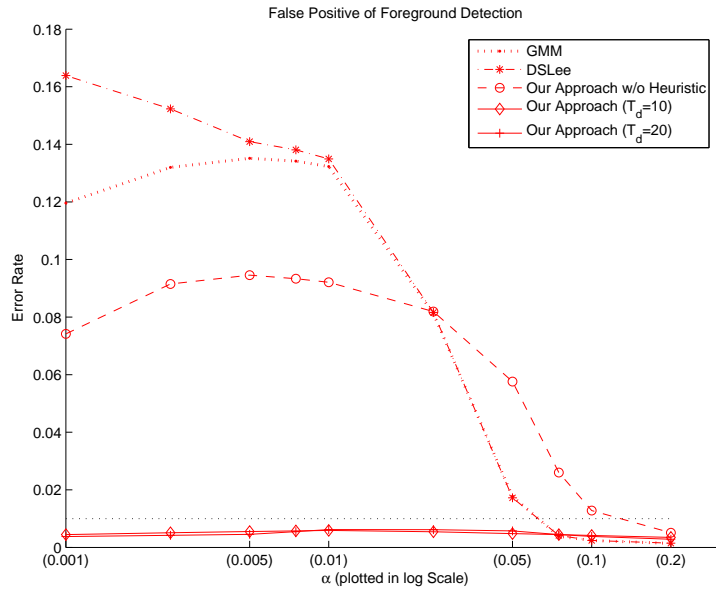
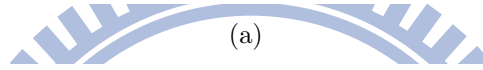
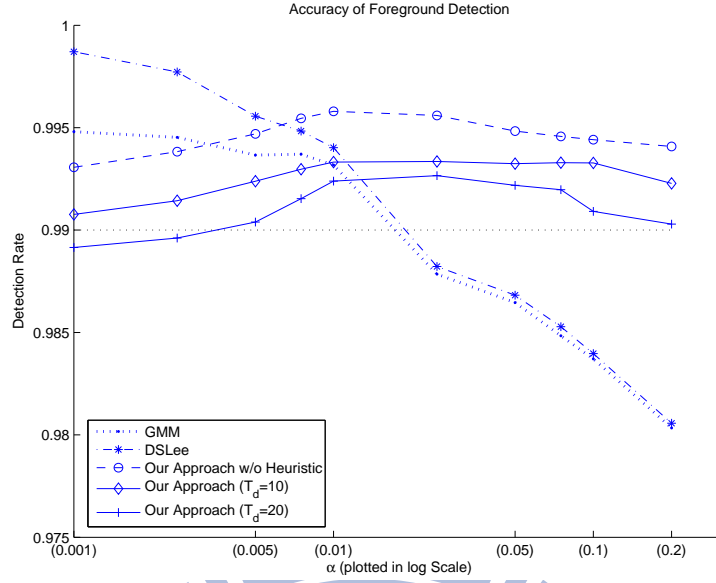


Figure 3.9: Snapshots of the ground-truth images for Seq. **B**

3.3.4 Quantitative Evaluations

In the quantitative comparisons among [36], [54], and our approach without/with the heuristic of (3.11), Seq. **B** is used as a benchmark for it is a real and challenging sequence. To construct the ground-truth data, we write a program to segment possible foreground regions of Seq. **B** with high sensitivity. Subsequently, 32 representative image frames are selected by visual inspection, and with their segmentation results refined manually. Note that all the vehicles in the scene, no matter in motion or resting, are marked as foreground in this evaluation. Snapshots of the ground-truth images are given in Fig. 3.9.

The statistical plots in Fig. 3.10 are generated by applying different α values to all the compared methods. Also, two T_d settings for our approach are included in the comparison. Results in Fig. 3.10 show that, with $T_d = 10$, the proposed approach constantly achieves low false positive rates ($< 1\%$) while keeping high detection accuracy ($> 90\%$) for all α s. If the heuristic is not used, then $\alpha = 0.2$ can be chosen for our approach to both catch the double-quick lighting changes and maintain high detection accuracy. As for the methods of [36] and [54], finding



(b)

Figure 3.10: Quantitative comparisons of [36] (DSLee), [54] (GMM), our approach without the heuristic, and our approach with $T_d = 10$ and $T_d = 20$ under different α settings using the 32 ground-truth images of Seq. **B**. Here, the values of 0.0010, 0.0025, 0.0050, 0.0075, 0.0100, 0.0250, 0.0500, 0.0750, 0.1000, and 0.2000 are set to α to generate the curves. (a) Comparisons of foreground detection rates. The detection level of 99% is marked for reference. (b) Comparisons of false positives rates in foreground detection. The false positive level of 1% is marked for reference.

a reasonably good parameter setting seems not possible for this case.

Although our approaches (with and without the heuristic) do not give the highest detection rate, they both have a feature of delivering stable detection results under various α settings, mainly owing to our independent controls of the two types of learning rates. Moreover, through examining the false positive rates with respect to different α s, we choose to bring the heuristic into our approach as a default practice since doing so will almost always give low false alarms. Note that, based on the evaluations, $\alpha = 0.01$ and $\alpha = 0.025$ can be suggested as default values for the heuristic because these values give slightly better detection accuracy.

To verify our argument that adjusting β does not affect the background modeling performance much, a quantitative evaluation is conducted by varying β_b to 0.001, 0.005, 0.010, 0.050, 0.100, and 0.500, with the other parameters fixed to the default values. While the detection and false positive rates for $\beta_b = 0.010$ (the default setting) are 99.3347% and 0.5420%, respectively, similar performance indices for the other β_b s are all within $(99.3347 \pm 0.0274)\%$ and $(0.5420 \pm 0.0199)\%$, respectively, which supports our argument.

3.3.5 Scene Change

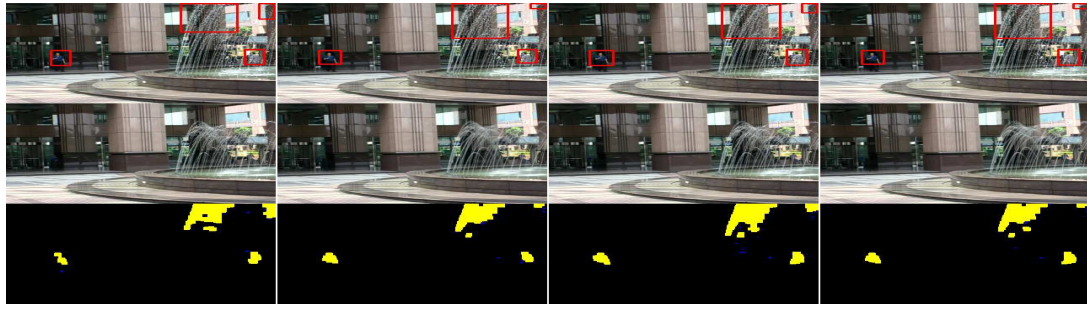
In Sec. 3.2.2, the *problem with feedback control* and possible solutions are discussed. An example illustrating such a problem is given in Fig. 3.11, where a fountain suddenly spurting high causes a bunch of false positives in foreground detection. The first column of Figs. 3.11 shows such dramatic changes of background scene may be adapted too quickly (in about 100 image frames) by [36] if a high learning rate ($\alpha = 0.025$) is used. On the contrary, as shown in the second column, these false

positives last for a very long time (> 2440 image frames) if a naive feedback control by setting $\rho(\alpha) = \eta(\beta)$ is used. The identical modeling of ρ and η , together with the feedback controls, makes a system behave as what the problem describes. However, as depicted in the third column of Figs. 3.11 (c) and (d), quicker adaptation of false positives into background can be achieved by separating the control of ρ from that of η with the conventional model matching rule of (3.1). Finally, as shown in the forth column, the false positives resulted from scene changes can be completely eliminated in about 2000 image frames (equivalent to about 1.11 minutes for a 30 fps video) by combining the proposed bivariate rate control scheme with the weight-based model matching rule.

3.3.6 Other Scenarios

Additional experiments for the scenes of waving water⁸ and crowded entrance are demonstrated in Fig. 3.12. In Fig. 3.12 (a), a floating bottle on the waving water can be successfully detected by the proposed approach. In the crowded entrance sequence shown in Fig. 3.12 (b), a black bag left by a passenger is stably detected as a foreign object in a busy scene. These experiments show the effectiveness of the proposed scheme of bivariate learning rate control for the surveillance applications associated with complex scenes.

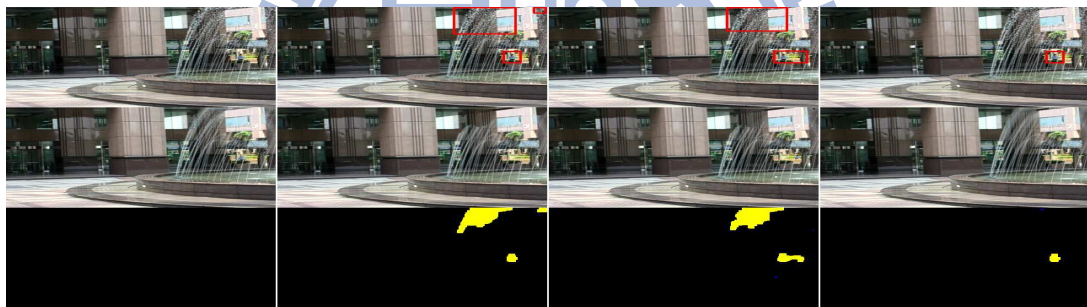
⁸The image sequence of waving water is from [68].



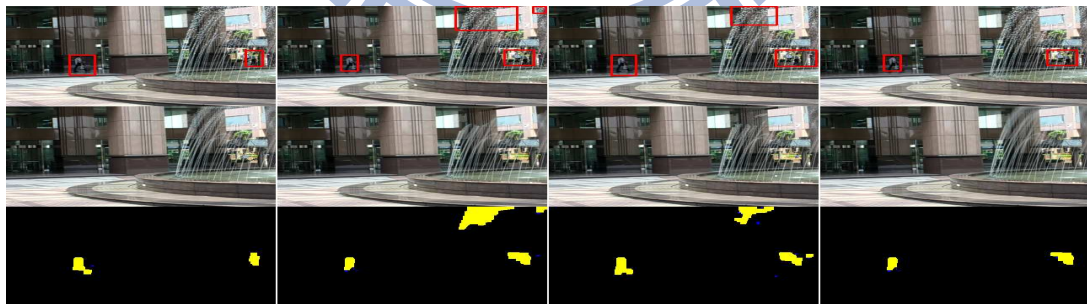
(a) I_{2900}



(b) I_{3000}



(c) I_{4885}



(d) I_{5340}

Figure 3.11: Comparisons of scene change adaptation among [36] (the 1st column), feedback control with $\rho(\alpha) = \eta(\beta)$ (the 2nd column), our approach with (3.1) (the 3rd column) and our approach with the weight-based model matching rule (the 4th column).



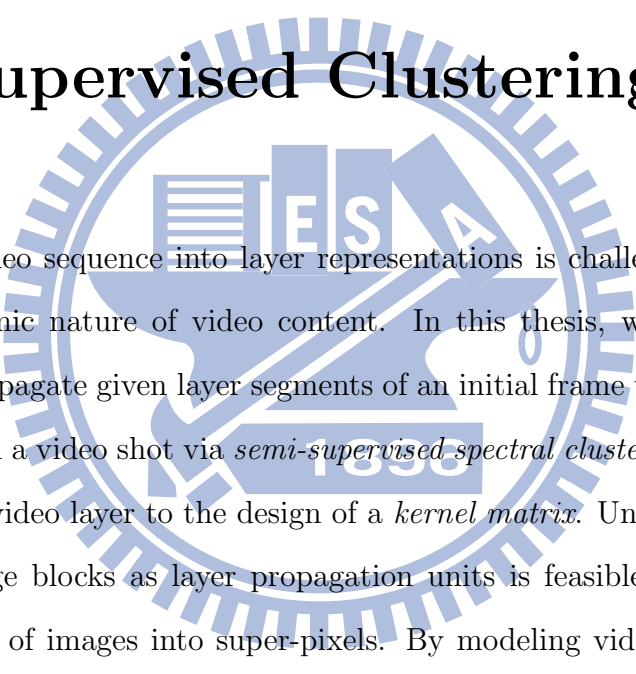
(a) Results for waving water [68]

(b) Results for crowded entrance

Figure 3.12: Foreground detection (top row) and background modeling (bottom row) results for the scenes of (a) waving water and (b) crowded entrance. The yellow box in (b) marks an abandoned bag.

Chapter 4

Video Layer Propagation via Semi-Supervised Clustering



Segmenting a video sequence into layer representations is challenging because of the highly dynamic nature of video content. In this thesis, we propose a new framework to propagate given layer segments of an initial frame to subsequent image frames within a video shot via *semi-supervised spectral clustering*, and link the segmentation of video layer to the design of a *kernel matrix*. Under such a framework, using image blocks as layer propagation units is feasible, avoiding costly pre-segmentation of images into super-pixels. By modeling video layer propagation between consecutive image frames as a label inference problem wherein new block labels are inferred from previous known ones, and by solving this problem via semi-supervised spectral clustering, video layers are progressively propagated. Experimental results show the proposed approach can track and segment video layers effectively, even for those undergoing large motion and deformation.

4.1 Overview

Decomposing video content into layer representations, e.g., foreground and background region segments, is beneficial for video understanding and coding. For example, extracted foreground layers can be used for efficient video searching and for better data compression. However, due to the highly dynamic nature of video content, the problem of video layer decomposition is challenging and remains an important research topic in computer vision.

Rather than decomposing video layers from scratch, a more restricted problem, video layer propagation, is investigated in this study. Given a video shot and the layer segments of its initial image frame, the goal of video layer propagation is to iteratively propagate the corresponding video layers to the subsequent image frames. Except for the initial layer information, no other assumptions on foreground appearances, and on background scenes, or restrictions with respect to camera motions are made. This makes the problem difficult because, for example, a background layer of a video shot may be cluttered and undergo large changes due to camera movements. It is thus hard to achieve reasonable decomposition of subsequent, time-varying layer segments by simply applying supervised learning methods to learn prior models from initially-given layers. Instead, we formulate the video layer propagation problem as a series of semi-supervised clustering problems to effectively capture the dynamic changes of video layers.

Specifically, propagating video layers from a previous image frame I_{t-1} to the current one I_t is regarded as a process of label inferencing, where the unknown layer labels of the image elements in I_t are inferred from the known ones in I_{t-1} . (An image element here can be embodied as a pixel, a block or a region.) Such

an inference process fits the formulation of semi-supervised clustering [69] very well. By clustering the image elements in I_t into the layer classes of I_{t-1} , new video layers can be obtained. Through a series of clustering processes, initial video layers can be propagated progressively to capture continuous layer changes. Experimental results show that the proposed approach can track and segment video layers effectively, even when they undergo large, non-rigid motions.

The proposed approach for video layer propagation has several features. First, a novel framework based on semi-supervised clustering is proposed for propagating video layers. In particular, inferring new layer labels from previously determined ones is linked to the construction of a *kernel matrix* in such a framework, which provides a new perspective to the problem of video segmentation. Second, the effectiveness of using image blocks as basic processing units for video layer propagation is demonstrated. Such a choice provides more information than using pixels in layer discrimination while avoids the costly pre-segmentation of image regions/super-pixels. Third, a new regularization scheme for controlling the reliability of block labels is developed. A block near layer boundaries often consists of more than one video layer and is less appropriate to be categorized into a single layer class. Hence the reliability of its layer label needs to be regularized as being propagated to the next image. Such control of individual label trustworthiness is novel and can be linked to the design of a *regularized kernel*. Fourth, since only layer labels need to be processed in the proposed layer propagation framework, optional user interventions for amending layer clustering defects can be incorporated into the propagation process easily. Thus, a very small amount of flawed layer labels that might later induce large layer segmentation errors can be corrected handily at an early stage by manually re-labeling.

4.1.1 Related Work

Previous research on video layer representation and/or segmentation often assumes that every video layer is planar and each can be associated with a distinct planar motion model. Based on the assumption, several studies of motion segmentation, e.g., in [3], [10], [28], [60], [61], [62], are proposed to estimate layer motions and to extract video layers. Particularly, in [61], [62], spatial coherence of layer labels is explicitly modeled and combined with layer motion estimation for stable layer segmentation. In [57], Torr *et al.* develop a Bayesian formulation for video segmentation in which parallax disparities of video layers are incorporated. Based on homographic projection, Ke and Kanade [31] propose a subspace clustering approach to group image patches into video layers. In [17], [29], [30], an image is modeled as a mixture of planar sprites that undergo different motion transformations. Sprite layers can then be extracted by optimizing the mixture formulation using EM-based techniques. Further, Aguiar and Moura [1] apply a rigidity constraint on objects to the mixture representation of layers for figure-ground separation.

Recently, video layer segmentation based on graph cuts has drawn much attention. An early work proposed in [52] uses normalized cuts to segment video layers in consistent motion. In [5], Boykov *et al.* apply graph min-cuts to approximate the solution of an energy function in general form and prove that the approximated solution is near the global optimum. Based on [5], several methods are then proposed to extract video layers using different cues, e.g., motion fields [63], gradients [7], and stereo [34]. In [38], Li *et al.* interpolate video layers between key frames via graph cuts and introduce feature tracking for the refinement of layer segmentation. In [8], a feature fusion formulation that combines motion,

color and contrast cues is proposed for bi-layer segmentation and can be solved by graph cuts in real-time. In addition, occlusion orders [65], layer rigidity [15], and learned layer filters [67] have also been integrated into the graph cut-based segmentation of video layers. Besides, solving layer mixture models via graph cuts is explored in [35]. According to our investigation, the proposed approach, though also originated from similar graph-cut concepts, has novelties in the adoption of semi-supervised clustering and in a link to kernel design.

4.2 Video Layer Propagation Framework

The main idea behind the proposed framework for video layer propagation is that propagating video layers between two consecutive image frames can be regarded as a label inference process. To estimate new layer labels, an algorithm of *semi-supervised spectral clustering* [69] is adopted. The advantages of choosing this algorithm include its adoption of prior labels in a natural way, the existence of a closed-form solution, and its connection to the kernel methods in machine learning. In addition, a generalization of the adopted algorithm that regularizes the reliability of layer labels is developed. This generalized regularization scheme not only provides better control over prior labels in theory, but also benefits our design of block-based layer propagation in practice.

4.2.1 Block Label Inference

The inference of video layer labels can be performed on various image elements, e.g., pixels, regions, or blocks. In this study, we explore the feasibility of using block units in layer propagation, which does not require costly pre-segmentation of

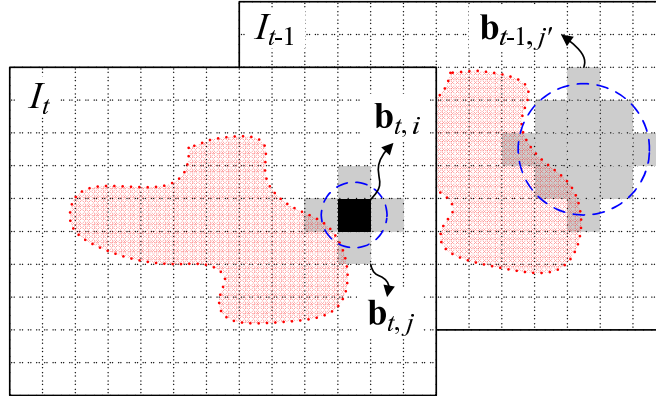


Figure 4.1: An illustration of the spatio-temporal neighbors of a block $\mathbf{b}_{t,i}$ (colored in black). The gray blocks whose centers are within the small and the large dashed circles are regarded as the spatial and temporal neighbors of $\mathbf{b}_{t,i}$, respectively. All the neighboring blocks are linked to $\mathbf{b}_{t,i}$ by edges.

images into small, homogeneous regions. Thus, the layer propagation results will not be affected by the stability of region segmentation. Also, block units provide more discriminability than pixels in inferring layer labels. However, an image block has resolution limitation in differentiating layer boundaries. An image block near layer boundaries may have multiple video layers and is less appropriate to be categorized into a single layer class. Hence, a regularization scheme of block label reliability, especially for those blocks near layer boundaries, is proposed as a remedy.

Consider two consecutive image frames, I_t and I_{t-1} , which are decomposed into image blocks of size $w \times w$, as shown in Fig. 4.1. Let l denote the total number of the image blocks in the two frames, and $\mathbf{b}_{t,i}$ denote the i -th image block in the image I_t . Each of its spatial (temporal) neighbors of the image block $\mathbf{b}_{t,i}$, denoted by $\mathbf{b}_{t,j}$ ($\mathbf{b}_{t-1,j'}$), is the block whose center is inside the small (large) circle. The radii of the spatial (small) and temporal (large) circles are denoted by r_{spa} and r_{tem} , respectively. The respective block labels are symbolized by $y_{t,i}$, $y_{t,j}$ and

$y_{t-1,j'}$, where y is in a label set $\mathcal{L} = \{1, \dots, C\}$ and $C = 2$ accounts for foreground and background layers. While $y_{t,i}$ and $y_{t,j}$ are unknown, $y_{t-1,j'}$ is assumed to be known, either from manual labeling or from previous layer estimation. Similarly, the spatio-temporal neighbors of every image block in I_t can also be defined. All the image blocks and their neighboring relationships can be conveniently denoted by an undirected graph $\mathcal{G}(\mathcal{E}, \mathcal{V})$, where \mathcal{V} and \mathcal{E} stand for the set of l block nodes and the set of all the neighboring links, respectively. We use $v_i \sim v_j$ to denote that two neighboring nodes v_i and v_j are linked by an edge. Now, our goal is to compute the new layer labels, $y_{t,i}$ s, based on \mathcal{G} .

To estimate the block label $y_{t,i}$, both the spatial and temporal neighbors of $\mathbf{b}_{t,i}$ should be considered. Intuitively, not only the temporal similarities among $\mathbf{b}_{t,i}$ and all $\mathbf{b}_{t-1,j'}$ s can be used to determine a probable label, but also the spatial similarities among $\mathbf{b}_{t,i}$ and all $\mathbf{b}_{t,j}$ s should also be consulted to maintain spatial consistency. To obtain a balanced solution, strategies that regularize both spatial and temporal similarities are often used. In the adopted semi-supervised spectral clustering, the optimal layer labeling is achieved through the diffusion of similarities. The similarities initially defined on \mathcal{E} are spread into neighboring node pairs, resulting in a new edge set $\tilde{\mathcal{E}}$. This step corresponds to the construction of a kernel matrix for $\tilde{\mathcal{G}}(\tilde{\mathcal{E}}, \mathcal{V})$ from a similarity matrix for \mathcal{G} . The most probable label $y_{t,i}$ can then be found based on the diffused similarities. As will become clearer later, the spatio-temporal consistency among layer labels can thus be ensured. By deriving all the layer labels, the propagation of video layers is realized.

4.2.2 Semi-Supervised Spectral Clustering

Let $\hat{\mathbf{y}}_i \in \mathbb{R}^C$ denote the initially-given label of a graph node v_i . Specifically, for the case of $C = 2$, we set

$$\hat{\mathbf{y}}_i = \begin{cases} [1 \ 0]^T & \text{if } v_i \text{ belongs to background;} \\ [0 \ 1]^T & \text{if } v_i \text{ belongs to foreground;} \\ [0 \ 0]^T & \text{if the label of } v_i \text{ is unknown.} \end{cases} \quad (4.1)$$

A prior label matrix $\hat{\mathbf{Y}} \triangleq [\hat{\mathbf{y}}_1 \ \hat{\mathbf{y}}_2 \ \dots \ \hat{\mathbf{y}}_l]^T \in \mathbb{R}^{l \times C}$ is then introduced to denote all the initial labels of the nodes in \mathcal{G} . By applying the semi-supervised spectral clustering shown in Algorithm 3, a new label estimate $\hat{\mathbf{Y}}^*$ that encodes the optimal solutions of the unknown labels is derived. The algorithm, originally proposed by Zhou *et al.* [69] and generalized in this study, is basically an adaptation of the spectral clustering method [44]. However, unlike the spectral clustering which uses only a similarity matrix, the semi-supervised modification takes both similarities and prior labels into consideration. Zhou *et al.* presented an iterative equation as

$$\mathbf{Y}_k = (1 - \alpha)\hat{\mathbf{S}}\mathbf{Y}_{k-1} + \alpha\hat{\mathbf{Y}} \triangleq \beta\hat{\mathbf{S}}\mathbf{Y}_{k-1} + \alpha\hat{\mathbf{Y}}, \quad (4.2)$$

where both $\hat{\mathbf{S}}$ (a normalized version of \mathbf{S}) and $\hat{\mathbf{Y}}$ are regularized by a scalar α . To initialize the calculation of (4.2), $\mathbf{Y}_{k=0}$ can be set arbitrarily, e.g., $\mathbf{Y}_0 = \hat{\mathbf{Y}}$. By iteratively computing (4.2), a new label matrix $\mathbf{Y}^* = \lim_{k \rightarrow \infty} \mathbf{Y}_k$ that corresponds to the ultimate clustering results can be obtained. They also showed that the iterative calculations converge to an analytic solution,

$$\mathbf{Y}^* = \alpha(\mathbf{I} - \beta\hat{\mathbf{S}})^{-1}\hat{\mathbf{Y}}. \quad (4.3)$$

Algorithm 3: Semi-supervised spectral clustering

- 1 **Similarity matrix construction.** Compute a symmetric similarity matrix \mathbf{S} , with each element

$$s_{ij} \triangleq \begin{cases} \text{Sim}(v_i, v_j) & \text{if } i \neq j \text{ and } v_i \sim v_j; \\ 0 & \text{otherwise.} \end{cases}$$

Note $0 \leq \text{Sim}(v_i, v_j) \leq 1$ is a function measuring the similarity between the linked nodes v_i and v_j .

- 2 **Similarity matrix normalization.** Compute $\hat{\mathbf{S}} = \mathbf{D}^{-1/2} \mathbf{S} \mathbf{D}^{-1/2}$, where $\mathbf{D} = \text{diag}(d_1, \dots, d_l)$ is a diagonal matrix with $d_i = \sum_{j=1}^l s_{ij}$.
- 3 **Label matrix calculation.** Apply one of the following steps to the derivation of a label matrix \mathbf{Y}^* .

- a) **Iteratively compute $\mathbf{Y}_k = \beta^{1/2} \hat{\mathbf{S}} \beta^{1/2} \mathbf{Y}_{k-1} + \alpha \hat{\mathbf{Y}}$ until convergence.** The converged matrix is denoted by \mathbf{Y}^* . The $\alpha = \text{diag}(\alpha_1, \dots, \alpha_l)$ is a diagonal matrix with $\alpha_i \in [0, 1]$ weighting the label reliability of $\hat{\mathbf{y}}_i$, and $\beta = \mathbf{I} - \alpha$.
- b) Alternatively, **compute $\mathbf{Y}^* = (\mathbf{I} - \beta^{1/2} \hat{\mathbf{S}} \beta^{1/2})^{-1} \alpha \hat{\mathbf{Y}}$.** Actually the \mathbf{Y}^* computed here is the limit of the sequence $\{\mathbf{Y}_k\}$ as $k \rightarrow \infty$.

- 4 **Label assignment.** Assign each un-labeled node v_i a new label $y_i^* = \arg \max_{j \in \{1, \dots, C\}} \mathbf{Y}_{ij}^*$
-

To provide better control over individual label reliability, we further extend the regularization scalar α to a diagonal matrix $\alpha = \text{diag}(\alpha_1, \dots, \alpha_l)$, and developed a counterpart of (4.2) as

$$\mathbf{Y}_k = \beta^{1/2} \hat{\mathbf{S}} \beta^{1/2} \mathbf{Y}_{k-1} + \alpha \hat{\mathbf{Y}}, \quad (4.4)$$

where $\beta = \mathbf{I} - \alpha$. Similarly, the convergence of (4.4) can be shown as

$$\mathbf{Y}^* = \lim_{k \rightarrow \infty} \mathbf{Y}_k = (\mathbf{I} - \beta^{1/2} \hat{\mathbf{S}} \beta^{1/2})^{-1} \alpha \hat{\mathbf{Y}}. \quad (4.5)$$

Benefits of adopting the matrix form $\boldsymbol{\alpha}$, rather than using a scalar, will be demonstrated later with experiments.

Analytic solutions of both (4.3) and (4.5) can also be derived from the optimization of cost functions in which the label consistency is explicitly modeled. It is presented in [69] that (4.3) can be derived from

$$\mathbf{Y}^* \triangleq \arg \min_{\mathbf{Y}} \frac{1}{2} \left[\left(\sum_{i=1}^l \sum_{j=1}^l s_{ij} \left\| \frac{1}{\sqrt{d_i}} \mathbf{y}_i - \frac{1}{\sqrt{d_j}} \mathbf{y}_j \right\|^2 \right) + \mu \sum_{i=1}^l \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \right], \quad (4.6)$$

where \mathbf{y}_i denotes the label vector of v_i to be estimated, \mathbf{Y} is defined as $[\mathbf{y}_1 \ \dots \ \mathbf{y}_l]^T$, and $\mu > 0$ is a regularization parameter. While the first term of (4.6), as mentioned in [69], represents a *smoothness constraint* for preventing large changes of labels between nearby points, the second term, instead, expresses a *fitting constraint* for maintaining the consistencies between a label estimate \mathbf{y}_i and its prior label $\hat{\mathbf{y}}_i$. Likewise, (4.5) can also be obtained from the optimization of

$$\begin{aligned} \mathbf{Y}^* &\triangleq \arg \min_{\mathbf{Y}} \frac{1}{2} \left[\frac{1}{2} \left(\sum_{i=1}^l \sum_{j=1}^l s_{ij} \left\| \frac{\sqrt{\eta_i}}{\sqrt{d_i}} \mathbf{y}_i - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} \mathbf{y}_j \right\|^2 \right) + \sum_{i=1}^l \mu_i \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \right] \\ &\triangleq \arg \min_{\mathbf{Y}} \mathcal{Q}(\mathbf{Y}) \end{aligned} \quad (4.7)$$

where $\eta_i > 0$, $\mu_i > 0$, and $\eta_i + \mu_i = \delta$, for all i s, with a constant δ introduced to constrain the sum of each regularization pair η_i and μ_i ¹. The derivation of the generalized solution (4.5) from (4.7) are given below.

¹The value of δ can be arbitrary, as will be seen clearly later in the proof.

Proof. By defining the first term of the proposed cost function (4.7) as

$$\begin{aligned} \mathcal{Q}_1(\mathbf{Y}) &\triangleq \frac{1}{2} \left(\sum_{i=1}^l \sum_{j=1}^l s_{ij} \left\| \frac{\sqrt{\eta_i}}{\sqrt{d_i}} \mathbf{y}_i - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} \mathbf{y}_j \right\|^2 \right) \\ &= \frac{1}{2} \left(\begin{aligned} &\sum_{j=1}^l s_{1j} \sum_{c=1}^C \left(\frac{\sqrt{\eta_1}}{\sqrt{d_1}} y_{1c} - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} y_{jc} \right)^2 \\ &\quad \vdots \\ &+ \sum_{j=1}^l s_{kj} \sum_{c=1}^C \left(\frac{\sqrt{\eta_k}}{\sqrt{d_k}} y_{kc} - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} y_{jc} \right)^2 \\ &\quad \vdots \\ &+ \sum_{j=1}^l s_{lj} \sum_{c=1}^C \left(\frac{\sqrt{\eta_l}}{\sqrt{d_l}} y_{lc} - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} y_{jc} \right)^2 \end{aligned} \right) \end{aligned}$$

and denoting the c -th element of a label vector \mathbf{y}_k as y_{kc} , the derivative of \mathcal{Q}_1 with respect to y_{kc} can be shown as

$$\begin{aligned} \frac{\partial \mathcal{Q}_1}{\partial y_{kc}} &= \left[\sum_{j \neq k} s_{kj} \frac{\sqrt{\eta_k}}{\sqrt{d_k}} \left(\frac{\sqrt{\eta_k}}{\sqrt{d_k}} y_{kc} - \frac{\sqrt{\eta_j}}{\sqrt{d_j}} y_{jc} \right) \right. \\ &\quad \left. - \sum_{j \neq k} s_{jk} \frac{\sqrt{\eta_k}}{\sqrt{d_k}} \left(\frac{\sqrt{\eta_j}}{\sqrt{d_j}} y_{jc} - \frac{\sqrt{\eta_k}}{\sqrt{d_k}} y_{kc} \right) \right] \\ &= 2 \left(\eta_k y_{kc} - \sum_{j \neq k} \sqrt{\eta_k} \hat{s}_{kj} \sqrt{\eta_j} y_{jc} \right), \end{aligned}$$

by using $s_{kj} = s_{jk}$, $\hat{s}_{kj} = \frac{s_{kj}}{\sqrt{d_k d_j}}$, and $\sum_{j \neq k} \frac{s_{kj}}{\sqrt{d_k d_k}} = 1$.

On the other hand, the derivative of (4.7) with respect to y_{kc} can be obtained as

$$\frac{\partial \mathcal{Q}}{\partial y_{kc}} = (\eta_k + \mu_k) y_{kc} - \sum_{j \neq k} \sqrt{\eta_k} \hat{s}_{kj} \sqrt{\eta_j} y_{jc} - \mu_k \hat{y}_{kc}.$$

By setting $\frac{\partial \mathcal{Q}}{\partial y_{kc}} = 0$, we have

$$y_{kc} - \sum_{j \neq k} \sqrt{\frac{\eta_k}{\eta_k + \mu_k}} \hat{s}_{kj} \sqrt{\frac{\eta_j}{\eta_k + \mu_k}} y_{jc} - \frac{\mu_k}{\eta_k + \mu_k} \hat{y}_{kc} = 0. \quad (4.8)$$

Since $\eta_k + \mu_k = \delta = \eta_j + \mu_j$, then $\sqrt{\frac{\eta_j}{\eta_k + \mu_k}} = \sqrt{\frac{\eta_j}{\eta_j + \mu_j}}$. By defining $\alpha_k \triangleq \frac{\mu_k}{\eta_k + \mu_k}$ and $\beta_k \triangleq 1 - \alpha_k = \frac{\eta_k}{\eta_k + \mu_k}$, (4.8) can be further expressed as

$$y_{kc} - \sum_{j \neq k} \sqrt{\beta_k} \hat{s}_{kj} \sqrt{\beta_j} y_{jc} - \alpha_k \hat{y}_{kc} = 0. \quad (4.9)$$

Finally, by representing (4.9) using matrix notation, we arrive the following result,

$$\begin{aligned} \mathbf{Y}^* - \boldsymbol{\beta}^{1/2} \hat{\mathbf{S}} \boldsymbol{\beta}^{1/2} \mathbf{Y}^* - \boldsymbol{\alpha} \hat{\mathbf{Y}} &= \mathbf{0} \\ \Rightarrow \mathbf{Y}^* &= (\mathbf{I} - \boldsymbol{\beta}^{1/2} \hat{\mathbf{S}} \boldsymbol{\beta}^{1/2})^{-1} \boldsymbol{\alpha} \hat{\mathbf{Y}}. \end{aligned}$$

□

By introducing more regularization parameters in (4.7), the reliability of prior labels can be better controlled. Moreover, from the optimization of the cost function (4.7), the spatio-temporal consistency among video layer labels can be well preserved.

In Algorithm 3, the *Sim* function listed in Step 1 is often calculated from a Gaussian distribution in which various distance measurements of two neighboring nodes can be applied. The adopted similarity measure will be detailed in Sec. 4.3.1. For the estimation of label matrix \mathbf{Y}^* , either the analytic solution of (4.5) or the iterative approximation of (4.4) can be applied. While the former involves the computation of matrix inversion which will be numerically unstable if $(\mathbf{I} -$

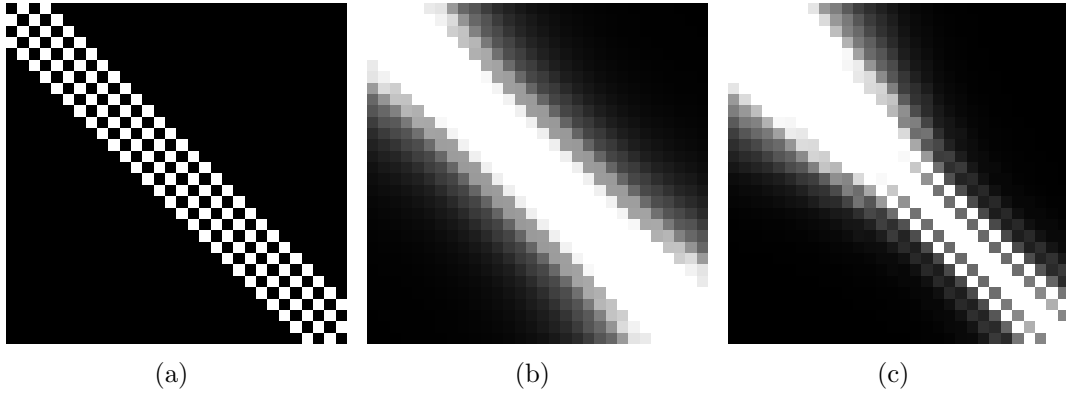


Figure 4.2: Examples of kernel construction. (a) A similarity matrix \mathbf{S} (shown as an image). All the similarities of connected nodes are set to 0.999. (b) The kernel \mathbf{K} computed from \mathbf{S} , with $\beta = 0.8\mathbf{I}$. Here, \mathbf{K} is normalized for better visualization. (c) The regularized kernel \mathbf{K} with the upper and the lower halves of β 's diagonal elements set to 0.8 and 0.5, respectively.

$\beta^{1/2}\hat{\mathbf{S}}\beta^{1/2}$) is near singular, an approximation based on the latter can be used to avoid such a problem. It is also worth mentioning that the algorithm is applicable to multiple layer decomposition, i.e., $C > 2$.

4.2.3 Algorithm Interpretation

The analytic solution of the semi-supervised spectral clustering given in (4.5) consists of the product of $(\mathbf{I} - \beta^{1/2}\hat{\mathbf{S}}\beta^{1/2})^{-1}$ and $\alpha\hat{\mathbf{Y}}$. The first part, $\mathbf{K} \triangleq (\mathbf{I} - \beta^{1/2}\hat{\mathbf{S}}\beta^{1/2})^{-1}$, can be regarded as a graph/diffusion kernel \mathbf{K} constructed from a similarity matrix \mathbf{S} [53], [69]. Fig. 4.2 shows simple examples of the kernel design. In Fig. 4.2 (a), a similarity matrix \mathbf{S} is depicted (shown as an image) to represent a graph where each node is connected to at most four neighbors. All the similarities are set to 0.999. By setting $\beta = 0.8\mathbf{I}$ and computing \mathbf{K} , the similarities are spread out, as shown in Fig. 4.2 (b). Note that, new edges with diffused similarities are established between some nodes that are not connected in the original

graph. Fig. 4.2 (c) shows a regularized kernel wherein the upper and the lower halves of the diagonal elements of β are set to 0.8 and 0.5, respectively.

For the second part, $\alpha \hat{\mathbf{Y}}$, it can be considered as a weighting scheme for label reliability. Subsequently, a matrix form of (4.5) can be written as

$$\mathbf{Y}^* = \begin{bmatrix} \mathbf{k}_1 \alpha \hat{\mathbf{y}}^1 & \dots & \mathbf{k}_1 \alpha \hat{\mathbf{y}}^C \\ \vdots & \ddots & \vdots \\ \mathbf{k}_l \alpha \hat{\mathbf{y}}^1 & \dots & \mathbf{k}_l \alpha \hat{\mathbf{y}}^C \end{bmatrix},$$

where \mathbf{k}_i denotes the i -th row of \mathbf{K} and $\hat{\mathbf{y}}^i$ denotes the i -th column of $\hat{\mathbf{Y}}$. The clustering label for v_i can then be determined by selecting the corresponding class with the maximum value in $\{\mathbf{k}_i \alpha \hat{\mathbf{y}}^1, \dots, \mathbf{k}_i \alpha \hat{\mathbf{y}}^C\}$. We interpret this labeling step as a weighted voting process among C classes. Together with similarity diffusion and weighted voting, the semi-supervised spectral clustering is done.

To sum up, the proposed framework for video layer propagation utilizes the semi-supervised spectral clustering to solve the label inference problem, which is different from the popular graph-cut approach [5]. A connection between video segmentation and kernel design is thus established, and an optimal solution derived from (4.7) that balances the spatio-temporal consistency in node labeling can be obtained. Moreover, a generalized formulation that regularizes the reliability of individual label is proposed, which makes the block-based propagation of video layers more feasible.

4.3 Algorithm Implementation

Several issues regarding the implementation of the proposed framework for video layer propagation are addressed in this section. First, the adopted similarity measure for the comparison of image blocks is presented. Next, to reduce the computational cost of kernel construction, a strategy of local clustering is applied. Due to this strategy, the propagation of video layers can be performed only within a band of image blocks near layer boundaries. Then, the embodiment of the regularization of label reliability is detailed. Finally, an optional user intervention scheme is introduced to allow possible refinements of layer propagation results.

4.3.1 Similarity Measure

Since the main focus of this work is on the investigation of effectiveness of video layer propagation using semi-supervised spectral clustering, only simple color features are adopted in our implementation. For each image block, three histograms computed from its YCbCr colors are used as image features. Specifically, denoted by $f_m(\mathbf{b}_i) \in \mathbb{R}^N$ is a normalized, N -bin color distribution of \mathbf{b}_i ($N = 16$), with $m \in \{1, \dots, 3\}$ indexing the m -th color channel. Though neither complex features nor high-level structures are used, the experimental results obtained by using the color cue alone, as will be shown in Sec. 4.4, are rather promising.

We choose the following multivariate Gaussian function as the similarity measure of two neighboring blocks:

$$Sim(\mathbf{b}_i, \mathbf{b}_j) = \exp \{-c \mathbf{D}(\mathbf{b}_i, \mathbf{b}_j)^T \boldsymbol{\Sigma}^{-1} \mathbf{D}(\mathbf{b}_i, \mathbf{b}_j)\},$$

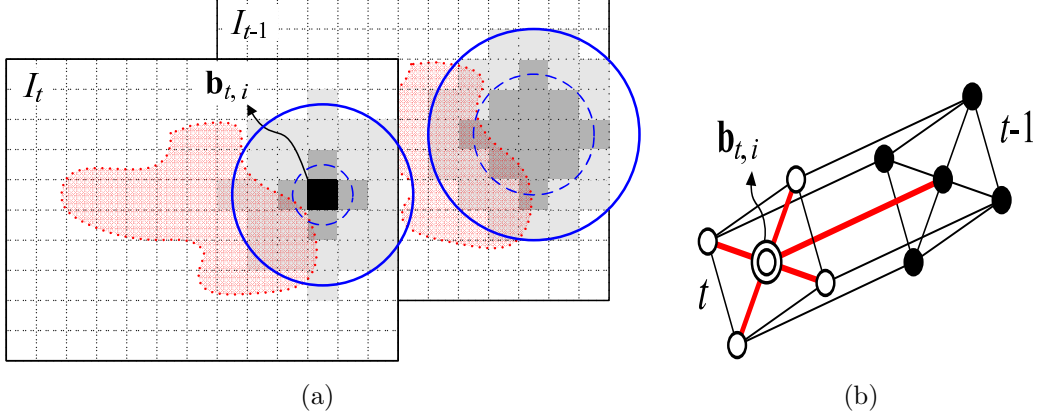


Figure 4.3: Examples of sub-graph \mathcal{G}'_i in different representations. (a) An example of \mathcal{G}'_i with $r_{spa} = w$, $r_{tem} = 2w$, $q_{spa} = 3w$ and $q_{tem} = 3.5w$. The dark-grayed blocks enclosed by the two dashed circles are the neighboring blocks with direct links to $\mathbf{b}_{t,i}$ (the black one). The light-grayed ones, which are not connected to $\mathbf{b}_{t,i}$ directly, are enclosed by the solid-line circles. (b) A simple example of \mathcal{G}'_i in graphical representation with $r_{spa} = q_{spa} = q_{tem} = w$ and $r_{tem} = 0$. The spatial and temporal neighbors of $\mathbf{b}_{t,i}$ (double-circled) are denoted by white and black circles respectively. As the red edges symbolize the direct links to $\mathbf{b}_{t,i}$, the black ones show the indirect links.

where c is a scaling factor, Σ is a covariance matrix, and $\mathbf{D}(\mathbf{b}_i, \mathbf{b}_j) \in \mathbb{R}^3$ is a distance vector with its m -th component $D_m(\mathbf{b}_i, \mathbf{b}_j) = B(f_m(\mathbf{b}_i), f_m(\mathbf{b}_j))$. The Bhattacharyya divergence $B(\mathbf{a}, \mathbf{b})$ is applied here to measure the distance of two color distributions \mathbf{a} and \mathbf{b} . As for Σ , it is assumed to be a diagonal matrix for simplicity. The estimation of Σ will be elaborated in the next subsection due to its coupling to the local clustering implementation. Regarding c , a typical value $c = 0.5$ is used in the experiments.

4.3.2 Local Clustering

As mentioned in Sec. 4.2.1, Fig. 4.1 can be represented by a graph \mathcal{G} . To find the unknown layer labels based on \mathcal{G} using Algorithm 3, all the graph nodes need to be taken into consideration, resulting in a very large similarity matrix. For the

construction of \mathbf{K} , the inversion of such a large matrix is required, which is often time consuming or even infeasible, e.g., due to limited computing resources. Accordingly, to reduce the computational cost, a local clustering strategy is employed by decomposing \mathcal{G} into many sub-graphs \mathcal{G}' 's and solving the layer labeling locally, block by block, based on each of the sub-graphs.

In local clustering, each image block $\mathbf{b}_{t,i}$ is processed independently by performing semi-supervised clustering in a small neighborhood of it. Fig. 4.3 (a) shows an example of \mathcal{G}'_i for estimating the label of $\mathbf{b}_{t,i}$, i.e., $y_{t,i}$. The nodes in \mathcal{G}'_i correspond to three types of image blocks: $\mathbf{b}_{t,i}$ itself (colored in black), those *directly* linked to $\mathbf{b}_{t,i}$ (colored in dark-gray, see also Fig. 4.1), and those *indirectly* linked to $\mathbf{b}_{t,i}$ (colored in light-gray). Although the indirectly linked nodes, defined by two the solid-line circles in Fig. 4.3 (a) with radii q_{spa} and q_{tem} , have no direct connection to $\mathbf{b}_{t,i}$, they still have effects on the inference of $y_{t,i}$ via similarity diffusion. Such nodes are included in \mathcal{G}'_i to maintain better label consistency in local clustering. Fig. 4.3 (b) depicts a simple example of \mathcal{G}'_i in graphical representation by setting r_{spa} , q_{spa} , and q_{tem} to the block width w , and $r_{tem} = 0$. In our experiments, these parameters are set to $w = 8$, $r_{spa} = 2w$, $r_{tem} = 3w$, $q_{spa} = r_{spa}$, and $q_{tem} = r_{spa} + r_{tem}$, respectively, resulting in more complex sub-graphs than that shown in Fig. 4.3 (b).

Based on the idea of sub-graph, a similarity matrix \mathbf{S} of much smaller size can be derived. The complexity of transforming \mathbf{S} to the kernel \mathbf{K} in (4.5) can thus be lowered down. Note that, before computing \mathbf{K} , we firstly check if $(\mathbf{I} - \beta^{1/2} \hat{\mathbf{S}} \beta^{1/2})$ is well-conditioned. If not, the iterative approximation of \mathbf{Y}^* is applied by computing (4.4) 100 times. Once \mathbf{Y}^* is obtained, we merely retrieve the matrix entries for $\mathbf{b}_{t,i}$ and determine its label. Thus, a sub-optimal yet practical estimate of $\mathbf{y}_{t,i}^*$ can

be derived. By combining all the local clustering results, new video layers of I_t are obtained. Owing to the usage of local clustering, spatial label inconsistency may occasionally occur. Therefore, a post-processing step is employed to eliminate isolated label estimates² in I_t and to preserve the global label consistency.

Using local clustering not only reduce the computational cost of matrix inversion, but also bring two mechanisms that will have beneficial effects on layer propagation: the estimation of local variances and the design of a propagation band. While the former, as being explained here, gives an estimation of Σ that captures local changes of layer appearances well, the latter, as will be discussed in the next subsection, enhances the layer propagation efficiency. When computing Sim , it is important to give a proper estimation of Σ . An intuitive choice is to compute Σ from all the distance vectors in \mathcal{G} . However, such a choice may result in a over-smoothed estimation. For example, the variance of block dissimilarities around a smooth area should not be as same as that around a textured region. Such a problem can be prevented by estimating Σ locally from \mathcal{G}' with the following steps:

1. **Distance vector normalization.** To enhance the local contrast among pairwise distances, each D_m regarding to \mathcal{G}' is re-scaled to $[0, 1]$ using the min-max normalization, for $m = 1, \dots, 3$.
2. **Covariance estimation.** To capture the within-class variances, a set of distance vectors \mathbf{D} 's computed from same-labeled node pairs of \mathcal{G}' are firstly selected. Then, the m -th dimensional variance of D'_m is calculated, $\forall m$, forming a diagonal covariance matrix Σ .

²A label estimate $y_{t,i}$ is said to be isolated if none of its four-connected neighbors in I_t is of the same class.

4.3.3 Propagation Band

With the adoption of local clustering, the propagation of video layers only needs to be performed for a spatial band of image blocks in I_t around prior layer boundaries found in I_{t-1} , and is thus more efficient. Given a sub-graph $\mathcal{G}'_i(\mathcal{E}'_i, \mathcal{V}'_i)$ for estimating $y_{t,i}$ and denoting the node constructed from I_{t-1} by $v'_{t-1,j}$, if the prior labels of all $v'_{t-1,j}$ s, i.e., $y_{t-1,j} \forall j$, are of the same class, i.e., $\mathbf{b}_{t,i}$ is located inside a previously determined layer, then $y_{t,i}$ can simply be assigned to the corresponding layer class without further clustering. An indicator function L on \mathcal{G}'_i is defined for this label check as

$$L(\mathcal{G}'_i) = \begin{cases} 1 & \text{if } v'_{t-1,j}, v'_{t-1,k} \in \mathcal{V}'_i \text{ and } y_{t-1,j} = y_{t-1,k}, \forall j, k; \\ 0 & \text{otherwise.} \end{cases}$$

A propagation band containing the image blocks to be processed in layer propagation can then be defined formally as $\mathcal{N} = \{\mathbf{b}_{t,i} | L(\mathcal{G}'_i) = 0\}$.

4.3.4 Regularization of Label Reliability

Due to the block-based processing, there is ambiguity in deciding the label of an image block located across multiple layer regions. Though the label of such a block can still be obtained either from user-labeling or from previous estimation, its reliability should be degraded while propagated to the next image frame. By adopting the regularization matrix α in (4.5), instead of using a scalar, the goal of controlling individual reliability of block label can be realized.

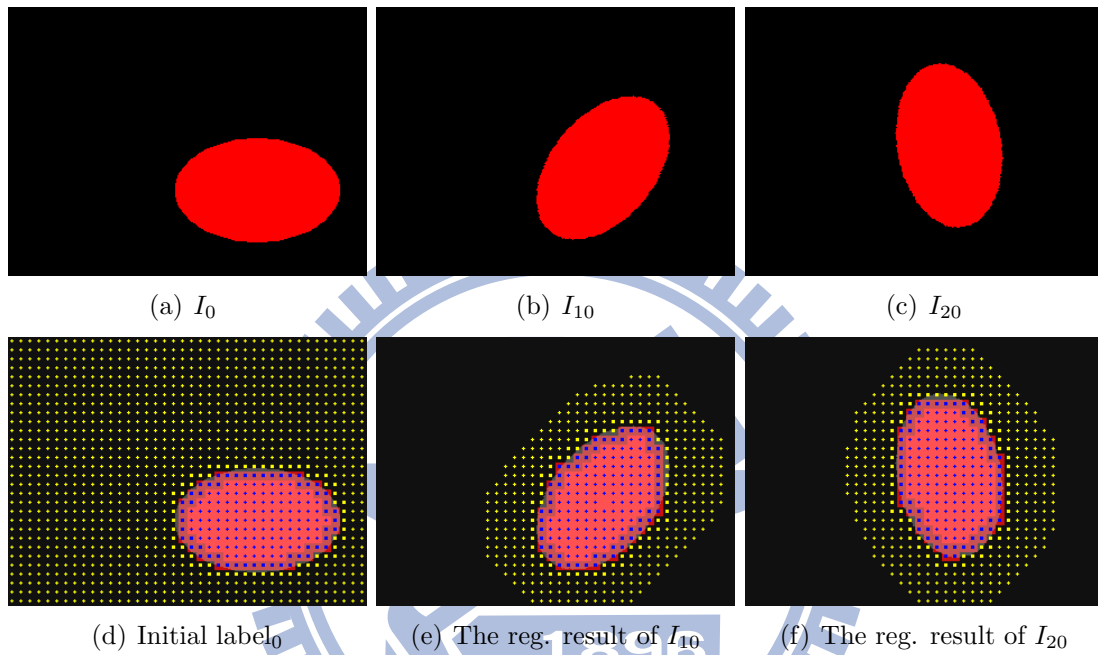


Figure 4.4: Simulated example for justifying the regularization of label reliability. (a)-(c) Snapshots of a synthesized image sequence. (d) The initial layer labels of I_0 , where the block centers are shown in yellow and blue for foreground and background respectively. The bold dots symbolize the image blocks of layer boundary. (e) and (f) Snapshots of the layer propagation results. All the dotted blocks (bold and non-bold) manifest the propagation bands.

Assume the regularized prior label matrix in (4.5) is sorted, i.e.,

$$\boldsymbol{\alpha}\hat{\mathbf{Y}} = \text{diag} \left(\begin{bmatrix} \boldsymbol{\alpha}_{t-1} \\ \boldsymbol{\alpha}_t \end{bmatrix} \right) \begin{bmatrix} \hat{\mathbf{Y}}_{t-1} \\ \hat{\mathbf{Y}}_t \end{bmatrix},$$

where the sub-matrices $\hat{\mathbf{Y}}_{t-1}$ and $\hat{\mathbf{Y}}_t$ denote the block labels of I_{t-1} and I_t , and their label reliability is regularized by $\boldsymbol{\alpha}_{t-1}$ and $\boldsymbol{\alpha}_t$, respectively. With the above notations, the regularization of label reliability is implemented by

1. **Boundary block identification.** The image blocks near layer boundaries are identified as those in I_{t-1} whose eight-connected neighbors have different layer labels.
2. **Regularization matrix construction.** To regularize individual label reliability, $\boldsymbol{\alpha}_t$ is set to a zero vector, the elements of $\boldsymbol{\alpha}_{t-1}$ that correspond to the non-boundary blocks are set to α_1 , and the rests of $\boldsymbol{\alpha}_{t-1}$ are set to $\alpha_2 < \alpha_1$.
3. **Quadruple labeling.** Besides the triple labeling in (4.1), one more assignment is proposed for the boundary blocks by setting their $\hat{\mathbf{y}}$ s to $[0.5 \ 0.5]$ to denote the label uncertainty.³

In our experiments, $\alpha_1 = 0.4$ and $\alpha_2 = \alpha_1/4$ are manually set. In short, the proposed regularization scheme adopts the novel matrix form $\boldsymbol{\alpha}$ for better control of individual label reliability, and represents boundary blocks by setting their labels to $[0.5 \ 0.5]$.

The above regularization scheme is justified by a simulated example shown in Fig. 4.4. A sequence of twenty image frames of size 320×240 is synthesized for

³The quadruple labeling is applied to the construction of $\hat{\mathbf{Y}}$ for the Step 3 in Algorithm 3. In the Step 4, the resulting layer labels are still in binary.

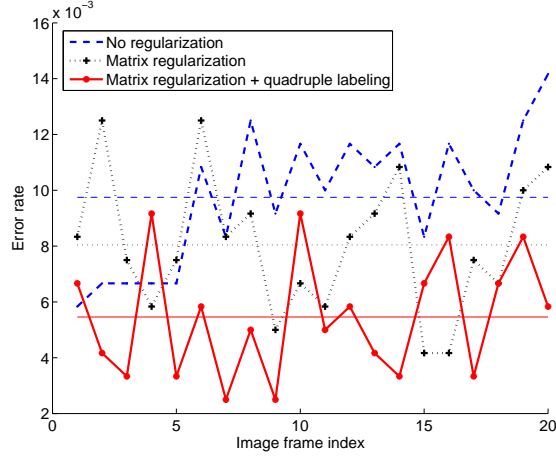


Figure 4.5: Quantitative evaluations of the proposed regularization of label reliability using the image sequence shown in Fig. 4.4. The dashed (blue), dotted (black) and solid (red) lines represent the error rates of layer label classification (i) without regularization, (ii) with matrix regularization using α , and (iii) with matrix regularization plus quadruple labeling, respectively. The three horizontal lines denote the corresponding error means, 0.00975, 0.00804, and 0.00546.

evaluation. To derive the ground-truth layer labels, an image block (8×8) located across the ellipse boundary is assigned to the foreground layer if it contains more than 50% red pixels of the ellipse. Totally, 24000 block labels are extracted. Three experiments are performed by propagating the initial video layers to the subsequent image frames (i) without regularization, (ii) with matrix regularization using α , and (iii) with matrix regularization plus quadruple labeling. Because the differences among the results of the three experiments can not be distinguished easily by visual inspection, only the results of (iii) are shown in Figs. 4.4 (e) and (f). For quantitative assessment, the error rates of the estimated labels with respect to the ground-truth are shown in Fig. 4.5. It is readily observable that the smallest error can be obtained for (iii). In Sec. 4.4, more experiments will be conducted using a real video to evaluate the proposed regularization scheme.

4.3.5 Optional User Intervention

In the proposed framework, only binary layer labels (for $C = 2$) are recorded and propagated to subsequent image frames. Thus, user interventions can be easily incorporated into the framework to correct layer propagation defects. When the video layers of I_t are derived, users can change the misclassified block labels to correct ones, e.g., via mouse clicks. The amended labels will override the clustering results during subsequent layer propagation, and the respective regularization parameters (elements of α) will also be set to 1. Some examples that demonstrate the usages and effects of user intervention for video layer propagation will be given in experiments.

4.4 Experimental Results

To assess the performance of the proposed framework for video layer propagation, two experiments are conducted using the *IU* [34] and the *Mobile* sequences. The *IU* sequence (320×240), being recorded by a static camera, is used for an investigation on effectiveness and limitation of our approach. Besides, the needs for user interventions are also presented in this experiment. The *Mobile* sequence (352×288), being captured by a moving camera, is used to demonstrate the capability of layer propagations for large motions, as well as to validate the regularization of label reliability. In the experiments, the experimental parameters (w , r_{spa} , r_{tem} , q_{spa} , q_{tem} , c , α_1 , and α_2) are set to the default values mentioned previously.



Figure 4.6: Results of the *IU* experiment. (a) The first image frame of the *IU* sequence. (b) The initial layer label mask. (c)-(i) Snapshots of the video layer propagation results for the *IU* sequence. While the brown circle marks the propagation errors due to undifferentiated regions (similar color distributions between the hair and the wall), the green ones indicate the errors resulting from uncovered regions.

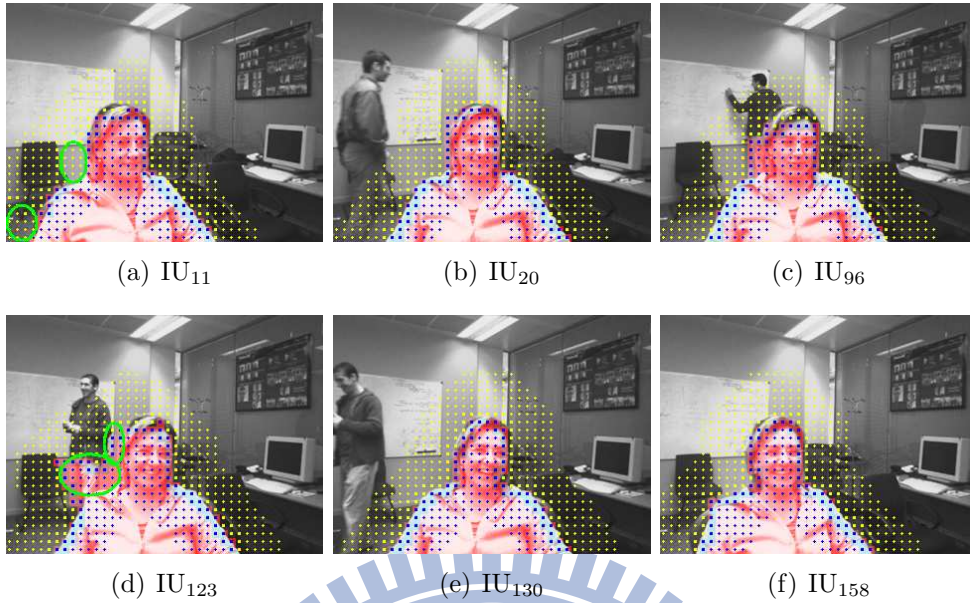


Figure 4.7: Results of the *IU* experiment with user interventions. (a)-(c) The results of the 1st user intervention. User re-labeling is applied to the circled regions in (a). (d) The propagation errors of uncovered region happened again and are corrected similarly. (e) and (f) The propagation results after the 2nd correction.

4.4.1 Video Layer Propagation in Static Background

The *IU* video is originally offered as a benchmark for mono and stereo video segmentation in [34]. To fit our needs, only the image sequence captured by the left camera is used. Ground-truth layer masks which contain foreground, background and in-between pixel classes are provided every fifth frame by [34], based on the depth information. The masks are then transformed into binary block labels, as the ground-truth for our evaluation, by majority votes.⁴ Figs. 4.6 (a) and (b) show the first image frame of the *IU* sequence and its initial layer label mask, respectively. Figs. 4.6 (c)-(i) are the video layer propagation results. Despite the effective propagation of video layers throughout the sequence, two problems can

⁴Because very few pixels located near layer boundaries are labeled as in-between, they have almost no effect on the subsequent binary labeling of image blocks.

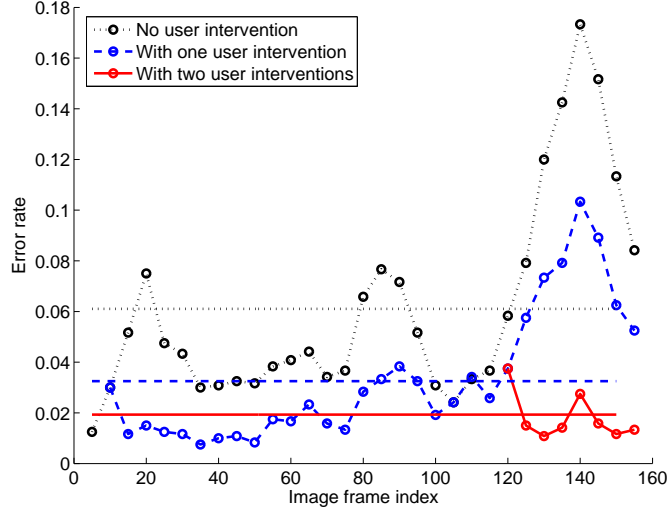


Figure 4.8: Quantitative evaluations for the *IU* experiment. The dotted (black), dashed (blue) and solid (red) lines are the error distributions without, with-one- and with-two- user interventions, respectively. The three horizontal lines denote the respective average error levels.

be observed from the results. The first one is due to undifferentiated regions, e.g., the region circled in brown in Fig. 4.6 (f). Because of our adoption of simple color feature, the circled area whose color distributions are similar to the surrounding scenes, is misclassified. Although such a problem might be alleviated by using more complicated feature design, it is beyond the scope of this study.

The second problem is due to uncovered regions, e.g., as marked by green circles in Figs. 4.6 (c) and (d). Being previously occupied by the foreground, these uncovered regions are misclassified, inducing more errors in later frames. Such a problem is more critical to the proposed approach, because using only local information in the current design to guide the propagation process cannot guarantee correct labeling of uncovered region. Fortunately, the above problems mostly start from small misclassified regions, and can be easily amended by user interventions. We hence choose to manually correct the erroneous block labels of

IU₁₁, in which the errors are more obvious, and then resume the layer propagation. Figs. 4.7 (a)-(c) show better results after the correction. In Fig 4.7 (d), the same problem happened again and is corrected similarly, resulting in more accurate layer segmentation, e.g., in Figs. 4.7 (e) and (f). The classification errors in terms of the numbers of incorrect block labeling for the *IU* sequence are plotted in Fig. 4.8, where the three curves correspond to the error distributions without-, with-one-, and with-two- user interventions. Thus, through manual corrections of only two frames, the mean error rate is greatly reduced from 6.10%^s to 1.93%⁵.

4.4.2 Video Layer Propagation in Moving Background

In the *Mobile* experiment, the performance of the proposed framework in propagating video layers undergoing large motions is quantitatively assessed. As shown in Fig. 4.9, snapshots and the corresponding ground-truth layer masks of the *Mobile* sequence are exhibited. The video layer propagation is performed on every other frame. Totally 30 ground-truth masks are provided, where the first 15 masks are extracted every other frame from Mobile₂-Mobile₃₀ and the others are extracted every 10 frame from Mobile₄₀-Mobile₁₈₀. Challenges of this experiment include the complex background scene, the large changes in motion of the foreground and background layers, and the similarity in color between the rolling ball and some background textures, e.g., the blue-circled regions in Figs. 4.9 (a) and (b). Snapshots of the video layer propagation results are shown in Fig. 4.10. The video layer changes are effectively captured throughout the entire sequence without any user

⁵In [67], the mean error of the *IU* sequence is 2.56%. However, it should not be directly compared to our result, for its pixel-wised evaluation is actually stricter than the block-wised counterpart. Besides, the method presented in [67] is based on supervised learning, which is quite different from the proposed approach.

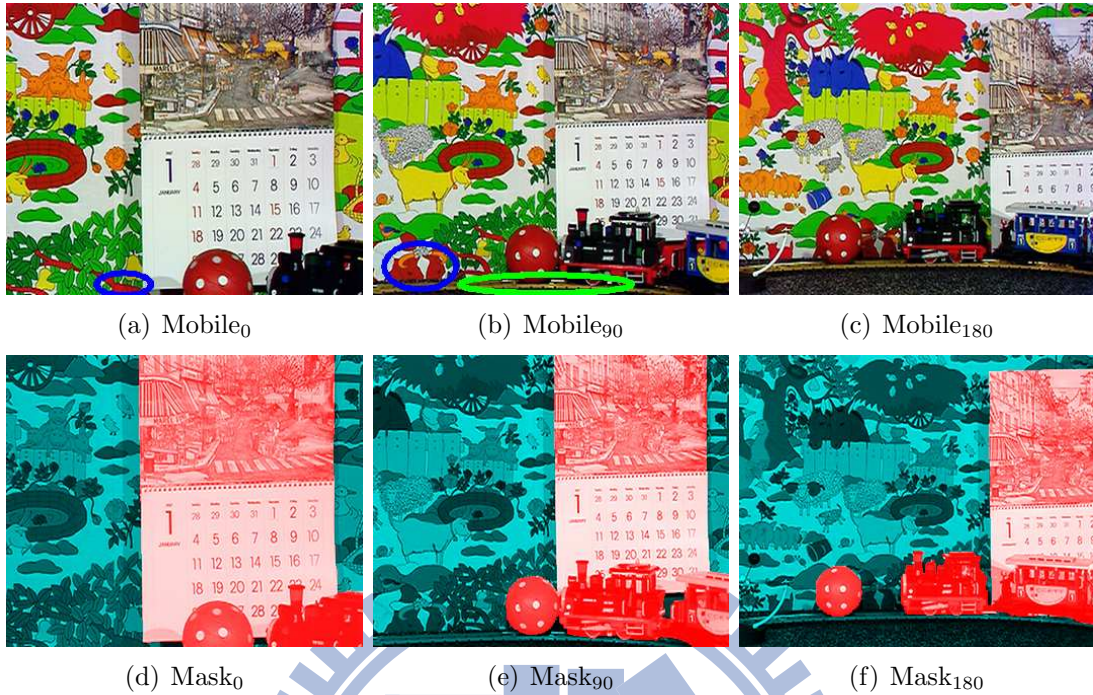


Figure 4.9: Snapshots of the *Mobile* sequence and the corresponding ground-truth layer masks. The blue circles indicate some background regions similar to the rolling ball. The green ones enclose some uncovered regions.

intervention involved. Particularly, the calendar, which undergoes translations and size changes, is correctly tracked. Also, the rolling ball is captured in unity, despite its similar color to some background textures. On the other hand, the layer propagation defects caused by uncovered regions can still be perceived, e.g., the circled regions in Figs. 4.10 (b) and (c). The results validate the effectiveness of the proposed approach using local, regional features to track deformable video layers, without prior knowledge, e.g., layer rigidity [15], being imposed.

The quantitative assessments of the layer labeling errors using the 30 masks are shown in Fig. 4.11. We also compare the error rates (i) without regularization, (ii) with matrix regularization using α , and (iii) with matrix regularization plus quadruple labeling in this evaluation. Again, the regularization scheme of (iii)

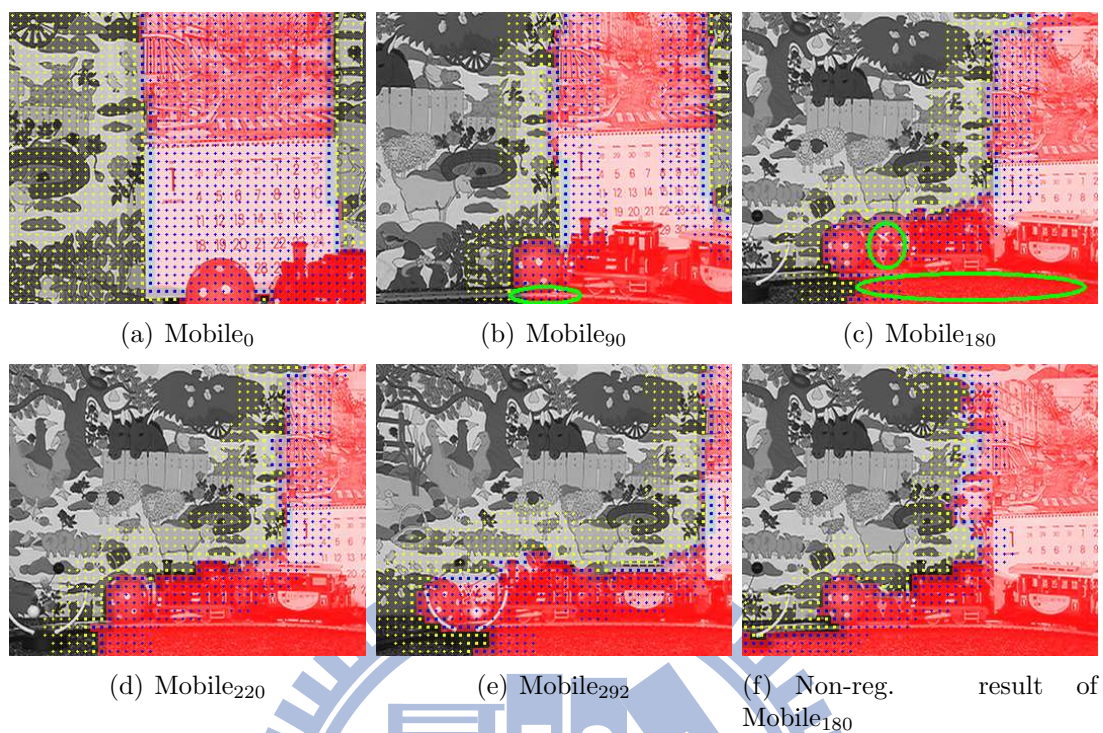


Figure 4.10: Results of the *Mobile* experiment. (a)-(e) The results obtained by using the label reliability regularization. Some problems caused by uncovered regions are marked by green circles. (f) A result obtained without using the label reliability regularization.

gives the best result. For a qualitative comparison, a video layer propagation result without regularization is shown in Fig. 4.10 (f).

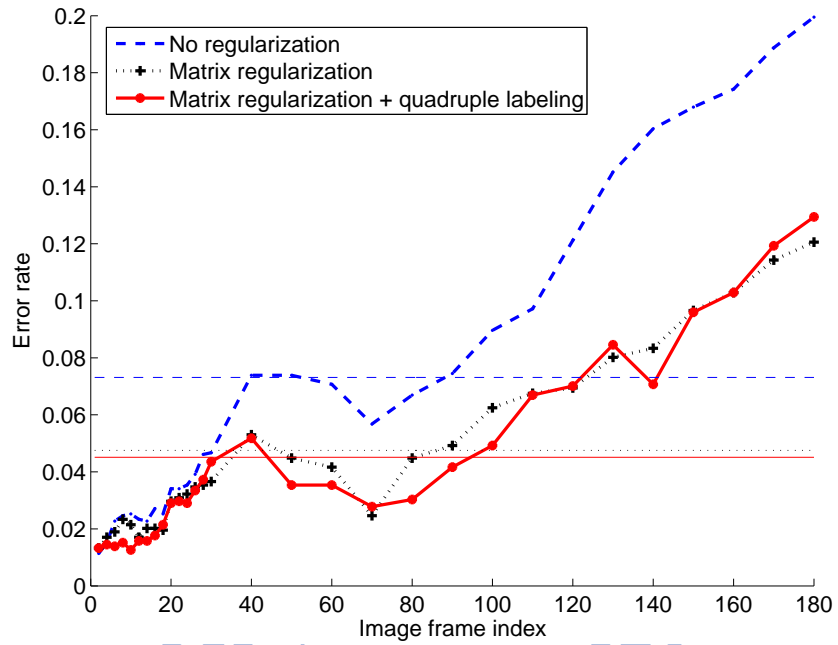
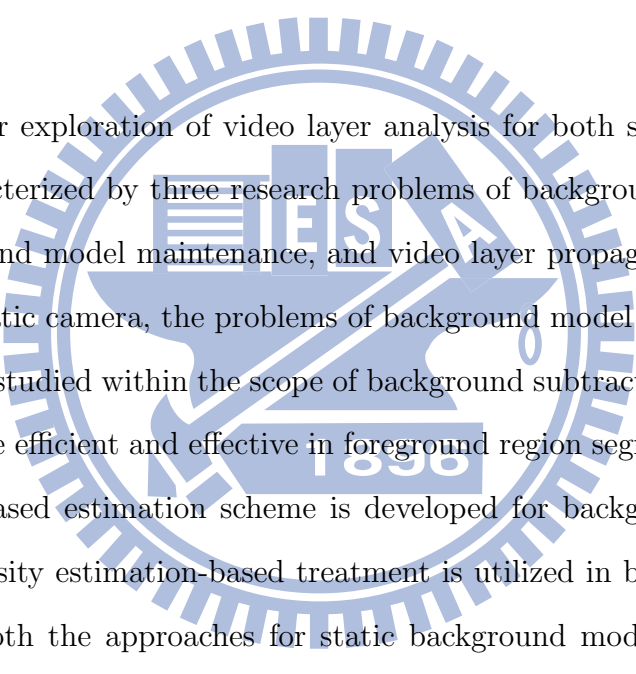


Figure 4.11: Quantitative assessment on regularization of label reliability using the *Mobile* sequence. The dashed (blue), dotted (black) and solid (red) lines represent the errors of layer labeling (i) without regularization, (ii) with matrix regularization using α , and (iii) with matrix regularization plus quadruple labeling. The respective mean errors are 0.0731, 0.0475, and 0.0451.

Chapter 5

Conclusion



In this thesis, our exploration of video layer analysis for both static and moving cameras is characterized by three research problems of background model initialization, background model maintenance, and video layer propagation. For videos captured by a static camera, the problems of background model initialization and maintenance are studied within the scope of background subtraction that has long been proven to be efficient and effective in foreground region segmentation. While a classification-based estimation scheme is developed for background model initialization, a density estimation-based treatment is utilized in background model maintenance. Both the approaches for static background modeling are demonstrated to outperform some existing methods. For the videos captured by moving camera, the research aim is on tracking the dynamic changes of video layers within a video shot. A new framework for video layer propagation via semi-supervised spectral clustering is proposed. Video layer changes induced by, e.g., camera and object motions, can be well-captured using the proposed framework.

In investigation of the three research problems, different machine learning tech-

niques, i.e., supervised classification, semi-supervised clustering, and un-supervised density estimation, are applied to solve the problems in systematic ways. While block-based processing for deriving stable features in foreground and background discrimination and for reduce computational complexities is adopted in supervised and semi-supervised learning, pixel-based processing for recording detailed background scene variations is applied in un-supervised learning. Further discussions with respect to each studied problem are given in the following sections.

5.1 Summary of Static Background Model Initialization

To establish an initial background model for tracking, an efficient on-line algorithm is proposed. The key idea of the proposed approach is simple but effective: If one can tell whether an image block is part of the background, this knowledge can help to perform bottom-up block updates to derive a complete background model. In addition, we introduce a top-down consistency check to eliminate noises in the updates. The two mechanisms, together, lead to a reliable system. Regarding the system tuning, there is only one parameter that needs to be determined. Indeed, the experimental results demonstrate that the algorithm is robust to different parameter settings, and can handle lighting variations.

In background classifier learning, a classifier formulation with probability outputs is adopted so that the classification boundary can be easily tuned. While both an SVM and a CGBoost classifier are appropriate for this purpose, the latter has an advantage of efficiency and is thus applied to the experiments. We also

note that the relevance vector machines (RVMs) [56] are another possible choice. Particularly, RVMs are derived from MAP equations, and are truly probabilistic.

Overall, the proposed system for background model initialization is shown to be useful and practical for real-time surveillance. For the future work, developing wide-range background model estimation schemes to accommodate a moving camera could be a direction worthy of further exploration. This extension may lead to a more challenging problem concerning spatial registration and temporal modeling of background regions in motions.

5.2 Summary of Static Background Model Maintenance

In background model learning, maintaining a balance between robustness to background variations and sensitivity to foreground changes has long been regarded as a hard problem. In our study, the trade-off between model robustness and sensitivity can be effectively regularized via the clarification of the different roles of different learning rates for the GMM and by adopting the proposed bivariate rate control scheme. Experimental results show that, with careful tuning of the learning rates for mixture weights, robustness to quick variations in background as well as sensitivity to abnormal changes in foreground can be achieved simultaneously for several surveillance scenarios. In addition, a heuristic for adaptation of double-quick lighting change is proposed and verified in this work. With the help of this heuristic, large lighting changes occurring in very short time intervals, e.g., within one second, can be absorbed into background.

Our design of the bivariate learning rate control for the GMM roots in the high-level feedback of pixel types identified by a surveillance system. Although, in our current setting, only a limited amount of pixel types are computed for the rate control, noticeable improvements in foreground detection over conventional GMM approaches are already observable. Owing to the scalability of the proposed scheme, more complex scenarios may be handled as more high-level information incorporated. For example, region-level classification results of skin/non-skin, face/non-face and human/non-human can be fed back to the pixel-level control of the learning rate η in background modeling to increase model sensitivity to these objects. Also, proper settings of the hyper-parameters α and β for pixels of high spatio-temporal gradients may be worth an investigation. Another interesting direction is to apply biological cues, e.g., discriminant saliency [40] between center and surround, to increase the adaptation rates for background pixels of highly dynamic background scenes that are often misclassified as foreground one.

5.3 Summary of Dynamic Video Layer Propagation

The characteristics of the proposed framework for video layer propagation include its adoption of semi-supervised spectral clustering and the choice of block-based processing. By casting video layer propagation into layer label inference and solving the label inference problem via semi-supervised spectral clustering, a new connection between video segmentation and kernel design has been built. Under such a framework, the feasibility of using block units for layer propagation is also ex-

plored. Adopting image blocks as processing units has advantages in forming useful regional features, avoiding preprocessing of region segmentation, and fitting video coding applications. However, due to the coarse resolution of block units, ambiguity in labeling a block over layer boundaries may occur. Fortunately, with the help of the proposed reliability regularization over block labels, the problem is alleviated.

To reduce the computational cost of large matrix inversion, a strategy of local clustering that solves layer labels by part is employed in our implementation. With this local processing, new video layers can be computed only within a band of blocks around layer boundaries, further increasing the processing efficiency. Moreover, because only layer labels need to be propagated, user interventions can be incorporated easily into the layer propagation process to amend defects. Experimental results given in Chapter 4 show that, with the mere use of simple color feature, the video layers can be effectively tracked, even for those video layers in complex scenes and undergoing large, non-rigid motions.

Regarding the future work, more experiments still need to be conducted for further evaluation of system efficiency and accuracy. Comparisons between the proposed method and the state-of-the-art graph cut techniques may also be included in future study. Besides, for increase the discriminability in clustering, the adoption of more features, e.g., motions and textures, may be helpful, and could lead to a mixture design of kernel matrices constructed from different features.



Appendix

MAP Formulation for

Background Model Estimation

Described below are the derivations of the MAP formulation (2.2) in Sec.2.2.1. For easy explanation, the derivations are decomposed into six parts, which are classifier training, iterative formulation, posterior probability decomposition, likelihood probability decomposition, background block classification, and the final MAP formulation.

Classifier Training

To begin with, a MAP classifier derived from the training data \mathbf{D} is defined by $f^* = \arg \max_f P(f | \mathbf{D}) = \arg \max_f P(f | \mathbf{X}, \mathbf{Y})$. It can be interpreted as a supervised learning process to train an optimal classifier f^* from the training data $\mathbf{D} = \{\mathbf{X}, \mathbf{Y}\}$. With the definition of f^* , we can start to derive the following

equations to estimate a background model.

$$\begin{aligned}
& P(\tilde{B}_t \mid \mathbf{I}_t, \mathbf{D}) \\
&= \int P(\tilde{B}_t \mid \mathbf{I}_t, \mathbf{D}, f) p(f \mid \mathbf{I}_t, \mathbf{D}) df \\
&= \int P(\tilde{B}_t \mid \mathbf{I}_t, f) p(f \mid \mathbf{D}) df \\
&\approx P(\tilde{B}_t \mid \mathbf{I}_t, f^*),
\end{aligned}$$

where $P(f \mid \mathbf{D})$ is assumed to peak at the optimal classifier f^* (e.g., see [49], pp.474–476).

Iterative Formulation

To develop an iterative form for estimating a background model, we first define

$$\tilde{B}_t^* = \arg \max_{\tilde{B}_t} P(\tilde{B}_t \mid \mathbf{I}_t, f^*),$$

and

$$\tilde{B}_{t-1}^* = \arg \max_{\tilde{B}_{t-1}} P(\tilde{B}_{t-1} \mid \mathbf{I}_{t-1}, f^*).$$

Then we have

$$\begin{aligned}
& P(\tilde{B}_t \mid \mathbf{I}_t, f^*) \\
&= \sum_{\tilde{B}_{t-1}} \left(P(\tilde{B}_t \mid \mathbf{I}_t, f^*, \tilde{B}_{t-1}) P(\tilde{B}_{t-1} \mid \mathbf{I}_t, f^*) \right) \\
&= \sum_{\tilde{B}_{t-1}} \left(P(\tilde{B}_t \mid \mathbf{I}_{t,\ell}, \mathbf{I}_{t-\ell}, f^*, \tilde{B}_{t-1}) P(\tilde{B}_{t-1} \mid \mathbf{I}_{t-1}, I_t, f^*) \right) \\
&\quad \text{(The image frames } \mathbf{I}_{t,\ell} \text{ are used later to compute} \\
&\quad \text{feature vectors for classification.)} \\
&= \sum_{\tilde{B}_{t-1}} \left(P(\tilde{B}_t \mid \mathbf{I}_{t,\ell}, f^*, \tilde{B}_{t-1}) P(\tilde{B}_{t-1} \mid \mathbf{I}_{t-1}, f^*) \right) \\
&\approx P(\tilde{B}_t \mid \mathbf{I}_{t,\ell}, f^*, \tilde{B}_{t-1}^*),
\end{aligned}$$

where, similarly, $P(\tilde{B}_{t-1} \mid \mathbf{I}_{t-1}, f^*)$ is assumed to peak at \tilde{B}_{t-1}^* .

Posterior Probability Decomposition

Using Bayes' rule, we decompose $P(\tilde{B}_t \mid \mathbf{I}_{t,\ell}, f^*, \tilde{B}_{t-1}^*)$ into a product of an image likelihood term and a prior.

$$\begin{aligned}
& P(\tilde{B}_t \mid \mathbf{I}_{t,\ell}, f^*, \tilde{B}_{t-1}^*) \\
&= P(\tilde{B}_t \mid I_t, \dots, I_{t-\ell+1}, f^*, \tilde{B}_{t-1}^*) \\
&\propto P(I_t \mid \tilde{B}_t, I_{t-1}, \dots, I_{t-\ell+1}, \tilde{B}_{t-1}^*, f^*) P(\tilde{B}_t \mid I_{t-1}, \dots, I_{t-\ell+1}, f^*, \tilde{B}_{t-1}^*) \\
&= P(I_t \mid \tilde{B}_t, \mathbf{I}_{t-1,\ell-1}, \tilde{B}_{t-1}^*, f^*) P(\tilde{B}_t \mid \mathbf{I}_{t-1,\ell-1}, f^*, \tilde{B}_{t-1}^*) \\
&= \underbrace{P(I_t \mid \tilde{B}_t, \mathbf{I}_{t-1,\ell-1}, \tilde{B}_{t-1}^*, f^*)}_{\text{image likelihood}} \underbrace{P(\tilde{B}_t \mid \tilde{B}_{t-1}^*)}_{\text{prior}}.
\end{aligned}$$

Because the classifier f^* is used to perform block-wise (local) classifications, and $\mathbf{I}_{t-1,\ell-1}$ are those image frames used in computing feature values, both of them are eliminated from the prior probability which is used to measure the global consistency over image blocks. That is, we simplify the prior term from $P(\tilde{B}_t | \mathbf{I}_{t-1,\ell-1}, f^*, \tilde{B}_{t-1}^*)$ to $P(\tilde{B}_t | \tilde{B}_{t-1}^*)$.

Likelihood Probability Decomposition

Applying the assumption of block-wise independencies, the likelihood term can be further decomposed as follows.

$$\begin{aligned}
& P(I_t | \tilde{B}_t, \mathbf{I}_{t-1,\ell-1}, \tilde{B}_{t-1}^*, f^*) \\
&= \prod_{i=1}^n P(b_t^i | \tilde{b}_t^i, \mathbf{b}_{t-1,\ell-1}^i, \tilde{b}_{t-1}^{*i}, f^*) \\
&\propto \prod_{i=1}^n \left(P(b_t^i | \mathbf{b}_{t-1,\ell-1}^i, \tilde{b}_{t-1}^{*i}, f^*) P(\tilde{b}_t^i | b_t^i, \mathbf{b}_{t-1,\ell-1}^i, \tilde{b}_{t-1}^{*i}, f^*) \right) \\
&= \prod_{i=1}^n \left(P(b_t^i | \mathbf{b}_{t-1,\ell-1}^i, \tilde{b}_{t-1}^{*i}, f^*) P(\tilde{b}_t^i | \mathbf{b}_{t,\ell}^i, \tilde{b}_{t-1}^{*i}, f^*) \right) \\
&= \prod_{i=1}^n P(b_t^i | \mathbf{b}_{t-1,\ell-1}^i) \prod_{i=1}^n P(\tilde{b}_t^i | \mathbf{b}_{t,\ell}^i, \tilde{b}_{t-1}^{*i}, f^*). \\
&\propto \prod_{i=1}^n P(\tilde{b}_t^i | \mathbf{b}_{t,\ell}^i, \tilde{b}_{t-1}^{*i}, f^*).
\end{aligned}$$

The term $P(b_t^i | \mathbf{b}_{t-1,\ell-1}^i, \tilde{b}_{t-1}^{*i}, f^*)$ is reduced to $P(b_t^i | \mathbf{b}_{t-1,\ell-1}^i)$, because b_t^i is the i th block of frame I_t from an arbitrary on-line image stream, and it should be independent from our choice of a classifier f^* and what the i th block of a background model is at time $t - 1$, i.e., \tilde{b}_{t-1}^{*i} .

Background Block Classification

To utilize background block classification in estimating a background model, we have

$$P(\tilde{b}_t^i | \mathbf{b}_{t,\ell}^i, \tilde{b}_{t-1}^{*i}, f^*) \doteq \begin{cases} P(\tilde{b}_t^i | b_t^i, \tilde{b}_{t-1}^{*i}), & \text{if } b_t^i \text{ is classified as background by } f^*, \\ P(\tilde{b}_t^i | \tilde{b}_{t-1}^{*i}), & \text{otherwise.} \end{cases}$$

Then we derive the decomposition for the image likelihood,

$$P(I_t | \tilde{B}_t, \mathbf{I}_{t-1,\ell-1}, \tilde{B}_{t-1}^*, f^*) \propto \prod_{i^+} P(\tilde{b}_t^i | b_t^i, \tilde{b}_{t-1}^{*i}) \prod_{i^-} P(\tilde{b}_t^i | \tilde{b}_{t-1}^{*i}),$$

where $i^+ = \{i | b_t^i \text{ is a background block}\}$ and $i^- = \{1, \dots, n\} - i^+$.

The Final MAP Formulation

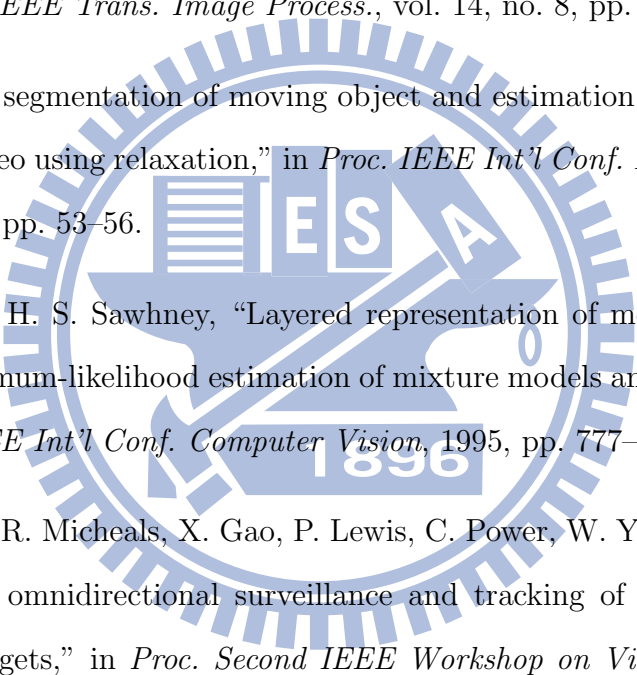
With all these derivations, we arrive at the following MAP optimization

$$\tilde{B}_t^* = \arg \max_{\tilde{B}_t} \left\{ \left(\prod_{i^+} P(\tilde{b}_t^i | b_t^i, \tilde{b}_{t-1}^{*i}) \prod_{i^-} P(\tilde{b}_t^i | \tilde{b}_{t-1}^{*i}) \right) P(\tilde{B}_t | \tilde{B}_{t-1}^*) \right\}.$$

□



References

- 
- [1] P. M. Q. Aguiar and J. M. F. Moura, “Figure-ground segmentation from occlusion,” *IEEE Trans. Image Process.*, vol. 14, no. 8, pp. 1109–1124, 2005.
- [2] —, “Joint segmentation of moving object and estimation of background in low-light video using relaxation,” in *Proc. IEEE Int’l Conf. Image Processing*, vol. 5, 2007, pp. 53–56.
- [3] S. Ayer and H. S. Sawhney, “Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding,” in *Proc. IEEE Int’l Conf. Computer Vision*, 1995, pp. 777–784.
- [4] T. E. Boulton, R. Micheals, X. Gao, P. Lewis, C. Power, W. Yin, and A. Erkan, “Frame-rate omnidirectional surveillance and tracking of camouflaged and occluded targets,” in *Proc. Second IEEE Workshop on Visual Surveillance*, 1999, pp. 48–55.
- [5] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.

- [6] H.-T. Chen, H.-H. Lin, and T.-L. Liu, “Multi-object tracking using dynamical graph matching,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2001, pp. 210–217.
- [7] S. Cohen, “Background estimation as a labeling problem,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 2, 2005, pp. 1034–1041.
- [8] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov, “Bilayer segmentation of live video,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 53–60.
- [9] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, “Detecting moving objects, ghosts, and shadows in video streams,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1337–1342, Oct. 2003.
- [10] T. J. Darrell and A. P. Pentland, “Cooperative robust estimation using layers of support,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 5, pp. 474–487, 1995.
- [11] F. De la Torre and M. J. Black, “Robust principal component analysis for computer vision,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 1, 2001, pp. 362–369.
- [12] A. Demiriz, K. P. Bennett, and J. Shawe-Taylor, “Linear programming boosting via column generation,” *Machine Learning*, vol. 46, no. 1-3, pp. 225–254, 2002.

- [13] A. Elgammal, D. Harwood, and L. S. Davis, “Non-parametric background model for background subtraction,” in *Proc. European Conf. Computer Vision*, vol. 2, 2000, pp. 751–767.
- [14] T. Ellis and M. Xu, “Object detection and tracking in an open and dynamic world,” in *Proc. IEEE Int’l Workshop Performance Evaluation of Tracking and Surveillance*, 2001.
- [15] M. Fradet, P. Pérez, and P. Robert, “Time-sequential extraction of motion layers,” in *Proc. IEEE Int’l Conf. Image Processing*, 2008, pp. 3224–3227.
- [16] Y. Freund and R. E. Schapire, “Experiments with a new boosting algorithm,” in *Proc. Int’l Conf. Machine Learning*, 1996, pp. 148–156.
- [17] B. J. Frey, N. Jojic, and A. Kannan, “Learning appearance and transparency manifolds of occluded objects in layers,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 45–52.
- [18] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting,” *The Annals of Statistics*, vol. 38, no. 2, pp. 337–374, April 2000.
- [19] N. Friedman and S. Russell, “Image segmentation in video sequences: A probabilistic approach,” in *Proc. Conf. Uncertainty in Artificial Intelligence*, 1997, pp. 175–181.
- [20] X. Gao, T. E. Boult, F. Coetzee, and V. Ramesh, “Error analysis of background adaption,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2000, pp. 503–510.

- [21] W. Grimson, C. Stauffer, R. Romano, and L. Lee, “Using adaptive tracking to classify and monitor activities in a site,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1998, pp. 22–29.
- [22] D. Gutchess, M. Trajkovics, E. Cohen-Solal, D. Lyons, and A. K. Jain, “A background model initialization algorithm for video surveillance,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 1, 2001, pp. 733–740.
- [23] I. Haritaoglu, D. Harwood, and L. S. Davis, “A fast background scene modeling and maintenance for outdoor surveillance,” in *Proc. Int’l Conf. Pattern Recognition*, vol. 4, 2000, pp. 179–183.
- [24] —, “W4: Real-time surveillance of people and their activities,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 809–830, 2000.
- [25] M. Harville, “A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models,” in *Proc. European Conf. Computer Vision*, vol. 3, 2002, pp. 543–560.
- [26] E. Hayman and J.-O. Eklundh, “Statistical background subtraction for a mobile observer,” in *Proc. IEEE Int’l Conf. Computer Vision*, 2003, pp. 67–74.
- [27] M. Heikkilä and M. Pietikäinen, “A texture-based method for modeling the background and detecting moving objects,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 657–662, 2006.
- [28] M. Irani and S. Peleg, “Motion analysis for image enhancement: Resolution, occlusion, and transparency,” *J. Visual Communication and Image Representation*, vol. 4, pp. 324–335, 1993.

- [29] N. Jojic and B. J. Frey, “Learning flexible sprites in video layers,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. 199–206.
- [30] N. Jojic, J. Winn, and L. Zitnick, “Escaping local minima through hierarchical model selection: Automatic object discovery, segmentation, and tracking in video,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 117–124.
- [31] Q. Ke and T. Kanade, “A subspace approach to layer extraction,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2001, pp. 255–262.
- [32] D.-W. Kim and K.-S. Hong, “Practical background estimation for mosaic blending with patch-based markov random fields,” *Pattern Recognition*, vol. 41, no. 7, pp. 2145–2155, 2008.
- [33] T. Ko, S. Soatto, and D. Estrin, “Background subtraction on distributions,” in *Proc. European Conf. Computer Vision*, vol. 3, 2008, pp. 276–289.
- [34] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother, “Bi-layer segmentation of binocular stereo video,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 407–414.
- [35] M. P. Kumar, P. H. S. Torr, and A. Zisserman, “Learning layered motion segmentations of video,” *Int’l J. Computer Vision*, vol. 76, pp. 301–319, 2008.

- [36] D.-S. Lee, “Effective gaussian mixture learning for video background subtraction,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 827–832, 2005.
- [37] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, “Statistical modeling of complex backgrounds for foreground object detection,” *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1459–1472, 2004.
- [38] Y. Li, J. Sun, and H.-Y. Shum, “Video object cut and paste,” *ACM Trans. Graphics*, vol. 24, no. 3, pp. 595–600, 2005.
- [39] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *Proc. DARPA IU Workshop*, 1981, pp. 121–130.
- [40] V. Mahadevan and N. Vasconcelos, “Background subtraction in highly dynamic scenes,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008, pp. 1–6.
- [41] S. J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler, “Tracking groups of people,” *Computer Vision and Image Understanding*, vol. 80, no. 1, pp. 42–56, October 2000.
- [42] A. Mittal and D. Huttenlocher, “Site modeling for wide area surveillance and image synthesis,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2000, pp. 160–167.

- [43] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh, “Background modeling and subtraction of dynamic scenes,” in *Proc. IEEE Int’l Conf. Computer Vision*, 2003, pp. 1305–1312.
- [44] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems 14*, 2002.
- [45] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds. MIT Press, 2000.
- [46] G. Rätsch, T. Onoda, and K.-R. Müller, “Soft margins for adaboost,” *Machine Learning*, vol. 42, pp. 287–320, 2001.
- [47] C. Ridder, O. Munkelt, and H. Kirchner, “Adaptive background estimation and foreground detection using kalman-filtering,” in *Proc. Int’l Conf. Recent Advances in Mechatronics*, 1995, pp. 193–199.
- [48] J. Rittscher, J. Kato, S. Joga, and A. Blake, “A probabilistic background model for tracking,” in *Proc. European Conf. Computer Vision*, vol. 2, 2000, pp. 336–350.
- [49] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, Massachusetts: MIT Press, 2002, pp. 469–516.

- [50] M. Seki, T. Wada, H. Fujiwara, and K. Sumi, “Background subtraction based on cooccurrence of image variations,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 65–72.
- [51] Y. Sheikh and M. Shah, “Bayesian modeling of dynamic scenes for object detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1778–1792, 2005.
- [52] J. Shi and J. Malik, “Motion segmentation and tracking using normalized cuts,” in *Proc. IEEE Int’l Conf. Computer Vision*, 1998, pp. 1154–1160.
- [53] A. J. Smola and R. Kondor, “Kernels and regularization on graphs,” in *Proc. Ann. Conf. Computational Learning Theory / Kernel Workshop*, 2003, pp. 144–158.
- [54] C. Stauffer and W. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 1999, pp. 246–252.
- [55] Y.-L. Tian, M. Lu, and A. Hampapur, “Robust and efficient foreground analysis for real-time video surveillance,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 1182–1187.
- [56] M. E. Tipping, “Sparse bayesian learning and the relevance vector machine,” *J. Machine Learning Research*, vol. 1, pp. 211–244, Jun. 2001.
- [57] P. H. S. Torr, R. Szeliski, and P. Anandan, “An integrated bayesian approach to layer extraction from image sequence,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 3, pp. 297–303, 2001.

- [58] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: Principles and practice of background maintenance,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 1, 1999, pp. 255–261.
- [59] D. Wang, T. Feng, H.-Y. Shum, and S. Ma, “A novel probability model for background maintenance and subtraction,” in *Proc. Int’l Conf. Vision Interface*, 2002, pp. 109–116.
- [60] J. Y. A. Wang and E. H. Adelson, “Representing moving images with layers,” *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 625–638, 1994.
- [61] Y. Weiss, “Smoothness in layers: Motion segmentation using nonparametric mixture estimation,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997, pp. 520–527.
- [62] Y. Weiss and E. H. Adelson, “A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1996, pp. 321–326.
- [63] J. Wills, S. Agarwal, and S. Belongie, “What went where,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 37–44.
- [64] C. R. Wren, A. Azarbayejani, T. J. Darrell, and A. P. Pentland, “Pfinder: Real-time tracking of the human body,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 780–785, July 1997.

- [65] J. Xiao and M. Shah, “Motion layer extraction in the presence of occlusion using graph cuts,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1644–1659, 2005.
- [66] H. Yang, Y. Tan, J. Tian, and J. Liu, “Accurate dynamic scene model for moving object detection,” in *Proc. IEEE Int’l Conf. Image Processing*, vol. 6, 2007, pp. 157–160.
- [67] P. Yin, A. Criminisi, J. Winn, and I. Essa, “Tree-based classifiers for bi-layer video segmentation,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007, pp. 1–8.
- [68] J. Zhong and S. Sclaroff, “Segmenting foreground objects from a dynamic textured background via a robust kalman filter,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 1, 2003, pp. 44–50.
- [69] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, “Learning with local and global consistency,” in *Advances in Neural Information Processing Systems 16*, 2004.
- [70] Q. Zhu, S. Avidan, and K.-T. Cheng, “Learning a sparse, corner-based representation for time-varying background modeling,” in *Proc. IEEE Int’l Conf. Computer Vision*, vol. 1, 2005, pp. 678–685.
- [71] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” in *Proc. Int’l Conf. Pattern Recognition*, vol. 2, 2004, pp. 28–31.