N° d'ordre : 4261

THÈSE

PRÉSENTÉE À

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Ronan SICRE**

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : INFORMATIQUE

Analyse vidéo de comportements humains dans les points de ventes en temps-réel

---

Soutenue le : 24 mai 2011

Après avis des rapporteurs :

M. Atilla BASKURT	Professeur, INSA Lyon
M. Ioannis PATRAS	Professeur, Queen Mary University of London

Devant la commission d'examen composée de :

M. Atilla BASKURT	Professeur, INSA Lyon	Rapporteur
M. Jean-philippe DOMENGER	Professeur, Université Bordeaux 1	Président
M. Olivier GAROSI	Directeur Scientifique, MIRANE S.A.S.	Examinateur
M. Henri NICOLAS	Professeur, Université Bordeaux 1	Directeur de thèse
M. Ioannis PATRAS	Professeur, Queen Mary University of London	Rapporteur
M. Quoc Cuong PHAM	Chercheur, CEA	Examinateur

– 2011 –

Acknowledgment

Je souhaite remercier mes encadrant, mes collègues dans l'équipe AIV, au LaBRI et dans l'entreprise Mirane.

Merci à Stéphanie Roquejoffre, ma famille et mes amis.

Merci à l'association Bordo latino swing, les 33 tours, et le service des sport de l'université de bordeaux.

Thanks to the 404 crew for the adventures and crazy memories.

à Juliette, Renée, Georges et Marcel.

Abstract

Along this thesis various subjects are studied, from the lowest to the higher level of video analysis.

We first present motion detection and object tracking that compose the low-level processing part of our system. Motion detection aims at detecting moving areas, which correspond to foreground, of an image. The result of motion detection is a foreground mask that is used as input for the object tracking process. Tracking matches and identifies foreground regions across frames.

Then, we analyze the behavior of the tracked objects, as the mid-level analysis. At each frame, we detect the current state of action of each tracked object currently in the scene.

Finally, the system generates a semantic interpretation of these behaviors and we analyze high-level scenarios as the high-level part of our system. These two processes analyze the serie of states of each object. The semantic interpretation generates sentences when state changes occur. Scenario recognition detects three different scenarios by analyzing the temporal constraints between the states.

Résumé

Cette thèse est effectuée en collaboration entre le LaBRI (Laboratoire bordelais de recherche en informatique) et MIRANE S.A.S., le leader français en Publicité sur Lieu de Vente (PLV) Dynamique. Notre but est d'analyser des comportements humains dans un point de vente.

Le long de cette thèse, nous présentons un système d'analyse vidéo composé de plusieurs procédés de divers niveaux. Nous présentons, dans un premier temps, l'analyse vidéo de bas niveau composée de la détection de mouvement et du suivi d'objets.

Puis nous analysons le comportement des objets suivis, lors de l'analyse de niveau moyen.

Finalement, l'analyse de haut niveau est composée d'une interprétation sémantique de ces comportements et d'une détection de scenarios de haut-niveau.

Publications

Sicre R., Nicolas H.: Analyse vidéo de comportement sur des lieux de ventes, *IEEE SETIT*, 2009.

Simonetto P., Koenig P.-Y., Zaidi F., Archambault D., Gilbert F., Phan Quang T. T., Mathiaut M., Lambert A., Dubois J., Sicre R. et al.: Solving the Traffic and Flitter Challenges with Tulip, *IEEE Symposium on Visual Analytics Science and Technology*, 2009.

Sicre R., Nicolas H.: Shopping behaviour analysis in videos, *International Journal on Graphics, Vision and Image Processing 09, vol. 6.*, 2009.

Sicre R., Nicolas H.: Shopping scenarios semantic analysis in videos, *CBMI*, 2010.

Sicre R., Nicolas H.: Analyse de comportements dans les points de ventes, *CORESA*, 2010.

Sicre R., Nicolas H.: Human Behaviour Analysis and Event Recognition at a Point of Sale, *PSIVT*, 2010.

Sicre R., Nicolas H.: Human behavior analysis at a point of sale, *ISVC*, 2010.

Sicre R., Nicolas H.: Improved Gaussian Mixture Model for the Task of Object Tracking, *CAIP*, 2011.

Contents

List of Figures

Chapter 1

Introduction

Over the last decade, computers and world-wide networks influenced our daily life, mainly by increasing our communication capabilities. Along with computers advances, image and video data is more and more accessible and becomes a common part of people's lives. For instance, most mobile phones are equipped with a camera capable of recording videos. At the same time, fast Internet access and increasing storage capacities enable the exchange of such data. For example in summer 2010, more than 100 millions people were active on Facebook and Youtube received 2 billions visitors every day.

With the growing amounts of data, it becomes more and more interesting to analyze videos. Therefore, applications are developed in various fields of computer vision to automatically process and analyze these videos.

For example, video retrieval in large scale databases currently requires costly manual annotation and Web search engines rely on textual description or tags to find relevant videos.

Another example are video-surveillance applications. Video-surveillance cameras are used for controlling access to special areas, identifying specific people in certain scenes, analyzing crowd flux, detecting anomalies, monitoring traffic, etc.

A further application area is computer games, where video analysis is used as a human-computer interface. Microsoft Kinect is a hardware device combining several sensors: a video camera, an infrared depth sensor, and a multi-array microphone. Several video games use full-body 3-D motion capture, face recognition, and acoustic analysis to play without controller devices and interact in a virtual world.

Motion capture of human actions becomes a standard for character animation in animated movies and for special effects in movies. Human motion analysis is also used for medical applications and analysis and optimization of movements of athletes or dancers.

These examples show that there is a large demand for computer vision systems that analyze and process videos automatically. However, automatic video analysis possibilities are rather limited yet and are far behind the capabilities of humans.

1.1 Marketing problematic and goals

The marketing field has new demands that require computer vision systems. These systems are used to measure digital media and display efficiency. In fact, the marketing field has evolved over the last years. The use of digital media, or digital signage, at points of sales becomes more and more popular. These new technologies bring along the potential to open up new communication channels with customers. However, several technical challenges have to be addressed to make these new channel deliver their message efficiently. For example, media playing advertising clips one after another have no significant impact on customers. It is then of primary concern to identify ideal content and location for these media in order to maximize their impact on customers. Nowadays several software systems help solving these challenges. A few systems track customers, in a video-surveillance context, to obtain statistical information regarding customers' habits and displacement inside shopping malls, such as [25]. Various systems directly calculate the media audience and opportunity to see the media, using face detection [116].

The study introduced in this thesis is along the same lines and aims at improving the impact of digital media by maximizing interaction between media and customers. Furthermore, we want to produce statistical data off the interactions between the customers and products. More specifically, we detect customers picking up products from known areas in real-time using a fixed camera. The detection of such an event results, for example, in playing a clip related to the product.

This work is achieved in collaboration with MIRANE S.A.S. that is the french leader on digital signage.

1.2 Shopping setting

This section presents the shopping setting with more details. As we can see in the previous work in chapter 4 section 4.2, behavior analysis and action recognition are used in various contexts. Several datasets were used as a baseline for many researchers. We categorize four sorts of datasets used for different applications.

First kind of datasets, such as [129] [12] [155], aims at detecting specific actions like people waving, jumping, walking, running, boxing, etc. Videos are mainly taken without camera motion and focus essentially on the actor.

The second type of datasets [78] [133] are directly extracted from movies. These datasets are used to detect human actions such as shaking hands, hugging, answering the phone, etc.

Different kinds of behavior can be detected in a video-surveillance setting [2] [133], like meetings, language drop, crowd analysis, etc.

The last type of datasets concerns sports videos [123]. The configuration varies and can be focused on the actors, extracted from a TV transmission, or similar to surveillance.

Nowadays, only a few studies used dataset coming from shopping environment, i.e. points of sale [54]. The shopping setting is between action recognition such as [129] that observes a person to detect its action or motion and video-surveillance that detects interactions between people, luggage, specific areas of the scene, etc. Thus, we want to detect customers' actions as well as interactions with specific areas of the scene, i.e. products areas.

1.3 Previous work overview

Along this thesis, we present some previous work in various fields of computer vision: motion detection, object tracking, human behavior understanding, human action recognition, and semantic description of behavior. Several surveys offer a good overview of the subject.

[149] presents an overview of motion detection with a focus on global models of the background using data analysis methods. [24] shows a panel of motion detection methods applied to traffic scenes [110] compares several motion detection algorithms in terms of speed and memory requirements. [145] proposes a motion detection and tracking system and compares the proposed detection technique with 9 other methods. [113] presents five pixel-based motion detection methods for outdoor surveillance. [165] is probably the more exhaustive state of the art concerning object tracking. This survey also present motion detection. [53] proposes a clear survey on video-surveillance systems. Motion detection, object tracking, behavior understanding, and semantic description of behaviors are presented. [100] is a survey on trajectory learning and analysis for surveillance. Many behavior analysis systems are based on trajectories, especially in traffic monitoring systems. [96] presents a state of the art on human motion capture and analysis. Various types of motion detections are presented. [153] presents the same topic with a major part dealing with motion detection techniques. [69] shows an overview of human action recognition. [156] and [115] present two survey on the same field. For further information on this topic, the reader can refer to [21], [95], [41], [4], and [147].

1.4 Chosen approach

1.4.1 Tasks

Our first goal is to detect motion to identify areas of interest in the image. We then want to track moving areas, corresponding to people, across frames. Once objects, or people, are detected and identified along the sequence, we define measurements related to their interest in the products from the scene. Then, we define a list of interesting states to recognize. These states are organized in a finite state machine and are first all detected deterministically. We further work on the detection of the "Interact" state, corresponding to a person picking up products. We propose a probabilistic approach to detect this state using various descriptors on the motion in a local spatio-temporal context. We further create two high-level processes that interpret the detected states of each person for non-specialist operators. We generate sentences in natural language to summarize each person's actions and we detect three different scenarios achieved by the customers.

1.4.2 Constraints

It is important to note that we have strong real-time constraints. The system developed has to be fast to generate responses quickly as soon as an event occurs. Furthermore, we want to avoid the use of algorithms that require a long training phase to simplify the installation of the system.

1.4.3 Datasets

To test our methods, we produce several video datasets. The setup is presented as follows: people walk around an indoor scene and grab products on a horizontal rack, such as a table. Several heaps of products are in the scene. We first shoot sequences in our LAB using various products of different sizes, colors, and shapes using various camera locations. Then, we take more videos in a real environment: a shopping mall. In both locations, we separate videos where only one person is acting in the scene at a time and videos with several actors interacting together. We note that we use other datasets, such as video-surveillance and traffic monitoring videos, to test the motion detection and object tracking systems.

1.4.4 Overview

We present our system that is based on a low-level analysis module composed of motion detection and object tracking. Then, mid-level analysis determines the current state of action of

each tracked object. Finally, the high-level analysis module is separated in two processes: semantic description of the objects behavior and recognition of predefined scenarios achieved by these objects, see figure 1.1. Each module is evaluated and we conclude with perspectives.

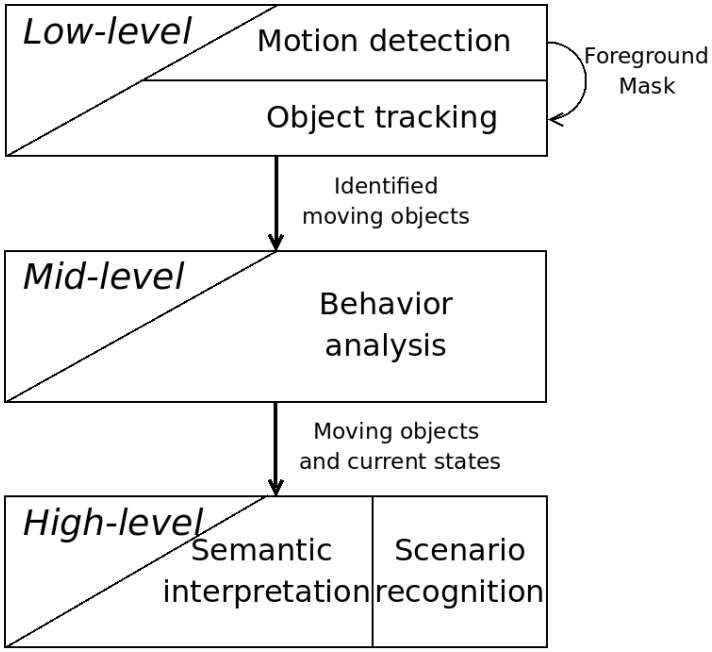


Figure 1.1: System diagram

Introduction

Ces dix dernières années, les ordinateurs et Internet ont influencé notre vie de tous les jours. Ils ont permis d'augmenter nos capacités à communiquer. De même, les images et vidéos sont de plus en plus accessibles et sont utilisées par tous. Par exemple, la plupart des téléphones portables sont équipés d'appareils photo et de caméras. Simultanément, les accès internet à haut débit et les grandes capacités de stockage permettent d'échanger ces données. Par exemple, en été 2010, plus de 100 millions de personnes étaient actives sur Facebook et Youtube recevait 2 milliards de visiteurs chaque jour.

Vu l'augmentation de la taille des données échangées, il devient de plus en plus important d'analyser les vidéos. En effet, des applications sont développées dans divers champs de la vision par ordinateur, afin d'analyser et de traiter automatiquement ces vidéos.

Par exemple, la récupération de vidéos dans des bases de données nécessite des annotations manuelles importantes et les moteurs de recherche Internet se basent essentiellement sur des textes de description et des tags pour trouver des vidéos.

Un autre exemple de type application sont les systèmes de vidéo-surveillance. Les caméras de vidéo-surveillance sont utilisées pour contrôler l'accès à certaines zones, identifier des personnes, analyser des groupes de personnes, détecter des anomalies, analyser le trafic routier, etc.

Les jeux vidéo représentent un autre domaine d'application. L'analyse vidéo sert d'interface entre homme et machine. Par exemple, la caméra Kinect de Microsoft est composée de plusieurs capteurs: une camera classique, un capteur de profondeur infrarouge et un micro. Divers jeux vidéo utilisent des modèles 3-D du corps humain, de la détection de visages et de l'analyse acoustique pour plonger le joueur dans un monde virtuel dans lequel il n'a pas besoin de manette pour interagir.

Les films d'animation utilisent des systèmes de capture de mouvement pour animer des personnages virtuels ou générer des effets spéciaux. L'analyse de mouvements humains est aussi utilisée dans le milieu médical et sportif.

Ces exemples laissent présager une forte demande pour des procédés permettant le traitement automatique de vidéos. Cependant, ces systèmes restent limités et loin des performances d'un opérateur humain.

Le point de vue marketing

Le domaine du marketing au point de vente a évolué avec l'ère de la publicité sur lieu de vente: PLV dynamique ou "Digital Signage". L'utilisation de grands écrans et autres systèmes d'affichage dynamique sont de plus en plus courantes dans les points de ventes. Cette technologie permet un nouveau type de communication avec les clients. Cependant, Ces nouveaux systèmes présentent de nouvelles difficultés. Par exemple, diffuser des clips publicitaires les uns après les autres n'a pas d'impact important sur les clients. Il est donc important d'optimiser la position et le contenu de ces systèmes. De nos jours, il existe quelques logiciels permettant de répondre à ces difficultés. Certains systèmes suivent les clients d'un magasin avec des camera de vidéo-surveillance afin de connaitre les lieux de fort passage dans les magasins [25]. D'autres systèmes permettent de calculer l'audience d'un écran en détectant les visages [116].

Nos travaux portent sur la même problématique: améliorer l'impact des systèmes d'affichages en maximisant les interactions entre l'écran et les clients. De plus, nous souhaitons générer des données statistiques sur les interactions entre les clients et les produits. En particulier, nous détectons les clients qui saisissent des produits dans des zones prédéfinies en temps réel en utilisant une caméra fixe. La détection d'un tel évènement pouvant générer la diffusion d'un clip lié au produit.

Le point de vente

Ce paragraphe présente les spécificités du point de vente. Comme on peut le voir dans la section "Previous work" du chapitre 4, l'analyse de comportement et la reconnaissance d'action sont utilisées dans divers contextes. Il existe beaucoup de vidéos utilisées pour de telles analyses. Nous présentons quatre types de vidéos:

Des vidéos telles que [129], [12] et [155] ont pour but la détection d'actions humaines comme faire des sauts, saluer, marcher, courir, boxer, etc. Ces vidéos sont focalisées sur la personne réalisant les actions et la caméra est généralement fixe.

Le deuxième type de vidéos [78] [133] sont des extraits de films. Dans ce cas, on cherche à détecter des actions telles que répondre au téléphone, se serrer la main, s'enlacer, etc.

Il existe beaucoup de vidéos de vidéo-surveillance [2] [133]. On cherche à y détecter des réunions, des bagages abandonnés, analyser des foules, etc.

Le dernier type de vidéos concerne le sport [123]. Les configurations varient beaucoup: caméra focalisé sur l'acteur, vidéo extraite de diffusion télévisée, etc.

De nos jours, Il n'y a que très peu d'études réalisées dans des points de ventes [54]. Ce type d'environnement se trouve entre les vidéos de reconnaissance d'action et de mouvement [129] et les vidéos de type vidéo-surveillance. Nous cherchons donc à détecter les actions des clients ainsi que leurs interactions avec des zones définies dans la scène, i.e. des zones correspondantes aux produits.

Présentation des travaux précédents

Le long de cette thèse, nous présentons des états de l'art dans plusieurs domaines de la vision par ordinateur: la détection de mouvement, le suivi d'objets, l'analyse de comportement humain, et la description sémantique de comportement. De nombreux articles offrent une vue d'ensemble des ces différents domaines:

[149] présente le domaine de la détection de mouvement et se focalise sur les modélisations globales de l'arrière-plan utilisant des méthodes d'analyse de données. [24] présente les méthodes de détection de mouvement appliquées au trafic routier. [110] évalue les performances en terme de vitesse et mémoire de diverses méthodes de détection de mouvement. [145] propose une méthode de détection de mouvement et de suivi d'objet et la compare avec 9 autres méthodes. [113] présente cinq méthodes de détection de mouvement basées pixel pour la surveillance de scènes extérieures. [165] propose un état de l'art très complet sur le suivi d'objet. La détection de mouvement est aussi présentée. [53] présente une étude des systèmes de vidéo-surveillance très claire. La détection de mouvement, le suivi d'objet, l'analyse de comportement et la description sémantique de comportement y sont présentés [100] propose une étude sur l'analyse et l'apprentissage des trajectoires pour la surveillance. En effet, beaucoup de systèmes d'analyse de comportement sont basés sur les trajectoires des objets, surtout dans des contextes d'analyse du trafic routier. [96] présente un état de l'art sur les systèmes de capture de mouvement ainsi que diverses méthodes de détection de mouvement. [153] présente le même sujet avec une grande partie consacrée à la détection de mouvement [69] présente la reconnaissance d'actions humaines. [156] et [115] proposent deux études sur le même domaine. Pour plus d'information sur ce domaine, se référer à [21], [95], [41], [4], et [147].

Approche choisie

Taches

Le premier but de notre système est de détecter le mouvement pour identifier les zones d'intérêt de l'image. Ensuite, nous suivons ces zones de mouvement le long de la vidéo. Ces zones de mouvement correspondent à des personnes marchant dans la scène. Une fois que ces zones, ou objets, sont détectés et identifiés, nous définissons des mesures liées à leur intérêt pour les produits présents dans la scène. Puis, nous définissons plusieurs états intéressants à reconnaitre et les organisons dans une machine à états finis. Les états sont d'abord détectés de manière déterministe, puis nous détectons l'état "Interact" de manière probabiliste. Cet état correspond à une personne saisissant des produits et est détecté en utilisant des descripteurs basés sur le mouvement dans un contexte spatio-temporel local. De plus, nous créons deux procédés de haut-niveau, qui interprètent les états détectés pour chaque personne. Des phrases en langage naturel sont générées, résumant les actions réalisées par chaque personne. Nous détectons aussi le scenario réalisé par chaque personne, parmi trois scenarios prédéfinis.

Contraintes

Nous devons faire face à de fortes contraintes temps-réel. Le système doit être rapide afin de générer des réponses rapides, des qu'un évènement apparait. De plus, nous souhaitons éviter d'utiliser des algorithmes nécessitant de longues phases d'apprentissage, afin de simplifier l'installation du système.

Données

Plusieurs jeux de données sont générés pour tester nos méthodes. Nous nous plaçons dans la configuration suivante: des personnes se déplacent dans une scène à l'intérieur et saisissent des produits sur des présentoirs horizontaux, telle une table. Plusieurs amas de produits sont disposés sur le présentoir. Nous prenons des vidéos dans notre laboratoire et dans un environnement réel: un point de vente. Dans ces vidéos plusieurs positions de caméra sont utilisées et les produits disposés ont diverses tailles, couleurs et formes. On note, que les vidéos sont classées en fonction du nombre de personnes interagissant avec les produits. Finalement, nous utilisons des jeux de données provenant d'applications de vidéo-surveillance et d'analyse du trafic routier, pour tester la détection de mouvement et le suivi d'objet.

Plan

Nous présentons notre système basé sur une analyse de bas-niveau, composée de la détection de mouvement et du suivi d'objets. Puis, l'analyse de niveau moyen détermine l'état actuel de chaque objet suivi. Finalement, l'analyse de haut-niveau est séparée en deux procédés: la description sémantique du comportement de ces objets et la reconnaissance de scenarios prédéfinis joués par ces objets, voir figure 1.1. Chaque module du système est évalué et nous concluons avec les perspectives.

Chapter 2

Motion detection

2.1 Introduction

This chapter presents motion detection. Motion detection aims at detecting moving regions, in a video sequence, i.e. separate these regions, or foreground, from the background.

Background is mainly composed of stationary areas, such as walls, furniture, buildings, vegetation, ground, etc. However, background is also composed of non-stationary areas: objects shadows, computer screens, fan, running escalators, tree branches, clouds, etc.

Foreground corresponds to moving objects. These objects are of interest and are detected by the motion detection method. Depending on videos, these objects have different size, shape, texture, speed, etc.

Motion detection is often the base for higher level process, such as object tracking, behavior analysis, human motion capture, etc. These applications mostly observe deformable objects like human, rigid objects like vehicles, or both. Depending on what we are observing, we can select relevant methods.

Motion detection aims at detecting changes of the pixel intensity values along time. There are various types of changes:

- **Global changes** are detected all over the image and are due to camera motion or lighting changes.
- **Local changes** often correspond to foreground object, or background non-stationary areas.
- **Once-off changes** can be global or local and can correspond to moving background objects, such as people depositing or taking objects. It is easy to miss some of these changes that can be related to relevant actions. For example, our application focuses on interactions between products that belong to the background and people acting in

the scene. A product taken by a person generates a "once-off" change that must be detected.

The goal is to detect the relevant changes and filter out the irrelevant ones.

Detecting motion presents several challenges, such as:

- Textural similarity between foreground objects and the background: it reduces changes in the images and makes them hard to detect.
- Detection of small objects: they might be considered as noise and are harder to detect.
- Camera motion: even if we assume that the camera is fixed, the camera can be moving because it is not properly fixed, or due to weather conditions in outdoor scenes.
- Lighting changes: they occur when clouds pass or when the camera auto-white-balance adjust the image intensity, for example.
- Complex scenes with partial or complete occlusions: these occlusions can be caused by the background or several people aligned with the camera.

In this chapter, we first review previous works and then present various motion detection methods. These methods are implemented and tested on various video datasets. First, we present a simple method that use image differencing and do not require a background model. Then we generate a model to improve the detections. This model is first calculated with a temporal averaging (TA) method. Then we use a method based on Bayes decision rules (BDR) classification of pixels. Next, we use a Gaussian mixture model (GMM) of the background. Finally, the last method that uses an improved Gaussian mixture model (iGMM) with shadows detection offers the best results and is used for our final application.

2.2 State of the art

Motion detection is a method that aims at detecting areas of an image where motion occur. literature can refer to motion detection as background subtraction, which is the process following the construction of the background model. The current image is compared to the background to obtain foreground areas corresponding to motion.

2.2.1 Taxonomy

Based on the various surveys described in the introduction [149], we can classify motion detection methods in four categories.

- **Detection without background model:** These methods aim at calculating for each pixel a value that is a function of the intensity or the color. These values represent motion intensity.
- **Pixel-based background model:** These methods assign to each pixel of the image a value or an intensity function that models the appearance of the background. We only use the measurement taken on the specific pixel. Most of these methods generate a statistical model. However, models can be generated from a probabilistic process, a predictive filter, or based on a single intensity value per pixel.
- **Local background model:** These methods use the neighborhood of a pixel instead of the pixel itself to calculate the similarity measurement. Specifically, these methods calculate if a block of pixels belongs or not to the background.
- **Global background model:** These methods use the entire image at each frame to build a model of the entire background.

2.2.2 Detection without background model

This section presents three examples of motion detection techniques: image differencing, spacial temporal entropy and optical flow.

2.2.2.1 Image difference

An intuitive way to detect motion in an image is to calculate the difference between two consecutive frames [60]. This method is further presented in chapter 2, section 2.3.1.

2.2.2.2 Spacial temporal entropy

Entropy is a thermodynamic property that is a measure of the energy not available for useful work in a physic process. In our case, we want to calculate the "variability" of the pixels intensity. When the pixel intensities vary a lot, the entropy increases. [88] calculates the entropy based on spatial temporal histograms and generates a Spatial Temporal Entropy Image (STEI). This image highlights the areas where the pixels intensities vary a lot in the temporal domain or in the spatial neighborhood. These areas correspond to motion areas as well as contours areas. In order to only detect motion, spatial variations can be down-weighted. Moreover, [88] proposes to apply morphological filters to improve the STEI.

[62] proposes to use image difference as input to calculate the entropy, instead of the frames.

2.2.2.3 Optical flow

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and the scene. Various papers deal with these methods [51] [87] [8] [50]. [8] and [50] propose surveys on various optical flow algorithms. We note that optical flow calculation requires some computation times and a fast method is required to satisfy real-time constraints. One of the most popular and fast method is [87] that is based on the "image constraint equation".

2.2.3 Pixel-based background model

2.2.3.1 Image model

The simplest way to model the background is an image of this background without foreground objects. This method can detect slow, fast, or even stopped objects. However such methods can not cope with lighting changes in outdoor environment. Moreover, it is not always possible to obtain a background image without foreground objects. It is therefore necessary to update this model over time.

[164] proposes to use image difference to update the background model. The method considers the first frame as an approximation of the background. Then, the difference between the previous and the current image is calculated. Pixels where no difference is observed for a certain amount of time are updated with current image values.

[29] calculates the background model by applying a median filter over N frames. In other words, the median value of the N last frames is selected as foreground for each pixel, see chapter 2 section 2.3.2. [28] improves this method, by using the medoid instead of the median filter.

Finally we present, in chapter 2 section 2.3.2, several temporal averaging methods that correspond to this category of motion detection techniques.

2.2.3.2 Statistical model

Statistical model aims at estimating probability to observe an intensity value on a given pixel. The model is build by learning the appearance of the background and is composed of a set of probability density functions for each pixel of the image. If the current value of a pixel has a high probability, the pixel is considered as background. When the value has a low probability, the pixel corresponds to a foreground area.

[159] and [93] model the background by a Gaussian distribution. For every pixel, the mean and variance are updated recursively. A pixel belongs to the background if its current value is close to the mean of the Gaussian; we also take the variance into account. This

model offers good results for indoor scenes with static background. However, the method is not robust to periodic motion, such as tree branches, because the model is uni-modal.

[136] proposes a multi-modal model by using a mixture of K Gaussian distributions. This method is further described in chapter 2 section 2.3.4.

Later on the Mean-Shift algorithm, used in clustering and segmentation, is used to create a multi-modal model of the background [45]. Based on a set of samples the algorithm detects the modes, and their number, of a complex distribution. However, this method is computationally expensive and can not be run at every frame.

2.2.3.3 Predictive model

Predictive methods are similar to statistic approaches. These methods apply Wiener or Kalman filter to predict the value that should be observed at a given pixel coordinates. The distance between the estimate and the current value is used to calculate the motion amplitude. These methods handle non-static backgrounds.

[145] presents a method operating on three levels: pixel-level, local-level, and global-level. The pixel-level segmentation is achieved by using a Wiener filter [47].

Most methods use Kalman filter [159]. However, many methods use different state vector to describe the system. [159] uses Kalman filter to calculate the parameters of the Gaussian, which model the background. [15] simply uses the intensity values as state vector, [121] uses the temporal derivative, and [72] uses the spacial derivative.

2.2.4 Local background model

2.2.4.1 Region detection

Local models use the neighborhood of a pixel instead of the pixel itself to calculate the similarity measurement. [122] builds a hidden Markov model (HMM) [117] based on 3x3 pixels blocks covering the entire image. Specifically, this method is based on the intensity mean of the block and the gradient and detects, for each block, one of the three predefined states: The block corresponds to background, foreground, or a shadow.

2.2.4.2 Texture

[48] uses Local Binary Pattern (LBP) code to calculate the pixels texture.

Similarly to [136], the model is composed of a set of K weighted modes. Modes correspond to LBP histograms calculated on each block. At a new frame, a new histogram h is calculated for each block. Then, h is compared to the K histograms of the model. If there is

no histogram matching h, the histogram with lower weight is replaced by h. If a histogram m is close to h, m is updated as well as the model weights.

Finally, only the b histograms with the higher weights, which are larger than a threshold, are considered as modeling the background. If h do not match any of these b histograms, the area represented by h is considered as foreground.

2.2.4.3 A posteriori improvement

Several methods [145], [35], and [144] use multiple levels of detection. First pixel-based motion detection is achieved. Then, these results are analyzed to improve the consistency of the detected regions.

[145] first detects moving regions with a Wiener filter. Then, a segmentation process improves the regions contour.

[35] uses the local neighborhood of each foreground pixel to detect false detections, due to small background motion or camera motion. On each foreground pixel, the system calculates the probability that the pixel correspond to a neighbor background. This probability is high when the motion is due to a slight background motion. However, this value is also high on real motion of objects that have similar texture as the background. This case is taken into account by doing the same test over the entire connected foreground region. The entire region must correspond to background motion to be considered as background.

[144] uses optical flow to improve the detection: areas where the optical flow had major variations in the past frames are deleted from the foreground.

2.2.5 Global background model

2.2.5.1 Model mixture

[145] wants to take the entire scene into account during the segmentation process. Thus, k models are calculated with "k-means" algorithm on a set of images, used as training. Then, pixel-based motion detection is achieved on each model. The model with the less motion detected is selected as the final result. We note that the models are updated when a large amount of pixels are detected as foreground.

[137] proposes a HMM that switches between several motion detection techniques to cope with dynamic environments. This model is build to handle fast light changes. For example, the HMM has two state (day/night) associated with two different pixel-based model of the background.

2.2.5.2 Eigen background

[102] presents a method for motion detection using a data analysis technique. The image sequence is represented as a three-dimensional matrix. The background is modeled by using N non-consecutive frames. Then, the covariance matrix is calculated and diagonalized to obtain the Eigen values. The M first Eigen values are kept in a sub-matrix Φ to model the background. We can reconstruct an image representing the background from the matrix Φ and compare it with the current image to obtain the Distance From Feature Space (DFFS) [97], or foreground.

Conclusion

The detection methods presented in this chapter use a pixel-based model of the background, except for image differencing. In fact, these models offer precise results, i.e., at a pixel level.

2.3 Evaluated Approaches

2.3.1 Image difference

One of the basic ideas to detect motion is image differencing, or frame differencing. This method aims at calculating the difference between two, or more, consecutive frames.

Method description and evaluation

First, we calculate the difference between two consecutive frames on each channel. Then, we average this difference over the three channels and binarize the result, see figure 2.1 and 2.3 (b). The following equations define the calculation of the difference between two consecutive frames: $Diff_2$ and how the result Res is binarized.

$$Diff_2(x,y,t) = |I(x,y,t-1) - I(x,y,t)| \tag{2.1}$$

$$\begin{aligned} if\ Diff(x,y,t) > T\ then\ Res(x,y,t) = 1 \\ else\ Res(x,y,t) = 0 \end{aligned} \tag{2.2}$$

Where $I(x,y,t)$ represents the pixel value at coordinates (x,y) at time t in the image sequence.

It is interesting to note that this method tends to detect motion on the contour of a person. Areas covered by the person in the current frame that were corresponding to background in the previous frame are well detected. However, parts of the person in the previous frame that becomes background in the current frame are also detected. These parts do not correspond to the person in the current frame. Furthermore, when a person is wearing relatively homogeneous clothes a major part of the person is not detected, see figure 2.1. Thus, a person that stops moving, or moves slightly is not detected anymore, see figure 2.5. This undetected parts correspond to the areas covered by the person in the two consecutive frames. These detections result in a shifting effect, i.e. motion is detected between the current person position and its previous position.

(a) frame (b) foreground

Figure 2.1: Frame differencing: previous frame

(a) frame (b) foreground

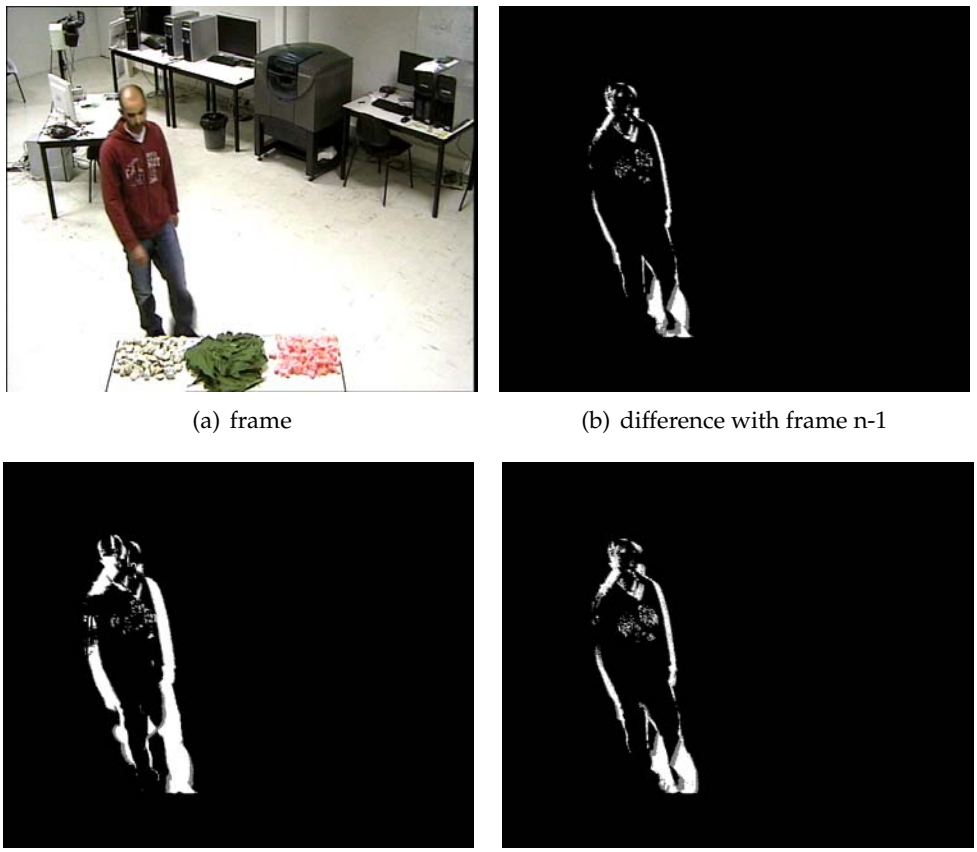
Figure 2.2: Frame differencing: previous and next frames

A first improvement to this technique is to compare the current image n to an older image $n-2$ or $n-3$. Following the same method, we detect larger areas of motion on both the moving object and the background, see figure 2.3 (c).

Another improvement of these techniques is done by adding to the first result the difference between the current frame and the next frame, as we see in the following equation.

$$Diff_3(x,y,t) = Diff_2(x,y,t) \cup Diff_2(x,y,t+1) \tag{2.3}$$

Similar effects are noted as in the previous case, such as motion detected on the side of the person, see figure 2.2. Furthermore, ghost effects appear when a person moves fast, see figure 2.4. However, the shifting effect is compensated, since motion is detected on both sides of the person, see figure 2.3 (d).



(a) frame (b) difference with frame n-1

(c) difference with frame n-2 (d) difference with frame n-1 and n+1

Figure 2.3: Comparison of three methods

(a) frame (b) difference with frame n-1 and n+1

Figure 2.4: Ghost effect due to fast motion

(a) frame (b) difference with frame n-1 and n+1

Figure 2.5: The person is stopped, only its arm is moving and is detected

Conclusions

These techniques are very fast: the processing of one 704x576 frame takes about 0.01 second on a Pentium M, 1.73 GHz with 500 Mb of RAM. We note that all the motion detection techniques in this chapter are tested in the same conditions.

However, these techniques share several limitations. The detections are not precise: the object is not fully detected and motion is detected on the background. If a person stops moving, there is no motion detected anymore. In the opposite case, when a person moves fast, ghost effects appear.

The use of a background model appears to be an interesting method to solve several issues.

2.3.2 Temporal averaging

Following the idea that a background model improves motion detection results, we build a background model by calculating the mean over several images while no object is in the scene.

2.3.2.1 Method description

We calculate the average value of each pixel along the three channels. The following equation shows how to calculate the background model M from the images of the sequence.

$$M(x,y,t) = \sum_{i=t-T}^{t} \frac{I(x,y,i)}{T} \tag{2.4}$$

T represent the frame interval, used to refresh the background.

Then we test a median filtering technique. Similarly to the previous method, we save the last T frames. For each pixel, we sort the T values and use the $\frac{T}{2}$th value to model the background.

$$M(x,y,t) = median_i\ I(x,y,i), with\ i = [t-T,...,t]. \tag{2.5}$$

Finally, we approximate the first solution 2.4 using an Infinite Impulse Response filter:

$$M(x,y,t) = \frac{T-1}{T} M(x,y,t-1) + \frac{1}{T} I(x,y,t) \tag{2.6}$$

This approximation allows us to save time and memory since we do not require to save the last T frames. Once the model is generated, we calculate the difference between the current image and the model. If there is a high difference, pixel are considered as being part of the foreground, i.e. corresponding to a moving region. The following equations show how the distance *Dist* between the current frame and the model is calculated and how the result *Res* is found.

$$Dist(x,y,t) = |M(x,y,t) - I(x,y,t)| \tag{2.7}$$

$$\begin{aligned} if\ Dist(x,y,t) > T\ then\ Res(x,y,t) = 1 \\ else\ Res(x,y,t) = 0 \end{aligned} \tag{2.8}$$

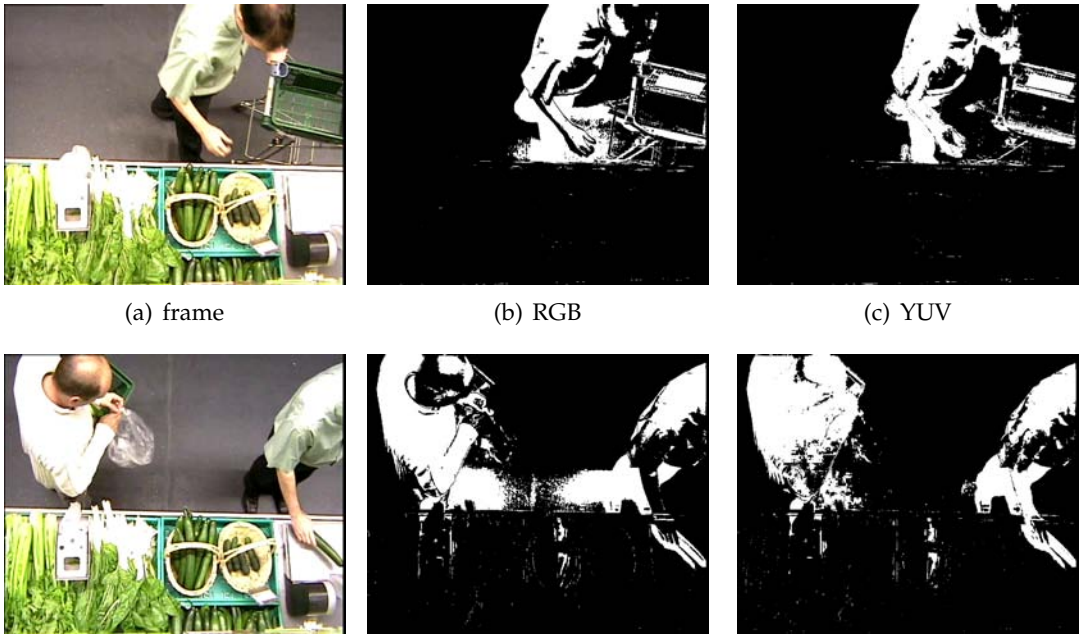
As we see on figure 2.6 (b) and (e), shadows generate false positive with such a model. Thus, we converted the images into YUV color space. Y stands for the luminance component, i.e. brightness, and U and V are the chrominance, or color, components. Since luminance and chrominance are separated on different channels, only the luminance channel is really sensitive to shadows. Furthermore, chrominance channels receive a higher weight than the luminance channel in order to reduce, even more, the shadows effects, see figure 2.6 (c) and (f). Then, the distance along the three channels together $Dist_{total}$ is calculated as follows:

$$Dist_{total}(x,y,t) = \sum_{i=1}^{3} \omega_i Dist_i(x,y,t) \tag{2.9}$$

Where $i = [1,2,3]$ represents the different channels and ω_i are the weight for each channel. The chrominance weights ω_2 and ω_3 are larger than the luminance weight ω_1. In our case $\omega_1 = 1/7$ and $\omega_2 = \omega_3 = 3/7$. However using this technique, small illumination variations on highly textured areas can lead to false positives, see figure 2.7.

It is interesting to note that the background model is further improved in order to detect people stopping in the scene. Using the temporal averaging technique, if a person enters the scene and stops moving, the person slowly disappears and becomes part of the background. Depending on the value of T, the disappearance can take seconds or minutes. But this problem is of major interest, since people tends to stop often in a shopping environment. In fact, people do not have homogeneous trajectories and look for products, wander around, look at prices, etc.

We assume that the background remains relatively stationary while being occluded by an object. Then, we can save the model values before the occlusion occurs and keep it identical while occluded by a moving object. Thus, we do not refresh areas of the model where foreground objects are detected, as well as its close neighborhood. Specifically, each pixel of the result mask outputted by the tracking process, see chapter 3, is tested to determine if the corresponding pixel of the background model should be refreshed or not.



(a) frame (b) RGB (c) YUV

(d) frame (e) RGB (f) YUV

Figure 2.6: Temporal averaging technique: sensibility to shadows comparison between RGB and YUV color space

2.3.2.2 Evaluation

We comment on the value of T in the equation 2.6. We first used small values such as $T = 5$. Small values do not produce a stable background model and large value, like $T = 100$, build a static model that can not adapt to general changes. We noted empirically that values between $T = 15$ and $T = 50$ were offering better results because the background model is stable against some level of noise and can adapt in case of general illumination changes.

We note interesting phenomenon due to the camera auto-white balance and auto-iris. These two automatic preprocess can generate false detections in specific cases. When very big objects with textures that are much darker, or brighter, than the background enter the scene the auto-white balance increases, or decreases, the value of each pixel of the image. Similarly, the auto-iris let more, or less, light enter the sensor. This event also results in an increase, or a decrease, of each pixel value. Such phenomenon can lead to false detections on highly textured areas, see figure 2.7

Also, in several datasets, we can observe periodic camera motion. Such motion generates a shifting of a couple of pixels over the entire image, generating motion close to edges in the entire image, see section 2.5 figure 2.16.

(a) frame (b) RGB (c) YUV

Figure 2.7: Temporal averaging technique and issues on highly textured area when light changes occur: comparison between RGB and YUV color space

Since our model is uni-modal, such noises (rapid modification of the pixels values) can lead to false detections over the image. These false detections mostly occur on highly textured areas, for lighting changes, and on inhomogeneous areas, for camera motion.

Although these automatic processes can be monitored and the camera is supposed to be fixed, a multi-modal model of the background can improve the detections and avoid these false detections.

Finally, we note that the time to process one frame is around 0.02 second.

Conclusions

Using a background model, the detection is fast and much more precise than the image differencing technique, since the entire person is correctly detected.

Shadows are well handled, but false-detections occur with light changes or when camera motion occurs. A multi-modal background model can probably offer a solution to these issues.

2.3.3 Bayes decision rules for classification

This section presents an elaborated method to describe the background [80].

Specifically, this method aims at formulating Bayes decision rules in order to classify background and foreground using selected feature vectors. Stationary background objects are described with color features, while moving background objects are described with color co-occurrence features. Then, foreground objects are extracted from the fusion of the classification results from both stationary and moving pixels.

The first idea is that background pixels may have multiple states in complex environment. Therefore these different parts of the background should be analyzed with different

features. Unlike most methods, [80] uses a general Bayesian framework based on various features to model the background

2.3.3.1 Problem formulation and representation

Let v_t be a feature vector extracted from the image sequence at the pixel coordinate $s = (x, y)$ and at time t. The following posterior probability of v_t from the background b or foreground f is calculated using Bayes rules:

$$P(C|v_t, s) = \frac{P(v_t|C, s)P(C|s)}{P(v_t|s)}, C = b, or f \tag{2.10}$$

Then a pixel belongs to the background if:

$$P(b|v_t, s) > P(f|v_t, s) \tag{2.11}$$

Since a feature vector is either associated to the background or the foreground,

$$P(v_t|s) = P(v_t|b, s).P(b|s) + P(v_t|f, s).P(f|s) \tag{2.12}$$

substituting 2.10 and 2.12 in 2.11, we obtain

$$2P(v_t|b, s).P(b|s) > P(v_t|s) \tag{2.13}$$

This relation allows us to determine if a feature vector v_t is associated with the background of the foreground.

$P(v_t|s)$ and $P(v_t|b, s)$ can be represented by the histograms of feature vectors over the entire feature space. For a feature vector with n dimension and L quantization levels, the entire histogram for $P(v_t|s)$ or $P(v_t|b, s)$ would contain L^n bins. If L or n have large values, operating on such a histogram would be expensive on computation and memory. Therefore, we look for a good approximation.

The background contains mainly static areas, while interest objects often move in the scene. If the selected features effectively represent the background, then background features vectors would concentrate in a very small subspace of the feature histogram. Therefore, foreground feature vectors would distribute widely in the feature space. In other words, with a good feature selection, we can cover a large percentage (more than 90%) of the fea-

ture vectors associated with the background using a small number of bins in the histogram. Furthermore, less features vectors from foreground objects would be distributed in these few bins.

Let $P(v_t^i|b,s)$, with $i = 1, ..., N$, be the first N bins from the histogram sorted according to decreasing order of $P(v_t|b,s)$. For given percentage values $M_1 = 90\%$ and $M_2 = 10\%$, there is a small integer N_1 that satisfies:

$$\sum_{i=1}^{N_1} P(v_t^i|b,s) > M_1 and \sum_{i=1}^{N_1} P(v_t^i|f,s) > M_2 \tag{2.14}$$

N_1 depends on the selected features and the number of quantitative levels used for the features.

A table of feature statistics $S_{v_t}^{s,t,i}$, with $i = 1, ..., N_2$ ($N_2 > N_1$), is maintained for each type of feature vectors. This table record the statistics for the N_2 most significant values. Each element of the table contains three components:

$$\begin{aligned} S_{v_t}^{s,t,i} &= p_v^{t,i} = P(v_t^i|s) \\ p_{vb}^{t,i} &= P(v_t^i|b,s) \\ v_t^i &= [a_1^i, ..., a_n^i]^T \end{aligned} \tag{2.15}$$

We note that the elements in the list are sorted by decreasing values of $p_v^{t,i}$. The first N_1 elements are enough to cover the main part of the feature vectors from the background. Therefore, these elements together with $p_b^{s,t}$ are used to classify foreground and background changes. The elements from N_1 to N_2 are used as a buffer to learn the new significant features through the background updating phase.

We select the feature vectors as follows: When a background pixel is considered stationary, we select its colors as feature vectors Meanwhile, the color co-occurrences of the inter-frame changes are selected as feature vectors for moving background pixels. In fact for a moving background object, even if the color has major variations, color co-occurrence of the inter-frames change is relevant since similar changes occur in the same location of the image.

We note that $S_{c_t}^{s,t,i}$ and $S_{cc_t}^{s,t,i}$ are maintained for each pixel to represent the multiple states of the background pixels.

2.3.3.2 Method description

This section describes the various parts of the algorithm, see figure 2.8. As we can see on this diagram, the system is composed of four parts: change detection, change classification, foreground objects segmentation, and background learning and maintenance. The light blocks represent the first three steps, from the left to the right. The gray blocks correspond to the background modeling step.

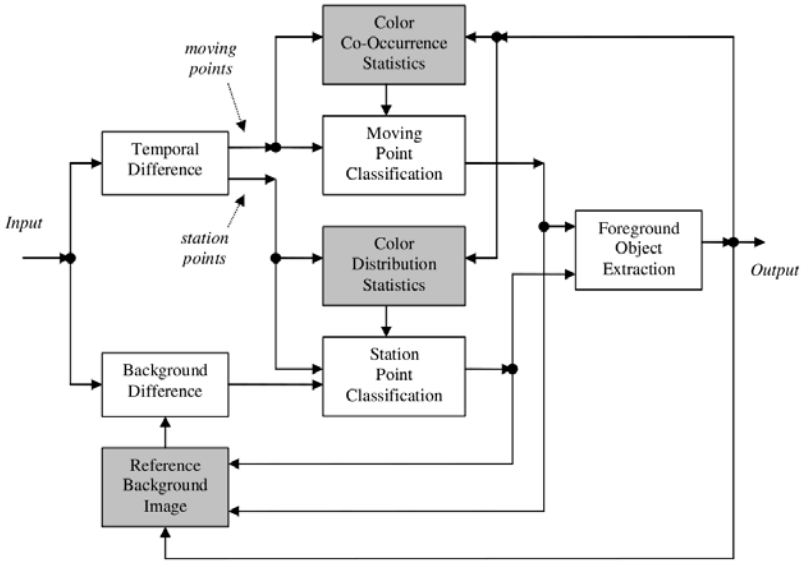


Figure 2.8: Block diagram of the algorithm

2.3.3.2.1 Change detection This module aims at filtering out pixels of insignificant change. We first calculate the background difference $F_{bd}(s,t)$ and the temporal difference $F_{td}(s,t)$ using [126].

2.3.3.2.2 Change classification Based on the result of change detection, we classify pixels as foreground or background. Specifically, if $F_{td}(s,t) = 1$, pixel is classified as a motion pixel. Otherwise, the pixel is considered as a stationary one. Then we need to determine if these pixels belong to the background or to the foreground. For stationary pixels, the color feature vector $v_t = c_t = [r_t, g_t, b_t]^T$ is generated. c_t is the color component at time t and has $L = 64$ levels. For motion pixels, the feature vector of color co-occurrence $v_t = cc_t = [r_{t-1}, g_{t-1}, b_{t-1}, r_t, g_t, b_t^T$ is generated with $L = 32$ levels. We compare this feature vector with v_t with the first N_1 learned features from the corresponding table of feature

statistics for background. Let $v_t = [a_0, ..., a_n]^T$ and v_t^i from the table $S_{v_t}^{s,t,i}$ 2.15. The conditional probabilities are obtained as follow:

$$\begin{aligned} P(b|s) &= p_b^{s,t} \\ P(v_t|s) &= \Sigma_{j \in M(v_t)} p_v^{s,t,j} \\ P(v_t|b,s) &= \Sigma_{j \in M(v_t)} p_{vb}^{s,t,j} \end{aligned} \tag{2.16}$$

Where the matched feature set in $S_{v_t}^{s,t,i}$ is defined as

$$M(v_t) = k : \forall m \in 1, ..., n, |a_m - a_m^k| \leq \delta \tag{2.17}$$

With $\delta = 2$ to retrieve the statistics if the similar features are quantized into neighboring vectors.

If no element of $S_{v_t}^{s,t,i}$ matches v_t, $P(v_t|s)$ and $P(v_t|b,s)$ are set to 0. We use the probabilities calculated in 2.16 into 2.13 to classify pixels as background or foreground.

2.3.3.2.3 Foreground object segmentation Morphological filters are applied on the foreground image to remove noises and improve objects' shape and generate an output image $O(s,t)$.

2.3.3.2.4 Background learning and maintenance This process aims at adapting the background model to the various changes occurring over time. There are two main parts in this section, updating the tables of feature statistics and the reference background image.

2.3.3.2.4.1 Updating tables of feature statistics Two tables of color and color co-occurrence statistics are maintained for each pixel. Two updating process are generated to adapt the background to gradual and "once-off" changes.

Gradual changes: Feature statistics for color and color co-occurrence are updated as follows:

$$\begin{aligned} p_b^{s,t+1} &= (1-\alpha_2) p_b^{s,t} + \alpha_2 M_b^{s,t} \\ p_v^{s,t+1,i} &= (1-\alpha_2) p_v^{s,t,i} + \alpha_2 M_v^{s,t,i} \\ p_{vb}^{s,t+1,i} &= (1-\alpha_2) p_{vb}^{s,t,i} + \alpha_2 (M_b^{s,t} \wedge M_v^{s,t,i}) \end{aligned} \tag{2.18}$$

For $i = 0, ..., N_2$, where α_2 is the learning rate. Matching labels are generated as follows: $M_b^{s,t} = 1$ when s is labeled as background from the final segmentation, otherwise $M_b^{s,t} = 0$. $M_v^{s,t,i} = 1$ when v_t^i is the closest match for v_t and $M_v^{s,t,i} = 0$ for the others.

If there is no match between v_t and the elements of the table, the N_2th element in the table is replaced by a new feature vector.

$$p_v^{s,t+1,N_2} = \alpha_2, p_{vb}^{s,t+1,N_2} = \alpha_2, v_t^{N_2} = v_t. \tag{2.19}$$

Once-off changes: When a "once-off" change occurs, the feature of the new background appearance becomes dominated immediately after this change. From 2.14 and 2.12, new background feature is detected if:

$$P(f|s) \sum_{i=1}^{N_1} P(v_t^i|f,s) > T \tag{2.20}$$

Where T is the percentage value that determines when the new features can be recognized as new background appearance.

As in section 2.3.2, large value for T offers a stable system. However, this system is responding slowly to "once-off" changes If T is small, the system easily learns the frequent foreground features as new background appearance. T was set to 90%.

$P(f|s)$ prevents updating from a small number of features. Using 2.12 and 2.15 in 2.20, we obtain:

$$\sum_{i=1}^{N_1} p_v^{s,t,i} - p_b^{s,t} \sum_{i=1}^{N_1} p_{vb}^{s,t,i} > T \tag{2.21}$$

Once the new background is detected, features are updated as follows:

$$\begin{aligned} p_b^{s,t+1} &= 1 - p_b^{s,t} \\ p_{vb}^{s,t+1,i} &= (p_v^{s,t,i} - p_b^{s,t}.p_{vb}^{s,t,i}) / p_b^{s,t+1} \end{aligned} \tag{2.22}$$

With $i = 1, ..., N_1$. This operation converts the observed domination features into the learned background features.

Convergence: It is interesting to note that the sum of probabilities for background features converges to 1, as long as the background features are significant and more frequent than foreground features.

Parameter selection for learning: In this paragraph, we focus on the selection of the learning rate α_2. α_2 should be small to adapt to gradual changes and not be perturbed by noises and foreground objects. However, α_2 should not be too small to response to "once-off" background changes.

2.3.3.2.4.2 Updating the reference background image A reference background image is maintained at each time step. This image represents the latest observation of the background.

For stationary background objects, we use an Infinite Impulse Response filter to update the image. For a stationary pixel s, the background is calculated as follows:

$$B_c(s,t+1) = (1-\alpha_1)B_c(s,t) + \alpha_1 I_c(s,t) \tag{2.23}$$

Where $c = r, g, b$. If $O(s,t) = 0$ and $F_{td}(s,t) = 1$ or $F_{bd}(s,t) = 1$, a background change is detected. Then, the pixel is replaced by the new background appearance.

$$B_c(s,t+1) = I_c(s,t), for\ c \in r, g, b \tag{2.24}$$

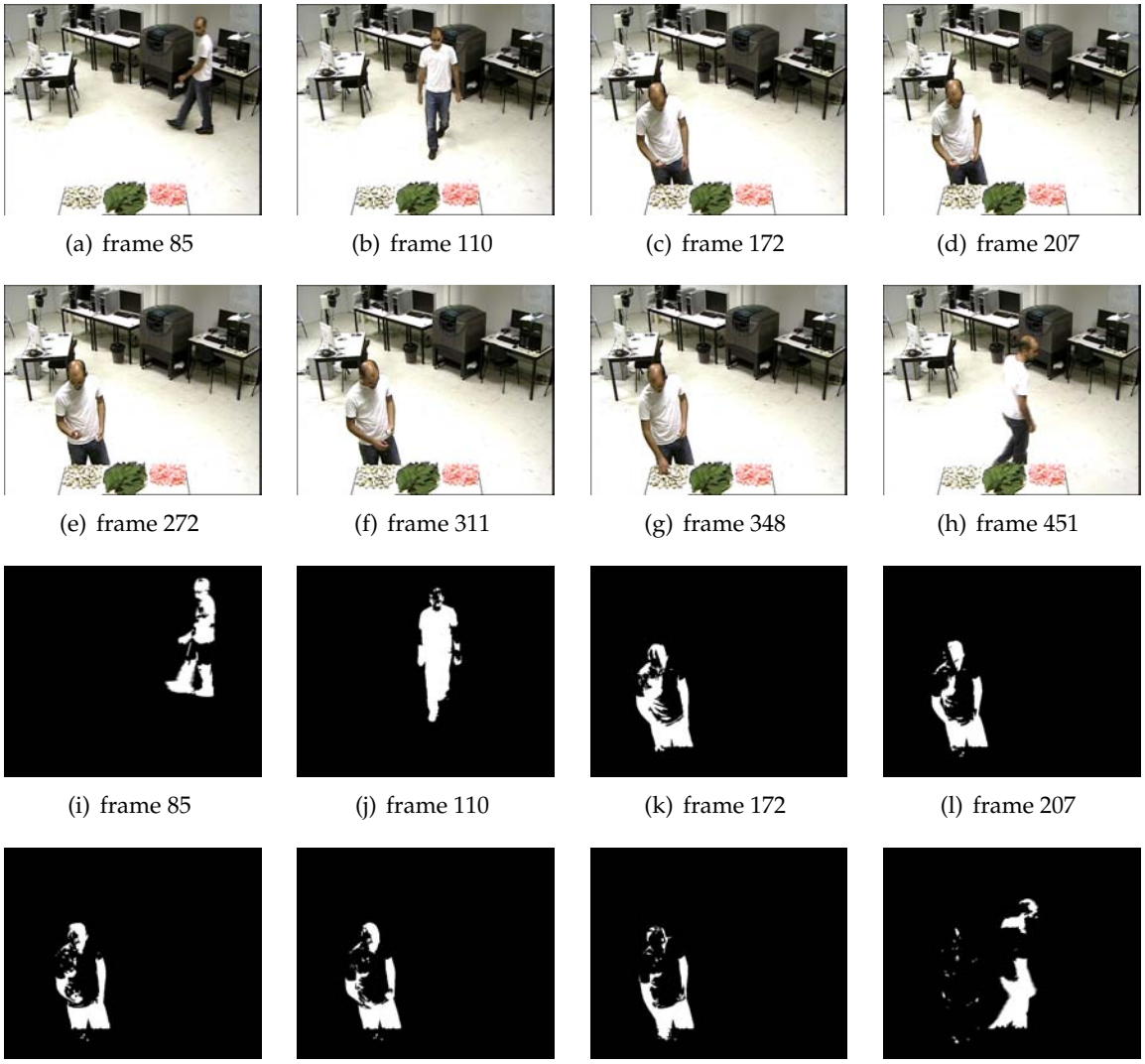
With this operation, the reference background image follows the background motion.

2.3.3.3 Evaluation

This method offers very accurate results, see figure 2.9 and 2.10. Having a background model based on the classification of moving and stable pixels improves the detection and is robust to camera motion and lighting changes, see section 2.5 figure 2.16.

We note that the method handles stopped people. However, a person leaving a location after remaining static for a certain amount of time can generate some noise, see figure 2.10.

We note that the processing of one frame last for 0.3 to 0.4 seconds.



(a) frame 85 (b) frame 110 (c) frame 172 (d) frame 207

(e) frame 272 (f) frame 311 (g) frame 348 (h) frame 451

(i) frame 85 (j) frame 110 (k) frame 172 (l) frame 207

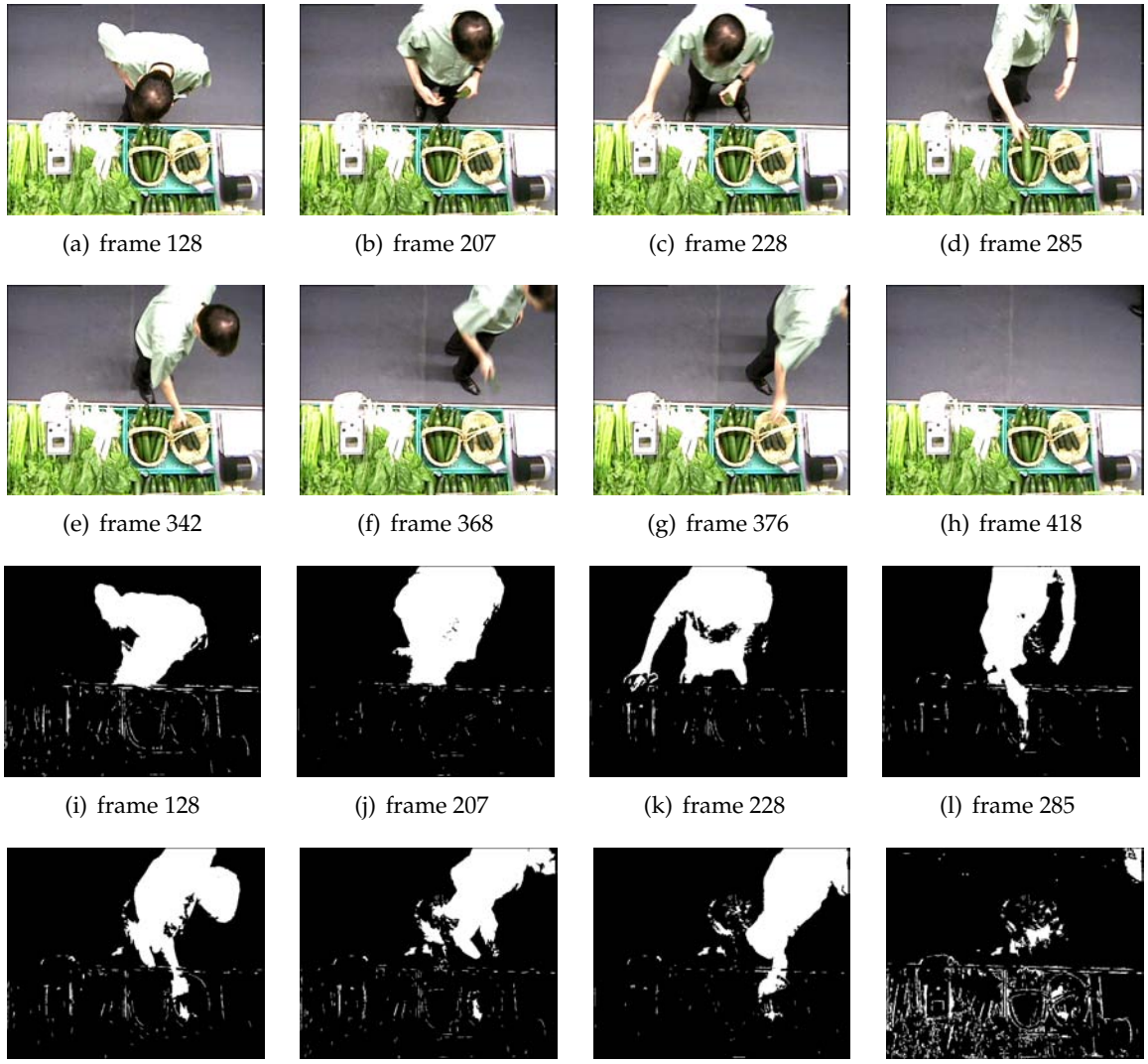
(m) frame 272 (n) frame 311 (o) frame 348 (p) frame 451

Figure 2.9: BDR detection, Dataset LAB 1 video1

Conclusions

This method offers very good results. Typical noises, such as camera motion and lighting changes, happening in the video datasets are well handled. However, noises are detected when a person stops in a location for some time and then leaves.

Meanwhile the computation expenses are high, compared to the other methods. We keep in mind that one of our primarily concern is to build a fast system able to produce a real-time analysis. Thus, we continue evaluating other methods.



(a) frame 128 (b) frame 207 (c) frame 228 (d) frame 285

(e) frame 342 (f) frame 368 (g) frame 376 (h) frame 418

(i) frame 128 (j) frame 207 (k) frame 228 (l) frame 285

(m) frame 342 (n) frame 368 (o) frame 376 (p) frame 418

Figure 2.10: BDR detection, dataset MALL 1 video 3

2.3.4 Gaussian mixture model

This section presents and tests a method that use a mixture of Gaussian to model the background [136]. Each pixel value is used to update the background model on-line. If the pixel does not match one of the Gaussian distributions, then the pixel is considered as foreground.

2.3.4.1 Method description

We consider a video as a series of images varying along the time. $I(s,t)$ represents the pixel value at time t at the coordinates $s = (x,y)$. We model the recent history of each pixel by a mixture of K Gaussian distributions. The probability of observing the current pixel value is:

$$P(I(s,t)) = \sum_{k=1}^{K} \omega_{k,t} * \eta(I(s,t), \mu_{k,t}, \Sigma_{k,t}) \tag{2.25}$$

Where K is the number of distributions, $\omega_{i,t}$ the weight of the ith Gaussian in the mixture at time t, $\mu_{i,t}$ is the mean value of the ith Gaussian at time t, $\Sigma_{i,t}$ is the covariance matrix of the ith Gaussian at time t, and η is a Gaussian probability density function defined as follows:

$$\eta(I(s,t), \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(I(s,t)-\mu_t)^T \Sigma^{-1} (I(s,t)-\mu_t)} \tag{2.26}$$

Stauffer and al. [136] advise to fix the value of K between 3 and 5.

The covariance matrix is defined as follows:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{2.27}$$

In this case, we assume that red, green, and blue channels are independent and have the same variance. Once the model is generated, the new pixel value $I(s,t)$ has to be matched to one of the K Gaussian distributions. Thus we use K-means approximation.

We note that a pixel value has to be within 2.5 standard deviation of a distribution to be matched. If none of the distributions is matched, the least probable distribution is replaced by a new one with the current value of the pixel, a high variance, and a low weight.

The following equation defines the weight of the K distributions at time t.

$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{2.28}$$

Where α is the learning rate and $M_{k,t}$ is equal to 1 for the matched distribution and 0 for the others. $1/\alpha$ corresponds to the number of frame that are taken into account to model the background.

Although μ and σ parameters of unmatched distributions do not change. The matched distribution is updated as follows:

$$\mu_t = (1-\rho)\mu_{t-1} + \rho I(s,t) \tag{2.29}$$

$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho(I(s,t)-\mu_t)^T(I(s,t)-\mu_t) \tag{2.30}$$

with

$$\rho = \alpha\eta(I(s,t)|\mu_k,\sigma_k) \tag{2.31}$$

The next goal is to determine the distribution that is the most likely to be generated by the background. The distribution must have the most evidence and the least variance. Thus, Gaussian distributions are ordered by the value ω/σ. Once the list is ordered, the B first distributions are selected as follows:

$$B = argmin_b\left(\sum_{k=1}^{b}\omega_k > T\right) \tag{2.32}$$

Where T is the measure of the minimum portion of the data that should be accounted for by the background. A small value for T is chosen for an uni-modal background model. Higher T allows multi-modal distributions due to background motion, lighting changes, etc.

2.3.4.2 Evaluation

This method offers interesting results and is more robust to periodic motion and light changes, compared to the temporal averaging technique, see section 2.5 figure 2.16.

However, we note a couple of limitations. The detection is very sensitive to shadows and stopped objects tend to disappear after a certain amount of time, see section 2.5 figure 2.13, 2.14, and 2.15.

We note that the processing of one frame last for 0.25 to 0.35 seconds. However our implementation is not very efficient, concerning the computation time, and can be improved.

Conclusions

The Gaussian mixture model solves the main issues of the temporal averaging method. This method handles camera motion as well as lighting changes due to a multi-modal model of the background

However this method is sensitive to shadows and stopped objects disappear.

2.4 Chosen approach: Improved Gaussian mixture model

This section presents an improved method of [136]. The Gaussian mixture model is first improved by adapting the number of selected Gaussian in the mixture [170]. Then, we use a shadow detection process [66] to filter them out. Finally we modify the updating process of the background model to cope with objects stopping in the scene.

2.4.1 Adapting the number of selected Gaussian

We first quickly go through the work of [136] presented in the section 2.3.4. The probability of appearance of a pixel value at time t is presented in equation 2.25. This model corresponds to a sum of Gaussian functions η with weights $\omega_{i,t}$. η is declared in equation 2.26. The three equations 2.28, 2.29, and 2.30 respectively show how the model parameters are updated along the image sequence. Finally, equation 2.32 selects the B most reliable distribution in the mixture.

We now want to adapt the number of selected distribution B by modifying the calculation of the distribution weight ω_k. In fact, this method can then adapt to the local nature of the data. For example, some areas of an image tend to be very homogeneous across frames, where others tend to be subject to noises. An adaptive model can better feet the data.

The weight ω_k is the part of the data that belong to the kth component, or distribution, of the GMM. This weight can be considered as the probability that a sample comes from the kth component of the GMM. Therefore, ω_k defines an underlying multinomial distribution.

Let us assume that we have t data samples and that each of them belong to one of the components of the GMM. Then, we calculate the number of samples that belong to the kth component: $n_k = \sum_{i=1}^{t} M_{k,i}$ Where $M_{k,i}$ is defined in the section 2.3.4. This assumed multinomial distribution for n_k gives a likelihood function: $L = \prod_{k=1}^{K} \omega_k^{n_k}$. The mixing of weights is summing up to one. Then, a Lagrange Multiplier λ is introduced. The maximum likelihood (ML) estimate follows from: $\frac{d}{d\omega_k}(logL + \lambda(\sum_{k=1}^{K} \omega_i - 1)) = 0$. By getting rid of λ, we obtain:

$$\omega_{k,t} = \frac{n_k}{t} = \frac{1}{t}\sum_{i=1}^{t} M_{k,i} \tag{2.33}$$

The weight $\omega_{k,t}$ can also be rewritten in a recursive manner, based on $\omega_{k,t-1}$ and $M_{k,t}$.

$$\omega_{k,t} = \omega_{k,t-1} + \frac{1}{t}(M_{k,t} - \omega_{k,t-1}) \tag{2.34}$$

Now, by fixing $\frac{1}{t}$ to the value of $\alpha - \frac{1}{T}$, we found the equation 2.28. By fixing the influence of the new samples, we rely more on the new samples and old samples weight is decreased in an exponentially decaying manner.

We introduce prior knowledge for the distribution by using its conjugate prior, the Dirichlet prior.

$P = \prod_{k=1}^{K} \omega_k^{c_k}$

For the multinomial distribution, c_k represents the prior evidence (in the maximum a posteriori (MAP) sense) for the class k, i.e. the number of samples that belong to that class a priori. According to [170], we use negative coefficients $c_k = -c$. In fact, negative prior evidence means that the class k exists only if there is enough evidence of the data. The MAP solution that include the prior comes from: $\frac{d}{d\omega_k}(logL + logP + \lambda(\sum_{k=1}^{K} \omega_k - 1)) = 0$, with $P = \sum_{k=1}^{K} \omega_k^{-c}$. Then, we obtain:

$$\omega_{k,t} = \frac{1}{O}(\sum_{i=1}^{t} M_{k,i} - c) \tag{2.35}$$

Where $O = \sum_{k=1}^{K}(\sum_{i=1}^{t} M_{k,i} - c) = t - Kc$. Then, we get:

$$\omega_{k,t} = \frac{\overline{\Pi_k} - c/t}{1 - Kc/t} \tag{2.36}$$

with $\overline{\Pi_k} = \frac{1}{t}\sum_{i=1}^{t} M_{k,i}$ is the ML estimate from 2.33 and the bias from the prior is introduced through c/t. For larger datasets, or large t, the bias decreases. However, if a small bias is acceptable we can fix it to $c_T = c/T$, with a large value for T. By replacing c_T in 2.35, we obtain:

$$\omega_{k,t} = \omega_{k,t-1} + 1/t(\frac{M_{k,t}}{1 - Kc_T} - \omega_{k,t-1}) - 1/t\frac{c_T}{1 - Kc_T} \tag{2.37}$$

Since we use only a few components K and c_T is small, we assume that $1 - Kc_T \approx 1$. We replace $1/t$ with α and obtain:

$$\omega_{k,t} = \omega_{k,t-1} + \alpha(M_{k,t} - \omega_{k,t-1}) - \alpha c_T \tag{2.38}$$

We use this equation to update the weights of each distribution instead of 2.28. We note that according to the method presented in [136] section 2.3.4, weights are normalized after each update. The Dirichlet prior with negative weights suppress the distributions that are not supported by the data. All distributions with negative weights get discarded.

To conclude, the model can better adapt to various types of areas within the same image and improves the detection.

2.4.2 Shadows detection and removal

An important issue of motion detection methods remains the sensibility to shadows. In fact, shadows can often lead to false detections. The first idea to reduce this limitation was to use a different color space that reduces the susceptibility, as in section 2.3.2. Such a representation improves the results, however the light areas tend to generate false detections. Furthermore, switching image color space requires some computation time.

Then, we use a new method that detects shadows [66]. The idea is to define a heuristic to label Gaussian components as moving shadows. We stay in the RGB color space, to save computation time, and look for a color model that is able to separate chromatic and brightness components. Such a model has to be compatible with the Gaussian mixture model. We use a low-computation color model similar to [52].

In fact, we compare foreground pixels against current background components. If the difference in both chromatic and brightness components is within some thresholds, pixels are considered as shadows. Specifically, we use the mean of the background: E, an expected chromaticity line: $\|E\|$, a chromatic distortion: d, a brightness threshold: τ. For the pixel value I, the brightness distortion: a, and the color distortion: d are calculated as follows, from the background model.

$$a = argmin(I - zE)^2 \tag{2.39}$$

$$c = \|I - aE\| \tag{2.40}$$

Since we assume that each component of the mixture has a spherical Gaussian distribution, the standard deviation of the kth component σ_k is set equal to d. A foreground pixel is considered as a shadow if a is within 2.5 standard deviations and $\tau < c < 1$. The brightness threshold τ is set to 0.45 in our case.

For conclusion, the method can detect shadows and remove them to improve the detection results.

2.4.3 Stopped object handling

In most of our video datasets, people tend to stop in the scene to look at products, or their prices. After some time immobile a stopped person is not detected as a foreground object anymore and becomes a part of the background.

We modify the updating process for the distributions parameters: we do not refresh areas that are considered as a tracked object. Tracked objects are defined in chapter 3. We introduce F_t that is a binary image representing foreground object, as they are further defined. F_t is a filtered foreground image where regions that were tracked for several frames, or objects, are displayed. Pixels covered by an object have value 1 while the others have value 0. It is interesting to note that these objects are slightly dilated to compensiate a slight shifting effect due to the morpholigical filtering. As a result, the close neighborhood of a region is not updated.

We modify the distribution parameters updating equations 2.38, 2.29, and 2.30. They now have the form:

$$\omega_{k,t} = \omega_{k,t-1} + (1 - F_t)(\alpha(M_{k,t} - \omega_{k,t-1}) - \alpha c_T) \tag{2.41}$$

$$\mu_t = \mu_{t-1} + (1 - F_t)\rho(I(s,t) - \mu_{t-1}) \tag{2.42}$$

$$\sigma_t^2 = \sigma_{t-1}^2 + (1 - F_t)\rho((I(s,t) - \mu_t)^T(I(s,t) - \mu_t) - \sigma_{t-1}^2) \tag{2.43}$$

Where ρ is defined in section 2.3.4.

To conclude, this final improvement allows us to detect people stopping for a long period of time in the scene.

2.4.4 Evaluation

This method improves the results obtained with the original work of [136]. The adaptive background and shadows detection remove a lot of noises, see figure 2.11 and 2.12. On these figures, shadows are represented as grey pixel, while white pixels correspond to the actual detection.

Stopped objects are well handled and there is no noise when a person leaves after a stop, see figure 2.11 and 2.12. However, the foreground object image has to be correctly created. Only meaningful objects should be represented. If noises are on the foreground image, they might be detected over several frames.

This methods has a few limitations and does not detect very precisely small objects as we see in section 2.5, figure 2.17.

Finally, the implementation of the algorithm is improved and allows the process of one frame in 0.05 to 0.08 second.

Conclusions

The new model can better adapt to various types of areas within the same image and improves the detection. The method can detect shadows and remove them to improve the detection results. People stopping for a long period of time are detected.

We note that this method requires a good foreground filtered image.

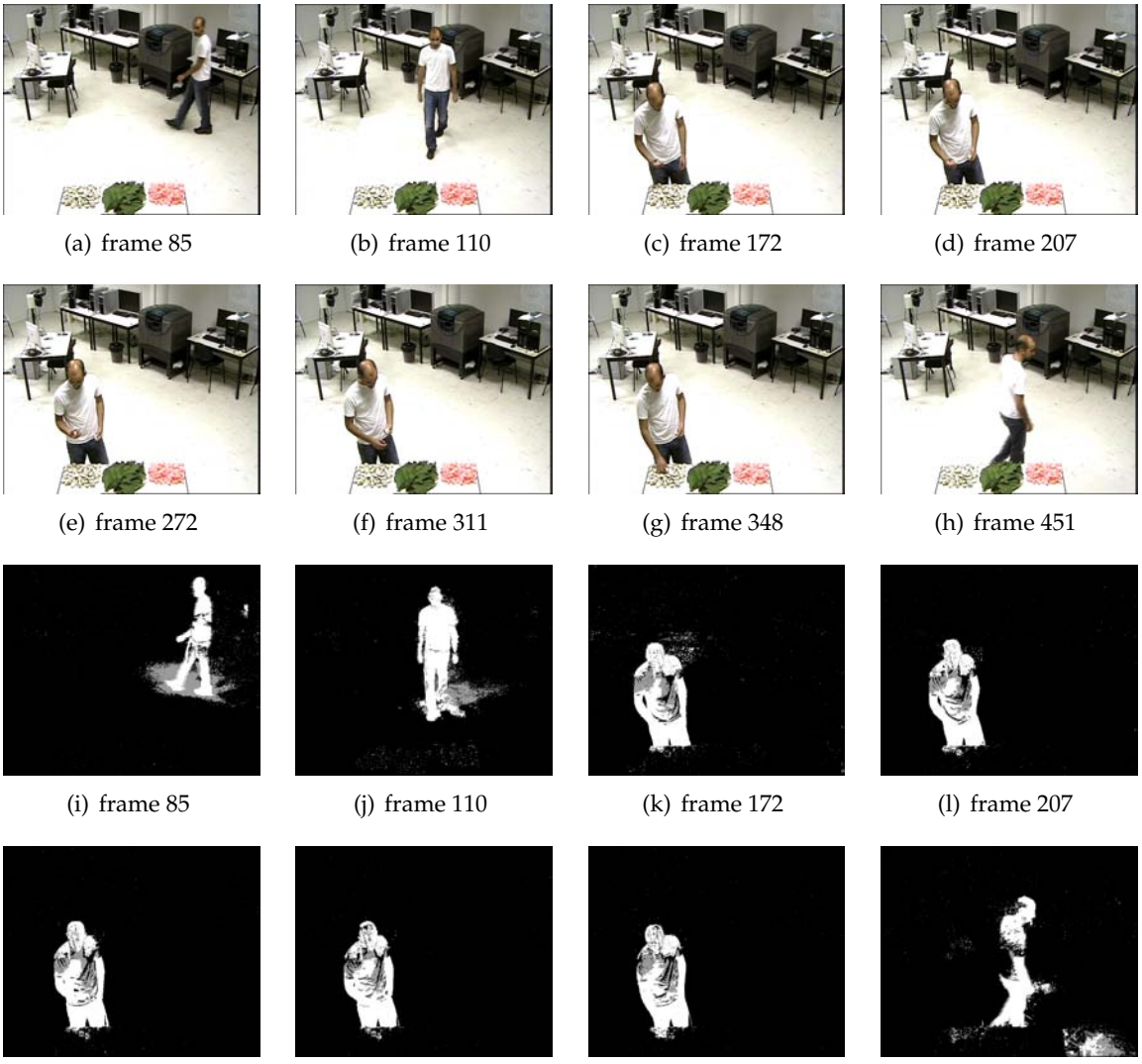
This method offers a good compromise between quality and speed.

2.5 Comparative evaluation and notes

This section proposes several tests that compare the different methods presented.

2.5.1 General comparison

We first compare the different methods on the same video sequence to highlight the differences in detection, see figure 2.13. We especially note that the Gaussian Mixture Model is highly sensitive to shadows. Otherwise, the other methods offer relatively similar results.



(a) frame 85 (b) frame 110 (c) frame 172 (d) frame 207

(e) frame 272 (f) frame 311 (g) frame 348 (h) frame 451

(i) frame 85 (j) frame 110 (k) frame 172 (l) frame 207

(m) frame 272 (n) frame 311 (o) frame 348 (p) frame 451

Figure 2.11: iGMM detection LAB 1 Video 1

2.5.2 Stopped people

Two different sequences are tested for stopped people, see figure 2.14 and 2.15. We note that GMM does not handle this cases and the person disappears. Although the iGMM seems to have a noisy detection, most of these noises are classified as shadows.

2.5.3 Camera motion

Camera motion is difficult to show. At a specific frame, each method reacts differently according to its model. Then, a method supposed to handle motion can generate poor results

(a) frame 128 (b) frame 207 (c) frame 228 (d) frame 285

(e) frame 342 (f) frame 368 (g) frame 376 (h) frame 418

(i) frame 128 (j) frame 207 (k) frame 228 (l) frame 285

(m) frame 342 (n) frame 368 (o) frame 376 (p) frame 418

Figure 2.12: iGMM detection MALL 1 video 3

on a single frame. This is the reason why, we test several frames in a row during camera motion, see figure 2.16. For this test, we only draw foreground pixel for the iGMM method and erase the shadows to make the observation clear. We note that iGMM detects noises, the modification of the model make it less robust to camera motion.

2.5.4 Small objects

We have used video sequences from PETS 2004 to test our system on other videos. This specific sequence offers an interesting challenge: the detection of left luggage, see figure 2.17. We compare the detection results for temporal averaging, BDR, and iGMM. It is interesting

to note that only BDR detects the bag left in the scene. However, BDR also detects a lot of noises in the image due to sun light shinning through windows.

2.5.5 Complex scene

We further test several video sequences from PETS 2004, IBM dataset, PETS 2002, and PETS 2001.

PETS 2004 dataset is composed of several videos taken in a corridor of a shopping mall, see figure 2.18. iGMM and BDR have similar results

Then, IBM dataset has many interactions between people and a complex background, see figure 2.19. Once again results are similar for the two methods.

PETS 2002 dataset has a complex scene taken behind a window and filming small objects, see figure 2.20. Here iGMM is less sensitive to noises than BDR.

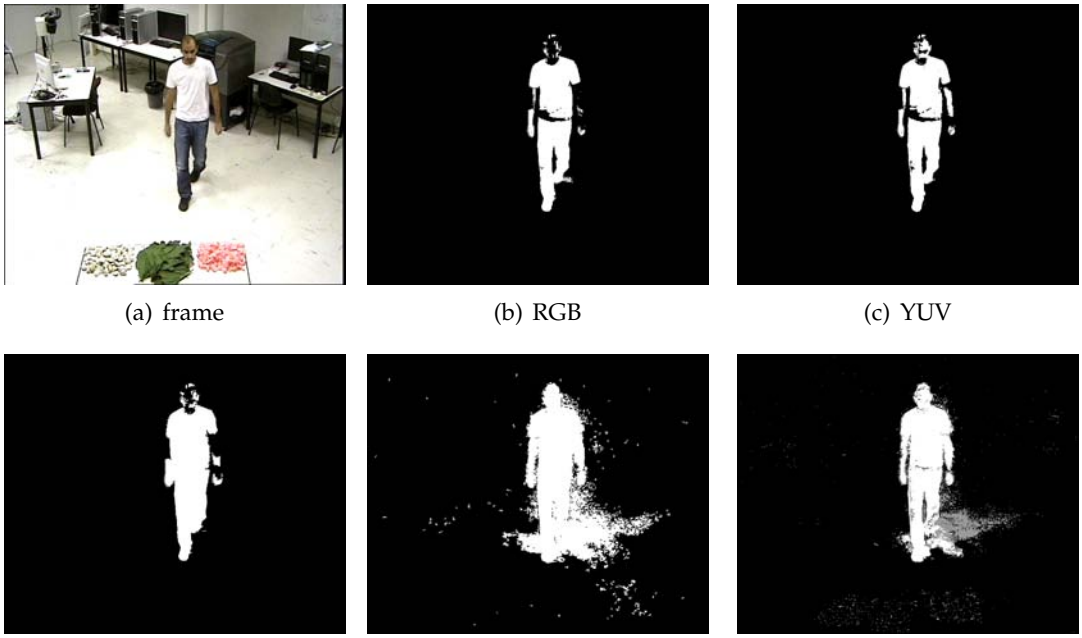
Finally we tested videos from PETS 2001. This video is taken outdoor and observes both pedestrian and cars, see figure 2.21. We observe that a car stopped for several hundred frames disappears for iGMM while it remains detected for BDR, at frame 2000. Also there are detected noises for BDR when the same car leaves.

2.5.6 Morphological filtering

It is interesting to note that we apply morphological filters on the final detection results. We first realize an opening process that removes noises. Then, closing improves the regions shape and fills the holes.

2.5.7 Conclusions

This section presents various tests. Several issues are studied to test the robustness of our methods. These issues correspond to stopped people, camera motion, small objects, and complex scenes.



(a) frame (b) RGB (c) YUV

(d) Bayes (e) GMM (f) iGMM

Figure 2.13: Detection results using each method on LAB 1 video 1 frame 110

2.6 Conclusions

This chapter presents motion detection. We present and evaluate several methods.

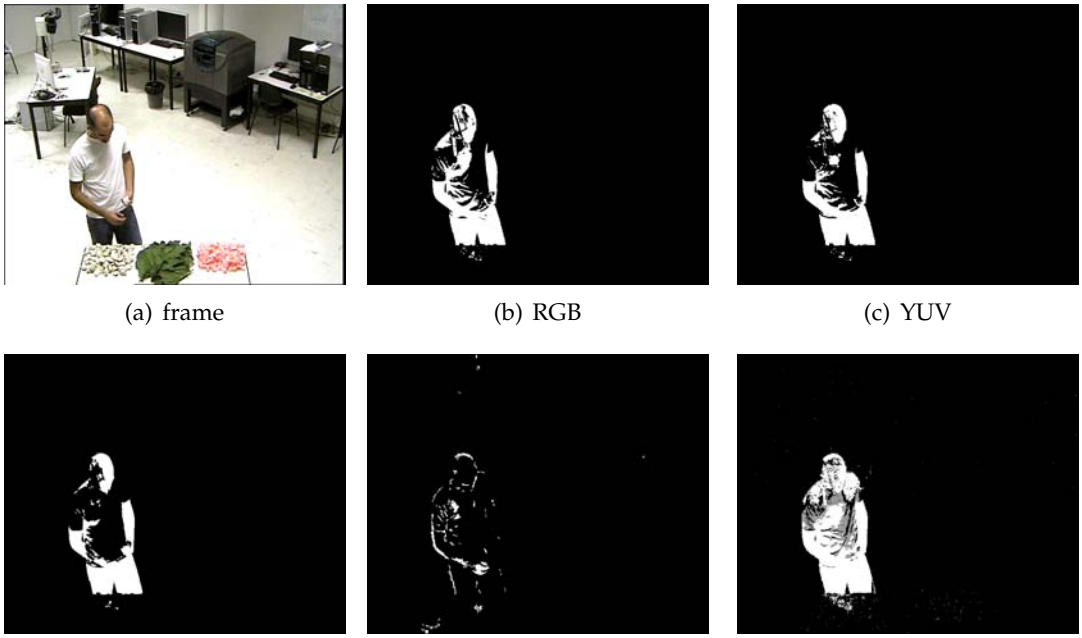
First, frame differencing simply calculates the difference between consecutive frames without background model. Although this method is very fast, it does not allow to entirely detect moving objects and the detection is really noisy.

Secondly, temporal averaging builds a background model by calculating the average of several frames. This method is also very fast and offers a full and accurate detection of the objects. However there are several limitations due to camera motion and lighting changes.

Therefore we test approaches using a multi-modal model of the background.

[80] proposes an interesting method based on Bayes decision rules that classify foreground and background pixels. This method offers very accurate detection of the moving objects even when camera motion and lighting changes occur. However, the method is much slower than the other methods.

Then, [136] proposes to model the background using a mixture of Gaussian distribution. The original version of this algorithm is faster than [80]. Although the method can handle light changes and camera motion, the method is very sensitive to shadows and stopped objects disappear after a while.



(a) frame (b) RGB (c) YUV

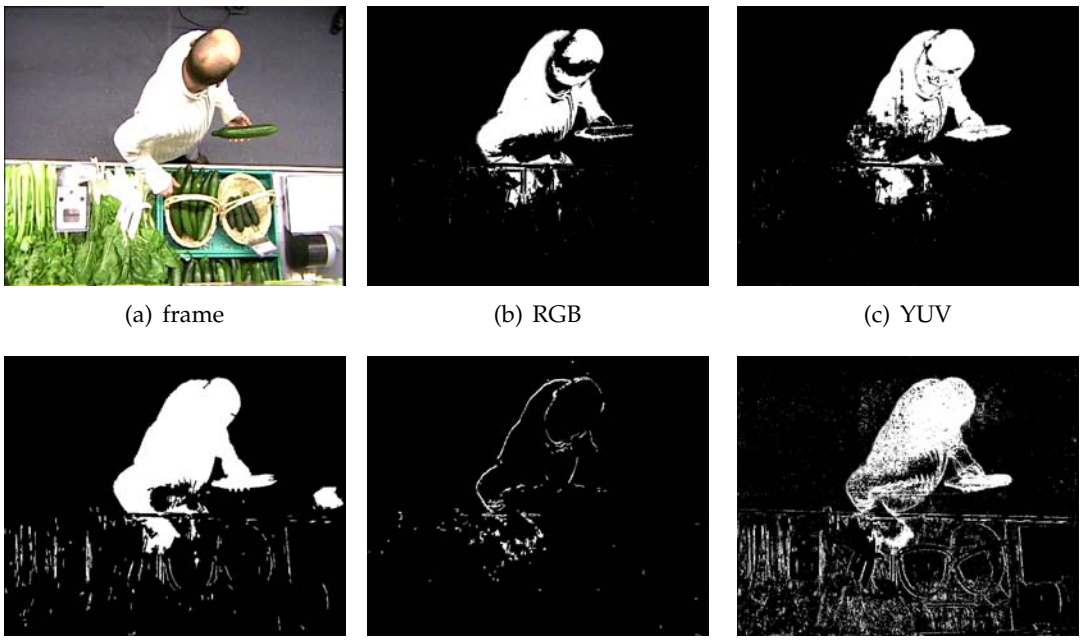
(d) Bayes (e) GMM (f) iGMM

Figure 2.14: Detection when a person stops for about 100 frames

Finally, an improved version of [136] is proposed. We modify the model to adapt the number of selected Gaussian distributions, add a shadows detection module to remove them, and modify again the model to handle people stopping in the scene. This method offers accurate results and handles camera motion, light changes, shadows, and stopped objects.

Furthermore, iGMM is about five times faster than BDR. This method has also limitations in very complex environments, as BDR, but offers us the best compromise between quality and speed.

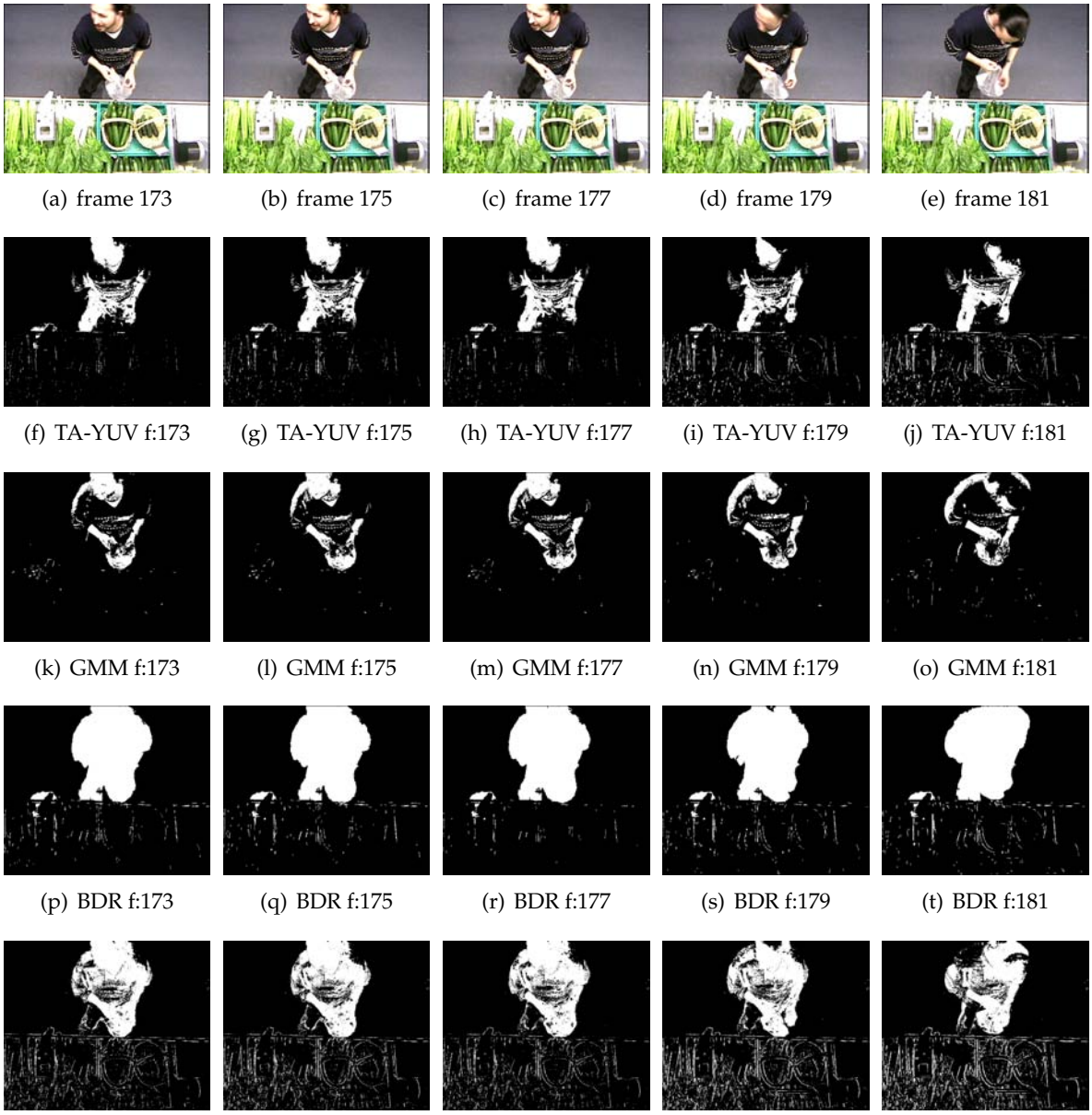
It is interesting to note that in many video sequences, parts of the moving objects are not detected. Detections tend to be noisy when there is textural similarity between moving objects and background. Therefore a merging post-process is required in order to track each person as one entity.



(a) frame (b) RGB (c) YUV

(d) Bayes (e) GMM (f) iGMM

Figure 2.15: Detection when a person stops in a real environment



(a) frame 173 (b) frame 175 (c) frame 177 (d) frame 179 (e) frame 181

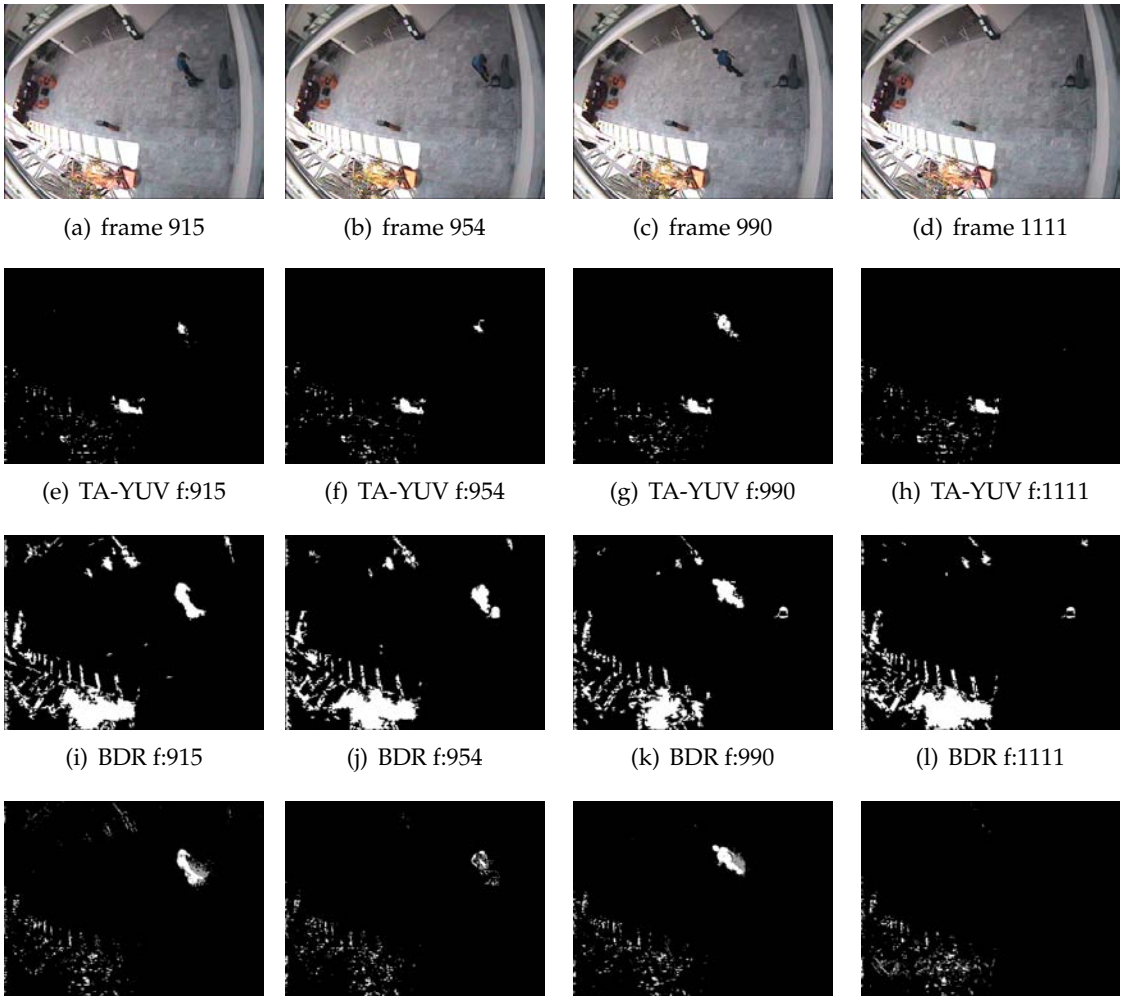
(f) TA-YUV f:173 (g) TA-YUV f:175 (h) TA-YUV f:177 (i) TA-YUV f:179 (j) TA-YUV f:181

(k) GMM f:173 (l) GMM f:175 (m) GMM f:177 (n) GMM f:179 (o) GMM f:181

(p) BDR f:173 (q) BDR f:175 (r) BDR f:177 (s) BDR f:179 (t) BDR f:181

(u) iGMM f:173 (v) iGMM f:175 (w) iGMM f:177 (x) iGMM f:179 (y) iGMM f:181

Figure 2.16: Effect of camera motion



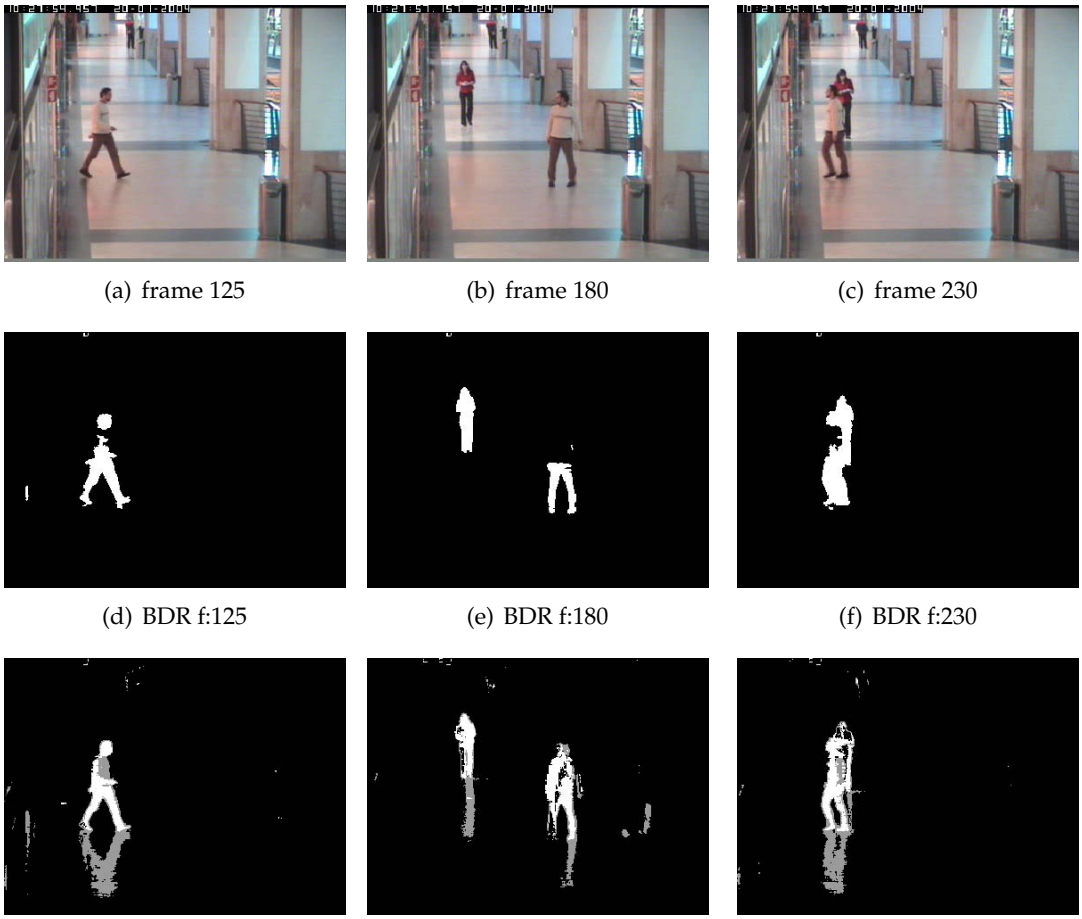
(a) frame 915 (b) frame 954 (c) frame 990 (d) frame 1111

(e) TA-YUV f:915 (f) TA-YUV f:954 (g) TA-YUV f:990 (h) TA-YUV f:1111

(i) BDR f:915 (j) BDR f:954 (k) BDR f:990 (l) BDR f:1111

(m) iGMM f:915 (n) iGMM f:954 (o) iGMM f:990 (p) iGMM f:1111

Figure 2.17: Example of detection of small object: Left luggage, dataset PETS 2004

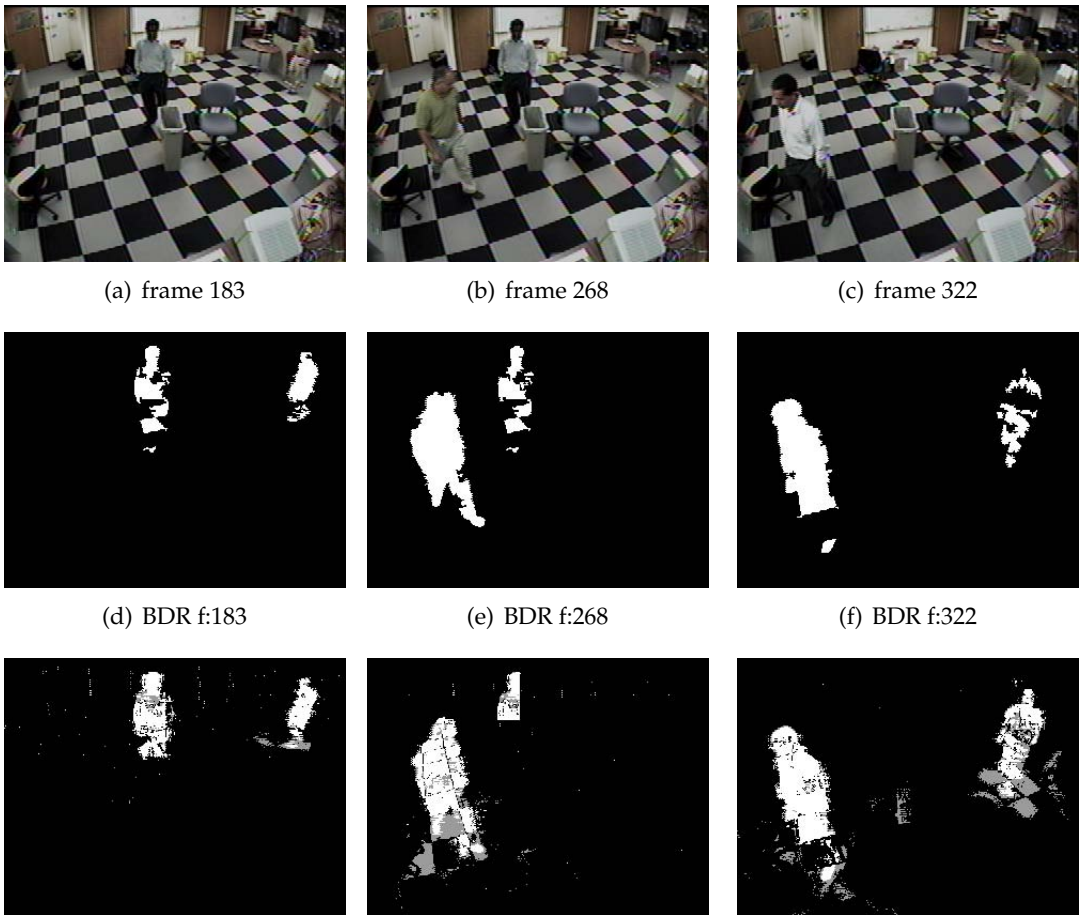


(a) frame 125 (b) frame 180 (c) frame 230

(d) BDR f:125 (e) BDR f:180 (f) BDR f:230

(g) iGMM f:125 (h) iGMM f:180 (i) iGMM f:230

Figure 2.18: Example of detection in a shopping mall corridor, dataset Caviar 2004

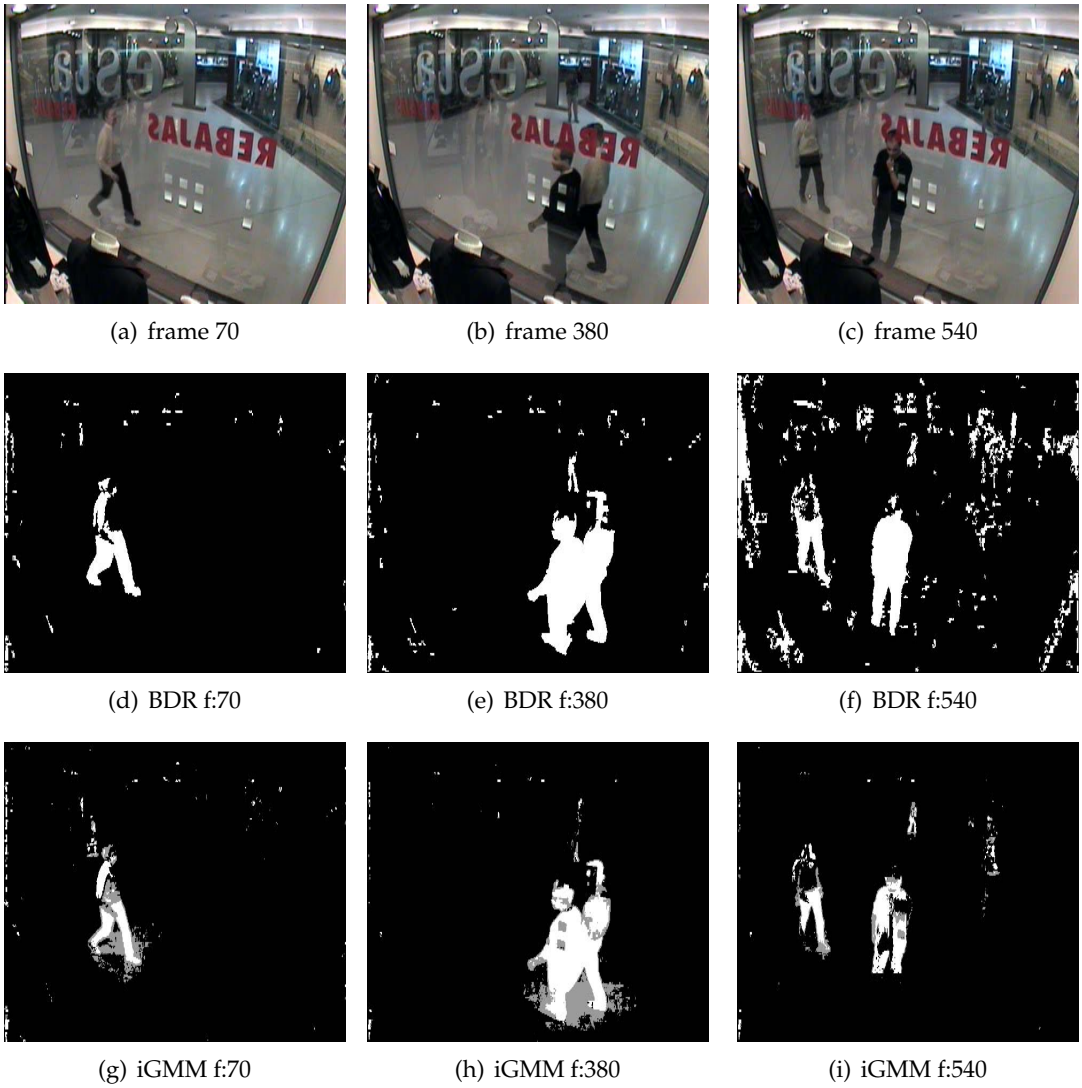


(a) frame 183 (b) frame 268 (c) frame 322

(d) BDR f:183 (e) BDR f:268 (f) BDR f:322

(g) iGMM f:183 (h) iGMM f:268 (i) iGMM f:322

Figure 2.19: Example of detection dataset IBM

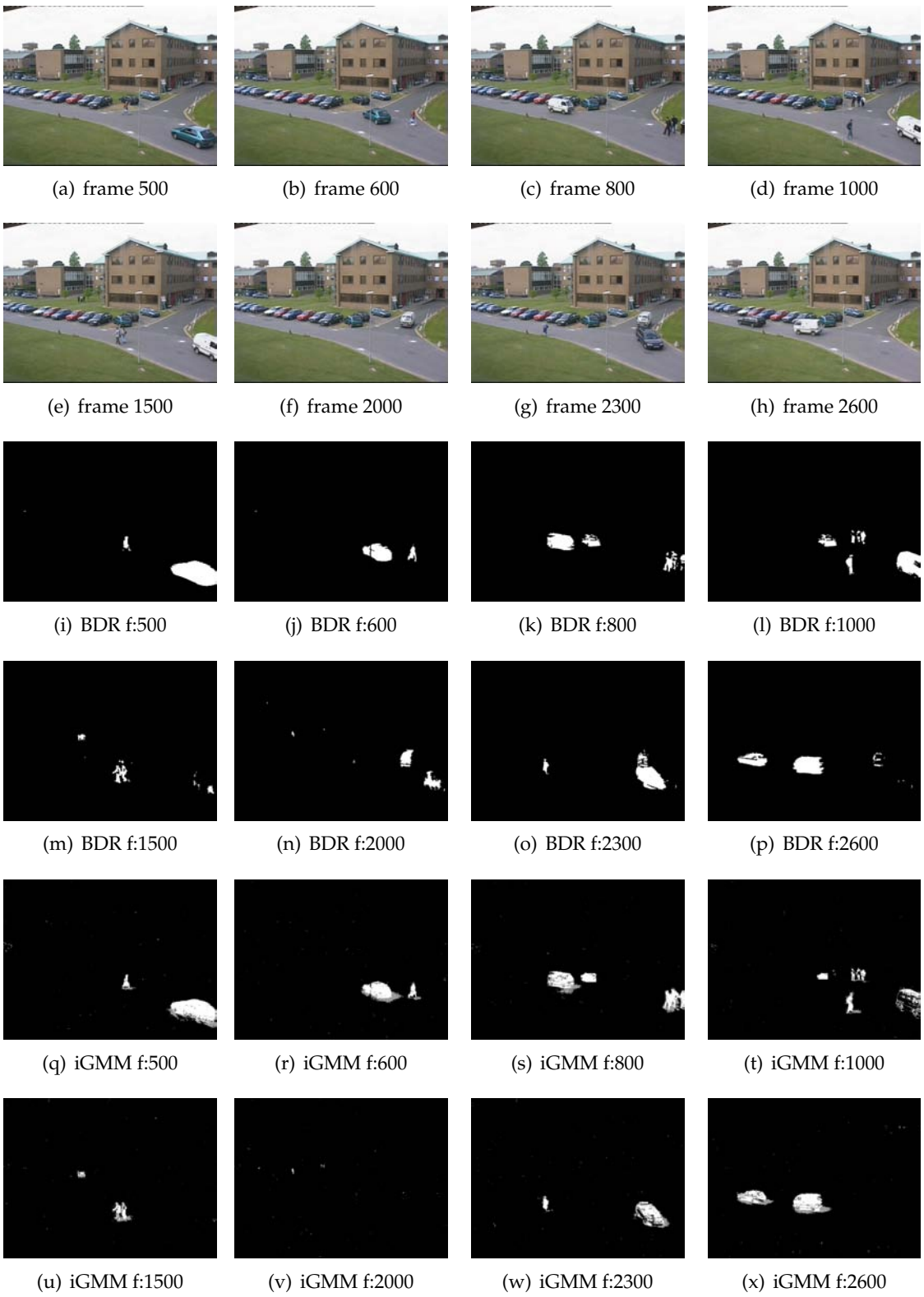


(a) frame 70 (b) frame 380 (c) frame 540

(d) BDR f:70 (e) BDR f:380 (f) BDR f:540

(g) iGMM f:70 (h) iGMM f:380 (i) iGMM f:540

Figure 2.20: Example of detection in a complex indoor scene, dataset Pets 2002



(a) frame 500 (b) frame 600 (c) frame 800 (d) frame 1000

(e) frame 1500 (f) frame 2000 (g) frame 2300 (h) frame 2600

(i) BDR f:500 (j) BDR f:600 (k) BDR f:800 (l) BDR f:1000

(m) BDR f:1500 (n) BDR f:2000 (o) BDR f:2300 (p) BDR f:2600

(q) iGMM f:500 (r) iGMM f:600 (s) iGMM f:800 (t) iGMM f:1000

(u) iGMM f:1500 (v) iGMM f:2000 (w) iGMM f:2300 (x) iGMM f:2600

Figure 2.21: Example of detection in an outdoor video from PETS 2001

Chapter 3

Object tracking

Introduction

This chapter presents object tracking. Object tracking aims at associating an identity to detected regions in consecutive frames. These regions correspond to the same identified object in different frames.

Objects are defined as anything of interest for further analysis. These objects can be: boats on the sea, fish in an aquarium, vehicles on a road, flying planes, walking people, etc, depending on the final application.

We want to build a tracking system that can be adapted in various contexts. However, we have several constraints. First, our application requires a real-time analysis. Therefore the tracking process must be fast enough to meet this constraint. Secondly, we note that our motion detection process, presented in chapter 2, requires a fixed camera. Even though our motion detection technique is robust to slight camera motion, an extra motion compensation process would be required to handle moving cameras. Finally the object contour must be precisely detected for further analysis of the objects' shape.

The motion detection technique presented in chapter 2 satisfies these constraints:

- The method is fast: the processing of one frame takes 0.05 to 0.08 second.
- The detection requires a fixed camera: slight motions are handled by the method.
- The method is precise: detection is achieved at a pixel level and the method is robust to noises.

Object tracking is a challenging task. We list several issues that have to be handled by the tracking system:

- Fast object relatively to the frame rate are hard to track since their position varies a lot from one frame to the next. Then, in presence of multiple objects, the objects position is not a relevant measurement and can lead to miss-match.
- Inhomogeneous trajectories can generate difficulties in the matching process because several tracking algorithms assume the trajectories to be smooth.
- Deformable, or non-rigid, object are harder to track than rigid objects. These objects appearance varies a lot in the same scene.
- Multiple objects increase the complexity of the matching process.
- Occlusions are a major issue in object tracking. Occlusions can occur between several objects or between an object and the scene.

Figure 3.1: Motion detection results for LAB1 and MALL1 datasets

In this chapter, we first review previous works. Then, various object tracking methods are presented. These methods are all based on the motion detection technique presented in the previous chapter. Two examples of motion detection results are shown in figure 3.1.

Motion detection is used at every frame or used to detect the entrance of objects that are further tracked without information about the foreground.

We first present several existing methods based on: Connected component, Mean-shift, Particle filtering, and Combined connected component and particle filtering.

Then we present our method. Based on multiple hypothesis of object appearance, we build a multi-level tracking system. The core of this system uses regions local and global information to match these regions over the frame sequence. Regions are first matched and identified from one frame to the next. Then, objects are created and identified. These objects correspond to higher-level instance of regions. Finally merges and splits are detected for better handling occlusions. A functional diagram of the system is presented in figure 3.2.

3.1 State of the art

Tracking aims at matching detected regions over the frame sequence and identifying these regions.

Detection and tracking often intersect because these two processes use the same techniques. As in [53], we divide tracking methods into four categories. Tracking can be based on regions, contours, features, or a model. We note that algorithms from different categories can be integrated together.

3.1.1 Region-based tracking

Region based tracking identifies connected regions corresponding to each moving object in the scene. Motion is usually detected using background subtracting methods.

[159] uses small blobs features to track a single person in an indoor environment. In this method, human body is composed of several blobs corresponding to various body parts such as head, torso, or limbs. Each pixel is assigned to a specific body part's blob. Finally, moving humans are tracked by tracking each blob.

[93] proposes a tracking method based on three levels: regions, people, and groups. Each region is represented by its bounding box and regions can split and merge. A human is composed of one or more regions and a human group consists of several people grouped together. Using the region tracking method and an individual color appearance model, tracking multiple people is achieved, even under occlusions.

[91] and [90] are two examples of region-based tracking applied to vehicles.

These methods offer good results on scenes with various objects. However, occlusions can not be properly handled. Thus, these methods can not properly handle clutter background or objects interacting together.

3.1.2 Contour-based tracking

Active contour based tracking represents objects by their outline, or bounding contours. These contours are updated at every frames [9] [98] [39] [160]. These methods extract objects shapes and obtain more effective description of objects than region-based method. [105] detects and tracks multiple objects with a geodesic active contour objective function. [109] uses an active contour model based on Kalman filter to track non-rigid objects. [56] explores stochastic differential equations to describe complex motion models with deformable template to track people. [73] and [91] have applied active contour to track vehicles

Active contour-based tracking represent objects more simply than region-based methods. However, the tracking precision is limited to the contour level and automatic initialization is challenging.

3.1.3 Feature-based tracking

Feature-based tracking achieves tracking by extracting various elements that are used to describe the corresponding objects. These features are matched between frames. We distinguish two categories of features: global features and local features.

- Global features are centroids, perimeters, surface areas, color moments, [106], [127], etc. [114] is an example of tracking using the centroid of the region's bounding box. Occlusions can be handled as long as the velocity of the centroids is distinguished efficiently.

  Several methods represent regions as points moving in the scene. To match these points in the frame sequence, [148] and [131] use deterministic methods, while probabilistic methods can be used such as Kalman filter [11] [125] and particle filter [57] [89]. We note that many researchers focused on particle filtering lately [169].

- Local features are line segments, curve segments, corner, interest points, [26], [91], etc.

We note that the presented features can be combined. [61] proposes a template based on shape, texture, color, and edge features of the regions. The matching process is performed by minimizing a feature energy function.

Feature-based tracking can handle multiple object tracking and local features can be matched when partial occlusions occur.

3.1.4 Model-based tracking

Model-based tracking aims at matching projected object models to image data. These models require prior knowledge and are usually produced off-line with manual measurements. We

present two separated types of models: models for non-rigid objects tracking (human body) and models for rigid objects tracking (vehicle).

3.1.4.1 Model-based human body tracking

These methods are based on three main steps: predict, match, and update. First, the pose of the model is predicted for the next frame. Secondly, the model is synthesized and projected into the image plane. Then, the similarity between the image data and the projected estimate is calculated using a pose evaluation function. This function can be recursive or using sampling techniques to end up with the correct pose and update the model.

There are three main difficulties:

The construction of human body model The human body model is the base of the tracking method. The more complex it is, the more accurate the system will be and also the more computationally expensive. We categorize four types of body models:

- Stick figure represents motion of head, torso, and limbs using sticks and joints. [67] uses HMM to encode the model.
- 2-D contours models the human body using 2-D ribbons [64] or blobs [101].
- Volumetric model represents human body with elliptical cylinders [124], cones [150], [31], [30], spheres, super-quadrics [134], etc.
- Hierarchical model can represent the body very accurately [112]: skeleton, ellipsoid meatballs simulating tissues and fats, polygonal surface representing skin, and shaded rendering.

We note that volumetric and hierarchical models are computationally expensive.

The representation of prior knowledge of motion models and motion constraints Human limbs and joints movement are strongly constrained. Thus, motion models of limbs and joints are used to predict motion parameters [168] [23] or to recognize human behaviors [18]. These motion models can be learned using HMM [18], minimum description length paradigm [168], multivariate principal component analysis [132], hierarchical principal component analysis [103], etc.

The prediction and search strategies There are four main types of search strategies: Dynamics, Taylor models, Kalman filtering, and stochastic sampling.

Dynamical strategies use physical forces applied to each rigid part of the model. These forces guide the minimization of the difference between the model and the image data [31].

Taylor models incrementally improve the estimated pose, using differentials of motion parameters and image observations [84].

Kalman filtering uses motion parameters modeled by Gaussian [150] [65].

Stochastic strategies, such as Markov chain Monte Carlo [10] or Condensation algorithm [58] [57], represent simultaneous alternatives hypotheses.

3.1.4.2 Model-based vehicles tracking

These methods are mainly based on 3-D wire-frame vehicle models [40]. [143] and [141] constraint vehicles to move on the ground plane. Thus the pose estimation is simplified and computational costs are reduced. [142] estimates the vehicle pose with a Hough transformation algorithm. [140] analyzes the 1-D correlation of the image gradients. [108] and [107] propose a statistical Newton method to estimate the vehicle pose.

[162] uses edge points as features to match the 3-D model projection. [83] tracks vehicles using improved extended Kalman filter and handles complex maneuver.

[72] also uses edges as features to match the 3-D projection. The maximum a posteriori estimate of the vehicle position is obtained using Levenberg-Marquardt optimization. [75] also bases his systems on image gradients. Virtual gradients are produced by spreading Gaussian distribution around line segments. Assuming that the real gradient in an image correspond to the sum of a virtual gradient and a Gaussian white noise, the pose parameters are estimated using extended Kalman filter. [44] applies the same methods based on optical flow.

Model based tracking is more robust than the other methods essentially due to the prior knowledge integrated in the model. This tracking can obtain results under occlusions, handles changes of orientation. However, the main disadvantages of this technique are the construction of the model and the computational costs.

3.2 Evaluated approaches

This section presents four methods: Connected Component (CC), Mean-Shift (MS), Particle Filtering with Mean-Shift weights (MSPF), and the combined Connected Component and Particle Filtering to solve collisions (CCMSPF). These methods are some of the most widely used and effective region-based and feature-based tracking methods.

3.2.1 Connected components

This method is directly based on the motion detection output. For each frame, each detected regions is added to a region list. For each region in the list, a Kalman filter is used to estimate the location and size of the region in the next frame. Once the estimate is calculated for each region of the previous frame, we look for the region, in the current frame, that is the closest to the estimate. Closest regions are matched to the corresponding regions and are used to update the region location and size. Once a region is matched for a certain amount of frames $T = 6$, the region is validated and receives its proper identification.

This method only uses the location and size of each region and is therefore very fast. We note that the longer an object is matched, the better the estimate is.

3.2.2 Mean-shift

The mean-shift algorithm is based on color information and is composed of 3 main phases [27]:

- Once a moving region is detected, an initial color histogram is calculated over its bounding box.
- This histogram is used to calculate the back projection over the image.
- In the result image, large connected areas with high intensities correspond to locations that are similar to the original histogram. Therefore Mean-shift algorithm is used to find the location and size of the matched object.

The distance calculation between two histograms is based on the Bhattacharyya coefficient. This coefficient general form is:

$$\rho(s) \equiv \rho[p(s), q] = \int \sqrt{p_z(s)\, q_z}\, dz \tag{3.1}$$

Where z represents the color of the model, with a density function q_z. The candidate at location s has the feature distributed according to $p_z(s)$

We want to find the discrete location s with the associate density $p_z(s)$ that is the closest to the model density q_z.

The derivation of the Bhattacharyya coefficient from sample data involves the estimation of the densities p and q. For these densities, we use the histogram formulation. q is estimated from the m-bin histogram of the target model. p is estimated from the m-bin histogram of the candidate at location y. Then the sample estimate of the Bhattacharyya coefficient is:

$$\hat{\rho}(s) \equiv \rho[\hat{p}(s), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(s)\ \hat{q}_u} \tag{3.2}$$

The distance between two distributions is defined as follow:

$$d(s) = \sqrt{1 - \rho[\hat{p}(s), \hat{q}]} \tag{3.3}$$

We note that this distance is invariant to the scale of the target.

3.2.3 Particle filtering

Particle filtering use Monte Carlo method for on-line filtering [76]. Monte Carlo approximation of a distribution is a set of random samples [32]

$$X = \{X(m), W(m)\}_{m=1,\ldots,M} \tag{3.4}$$

Where $X(m)$ are particles paired with weights $W(m)$. m indexes the set of M particles. The density is approximated from X as:

$$p(X) \approx \sum_{m=1}^{M} W(m)\delta(X - X(m)) \tag{3.5}$$

Where Dirac delta $\delta()$ is weighted with particles.

The algorithm for Sequential Importance Sampling (SIS) [7] is the core of the sequential Monte Carlo methods. However, a bad choice of "importance density function" can rapidly decrease the particle weights, leaving a few particles with low weights. This degeneracy problem can be avoided by resampling particles, with resampling probability being proportional to particles weight [42]. This process consists in eliminating particles with low weights and multiplying the ones with higher weights.

The method using Sampling Importance Resampling (SIR) is achieved by resampling and a choice of the "prior importance function" [32].

$$\Pi(X_t|X_{t-1}(m), y_{1:t}) = p(X_t|X_{t-1}(m)) \tag{3.6}$$

This prior importance function reduce the effects of the measurements z from the calculations. The SIR filter with prior importance function has a reduced evaluation of weights.

$$W_t(m) \propto W_{t-1}(m)\ p(y_t|X_t(m)) \tag{3.7}$$

This approach is also called bootstrap filter and is described in the following algorithm [33]:

Initialization

- Sample N particles from initial distribution $X_0(m) \sim p(X_0)$ with $m = 1, \ldots, N$.
- Set $t = 0$.

Importance sampling step

- Sample particles from importance function $X_t(m) \sim p(X_t|X_{t-1}(m))$ with $m = 1, \ldots, N$.
- Evaluate the importance weights with equation 3.7 $W_t(m) = p(y_t|X_t(m))$ with $m = 1, \ldots, N$.
- Normalize the importance weights.

Selection with resampling

- Resample with replacement N particles each particle $X_t(m)$ is replaced with probability equal to its importance weight $W_t(m)$.
- Set $t \leftarrow t + 1$ and go back to step 2.

Finally, the particles are propagated according to the importance distribution function while weights are evaluated to reflect fitness of particles to observations y_t. We note that the initialization of the tracking is done on moving regions detected through motion detection.

3.2.4 Combined connected components and particle filtering

This method is based on connected components, see section 3.2.1. Motion detection is achieved on every frame and connected regions are matched from one frame to the next using location and size measurement and a Kalman filter. Kalman filter estimate the next position of a region and can therefore detect collision between regions. When such a collision is detected, the regions are tracked using the particle filtering technique presented in section 3.2.3. We note that the positions and sizes of the objects are still updated and analyzed with the Kalman filter and when the collision is over, objects are tracked with the connected components technique.

3.3 Proposed tracking process

3.3.1 Introduction

Based on motion detection, we want to match the detected connected regions over the frame sequence. A region, or blob, is defined here as a connected surface considered as a foreground area, see chapter 2. An overview of our system is presented in figure 3.2.

In this section, we first present the multiple hypothesis of the object appearance. Then, we merge regions in order to obtain regions that better match actual people. We use various types of merge to handle uncertainty. The core of our tracking system is the frame to frame matching process. This process builds a descriptor for each region of a frame and matches these descriptors to the previous frame regions' descriptors. Regions are further identified by propagating Ids across frames to matched regions. However, tracks can be lost and miss-matches can occur. Therefore we build an object list that correspond to higher-level instances of regions that handles lost tracks and miss-matches by using extra temporal information. We finally detect merges and splits of these identified objects to better detect occlusions between these objects.

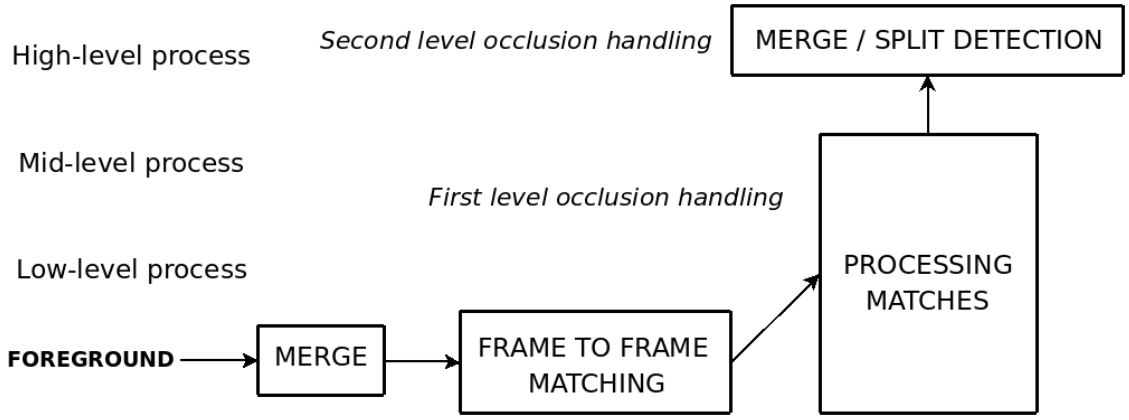


Figure 3.2: Tracking system diagram

3.3.2 Multiple hypothesis

Our system is based on the motion detection technique presented in chapter 2. Even though motion detection offers good results, we note that the algorithm can miss parts of an object, or person, see figure 3.1.

In fact, a detected object can be covered by several disconnected regions. We also note that two objects occluding each other are merged as a single connected region.

Thus we define three hypothesis concerning detected regions. A detected region can be:

- A part of an object.
- An entire object.
- A group of objects.

We then have to cope with several cases concerning the appearance of the detected regions.

3.3.3 Regions merge

To have regions that better match actual objects, we achieve a merging process, see figure 3.3. There are two separated parts in the merging process, first reliable merges are made, then potential merges are considered. It is interesting to note that regions are represented by their bounding box.

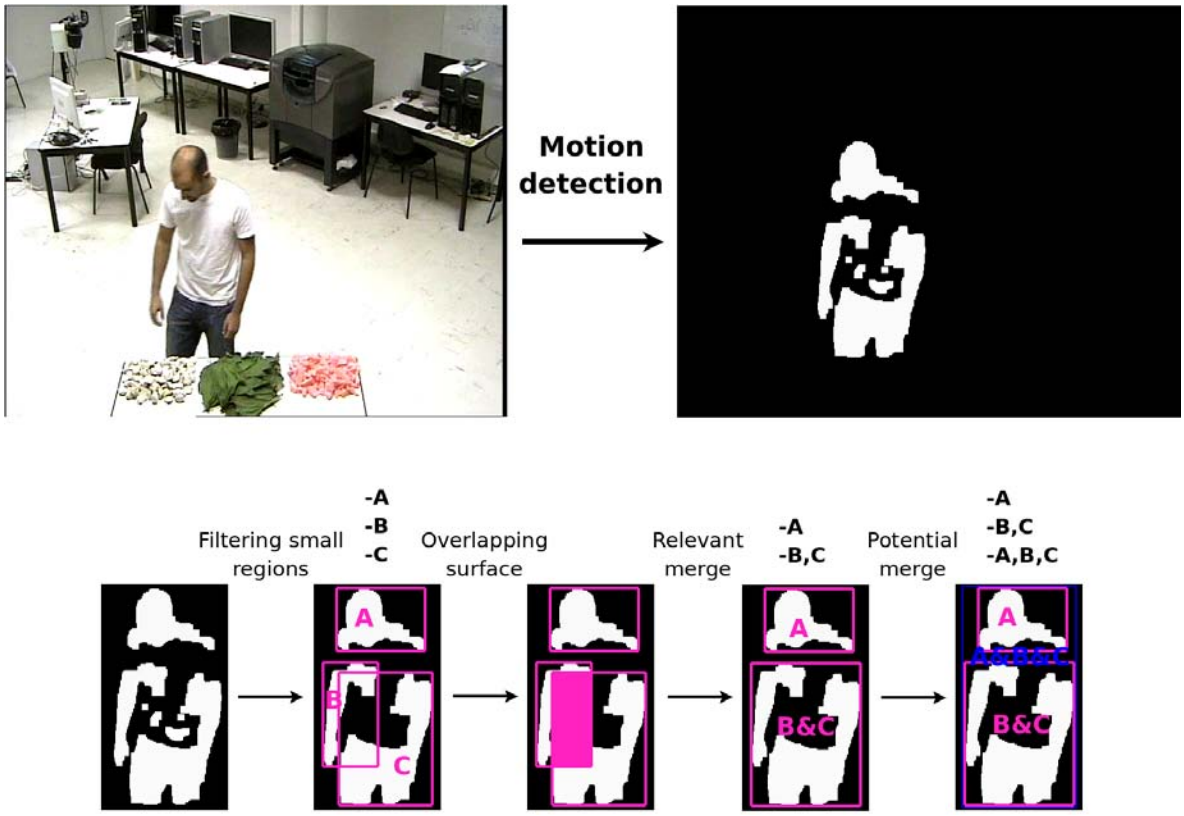


Figure 3.3: Diagram representing the merging process. Each step and the region list are shown. The blue region represent the potiontial merge of A and B,C

3.3.3.1 Relevant merge

The first step consists in detecting regions located inside other regions bounding box. Then overlapping bounding boxes are detected and overlapping surface areas are calculated. If a surface area is larger than a given value s, regions are merged, see figure 3.3.

After this process, regions better match actual persons.

3.3.3.2 Potential merge

This next process aims at achieving less reliable merge. Regions may correspond to the same person.

Slightly overlapping regions First, regions with overlapping bounding boxes are detected. These regions are then potentially merged.

Close by regions Then, distances between regions are calculated. If the distance is smaller than a given value d_1, regions are once again potentially merged.

Regions in the same vertical axis In a few cases, we notice that the motion detection fails to correctly detect parts of a person. As we see on figure 3.1, textural similarity between the person's clothes and the background is important.

In order to handle this type of cases, we assume that a person is significantly taller than wide. We note that this ratio depends on each person and on the camera view point. As a consequence, two regions located on the same vertical axis might be considered as part of the same person and then be "potentially" merged, see figure 3.3.

Specifically, potential merge occurs when the gravity center of the highest region is horizontally located within the bounding box of the lower region and if the distance is not larger than a given value d_2.

3.3.3.3 Merge versus potential merge

The two types of merge have different effects on the matching process that follows.

When two regions are reliably merged, the two original regions become one merged region. In other words, the two original regions are deleted and a new one is generated, see figure 3.3.

However, when two regions are potentially merged, the two original regions are kept in the list and a new merged region is generated, see figure 3.3. This new merged region has

type P for easier notation. Since we are not sure about the reliability of these merges, we use the following matching process to decide whether merging is relevant or not.

3.3.4 Frame to frame matching

In order to match regions, we first build a descriptor for each of them. Such a descriptor must be filled with measurements that are discriminant and relatively stable across frames. Furthermore, these measurements must allow us to match regions of different sizes and shapes, to be consistent with our hypothesis. We are therefore interested in the location, size, and color appearance of the regions.

The descriptor is composed of the region gravity centre position (x_g, y_g), size (h, w), position of the bounding box centre (x_b, y_b), surface area s, and first and second order color moments $(R_1, G_1, B_1, R_2, G_2, B_2)$. The descriptor is then a vector of measurement $x = x_1, x_2, ..., x_n$ with $n = 13$.

The regions' matching is achieved by using a descriptor matching algorithm, similar to [92], which works as follow.

We define two sets of regions S_1 and S_2. S_1 corresponds to the previous frame and S_2 to the current one. Each set is composed of one or several regions. Two regions with descriptors $x \in S_1$ and $y \in S_2$ are matched if and only if x is the most similar descriptor to y and vice-versa, i.e.

$$\begin{aligned} &\forall y' \in S_2 | y : sim(x, y) \geq sim(x, y') \; and \\ &\forall x' \in S_1 | x : sim(y, x) \geq sim(y, x') \end{aligned} \tag{3.8}$$

Where sim is the asymmetric similarity measure defined below. To calculate sim, each component of the descriptor is treated independently. The similarity between the ith component of x and y is equal to 1 if y ith component is the closest measurement to x ith component. Otherwise, the similarity is equal to 0. Closest measurements have smaller Euclidean distance.

$$\begin{aligned} sim^i(x, y) = 1 \; if \; \forall y' \in S_2, sim^i(x, y) \geq sim^i(x, y') \\ 0 \; otherwise \end{aligned} \tag{3.9}$$

The overall similarity measure is defined as follows

$$sim(x,y) = \sum_{i=0}^{n} \omega_i \, sim^i(x,y) \tag{3.10}$$

Where n is the dimension of the descriptor, defined above, and ω_i is the weight of the ith measurement. We choose to give the same weight $\omega_0 = 1$ to each measurement of the descriptor. The calculation of $sim(y,x)$ is analogous with the roles of S_1 and S_2 interchanged.

An interesting property of this calculation is that the influence of any single measurement is limited to 1. In other words, no matter how large the distance is between the ith measurement of two descriptors, the overall similarity measure will only lose 1.

Another major property of this algorithm is that measurements of different orders of magnitude can feet together in the descriptor and are easily handled. For example, the value of the surface area of a person is significantly larger than the value of its width.

Finally, since we define matches as the most similar regions from the first set to the second one and vice-versa, the more regions are in both sets the harder it is to find a match. On the contrary, the algorithm can match highly different regions when there are only a couple regions in the sets.

3.3.5 Processing matches

Once the matching process is achieved, we have couples of matched regions. We first filter these matches, then identify regions. Finally objects are identified, see figure 3.4.

We have to be aware that the system matches regions on each frame of videos that have 15 to 25 frames per seconds. With such a large amount of frames, we are sure to have missmatch occurring. Thus, we need to achieve matching on several levels to handle the uncertainty. We use objects to represent tracked regions. These identified objects use extra temporal information and are not limited to two consecutive frames.

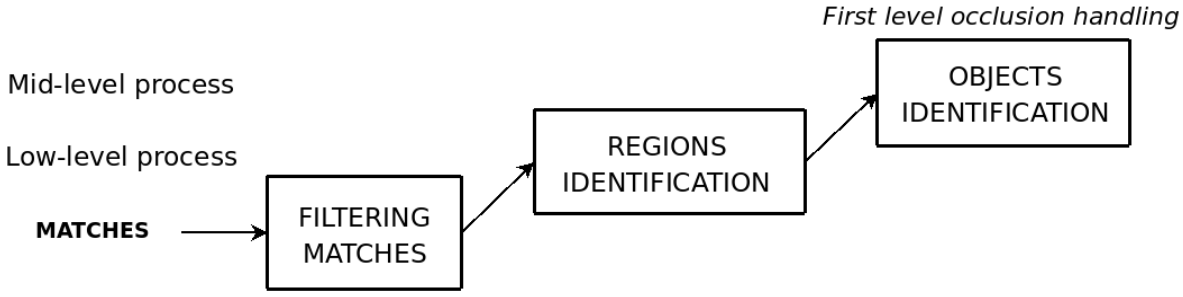


Figure 3.4: Diagram representing the processing of matches

3.3.5.1 Matches filtering

First of all, tests are achieved on the list of matched regions. These tests aim at removing sub-regions, i.e. regions that are a part of other matched regions. This case can occur when a type P region is matched as well as regions that compose it.

Then, we test matched regions for merging. As we show in section 3.3.3, we achieve relevant merge. In fact, in a few cases type P regions can be merged to other regions.

3.3.5.2 Regions identification

Regions are identified only if they are matched in the current frame. These matched regions receive the identification of the region they are matched with. If this region is not identified, we create an identity for the matched region. It is interesting to note that the identification is propagated to sub-regions.

3.3.5.3 Objects identification

Once regions are identified, we compare each matched region with the list of tracked objects. There are various cases:

- A matched region corresponds to an object and this object is corresponding to only one region. The region is used to update objects information, such as its location, size, surface area, color, etc.
- No object is corresponding to a matched region; this region can be a new object entering the scene or an old object that was lost, due to an occlusion for example.

In order to retrieve an object after a miss-detection or an occlusion, we reiterate the matching process. However, we modify the descriptor by only keeping the measurements that are invariant to displacement. In other words, we only keep the color and size information and delete the location measurements in the descriptor. In fact, during an occlusion a person keeps moving and rarely comes back to its exact first position at the beginning of the occlusion.

Then, we require two sets of regions for the matching process, see section 3.3.4. The set S_1 is composed of every object and the set S_2 remains the list of region detected in the current frame. We keep all regions of the current frame and all objects to have a selective matching and avoid matching regions that are not similar. If the region is matched to an inactive object, we may have encountered an occlusion. Otherwise, a new object is created and filled with the region's information.

3.3.6 Merge - split detection for occlusions handling

Handling occlusion is a tough challenge. We are under the constraint that a detected region can either be a part of a person, a person or a group of people. Therefore, it is difficult to detect the difference between a person merging and splitting into one to several tracked regions and a group of people occluding each other: crossing, meeting, etc.

It is interesting to note that the process previously presented, see section 3.3.4, already solves most occlusions. In fact, when an object disappears, during an occlusion for example. As soon as this object reappears, our method can not find a match to the detected region. Therefore, the algorithm tries to find a match with an old object, which disappeared.

However, some cases can be complicated and splits and merges detection offer us another clue to identify occlusions.

3.3.6.1 Merge detection

Several regions are merging when these regions are considered as different identified regions in the previous frames and then become one single region at the current frame. We remember that a region is identified when it is matched.

For example, two regions are tracked A, B and a type P region $C = A, B$ representing the potential merge of these two regions is matched at the current frame. Then a merge just occur, see figure 3.5.

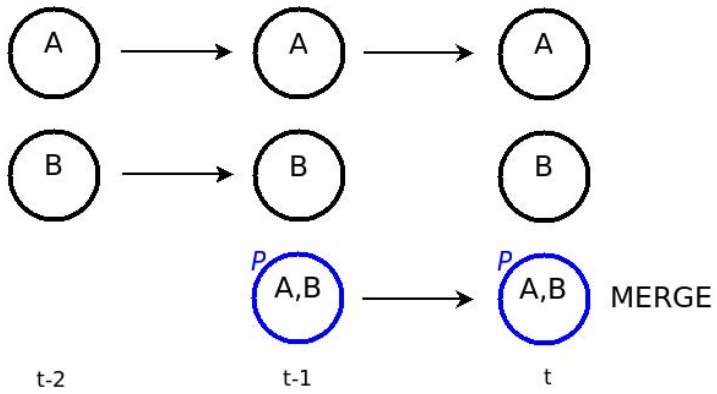


Figure 3.5: Merge occur: A and B merge into C (in blue). Arrows link matched regions

3.3.6.2 Split detection

A region is splitting into several regions when a tracked region is not matched at the current frame and only its sub-regions are matched. These sub-regions have same id.

For example, one region of type P: $C = A, B$ is not tracked anymore and two sub-regions are matched at the current frame A and B, see figure 3.6.

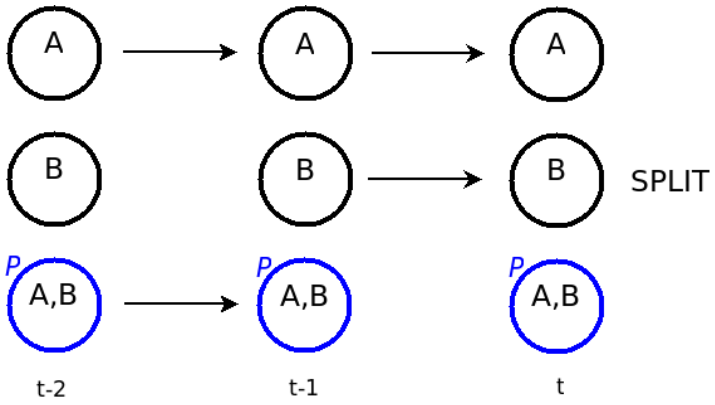


Figure 3.6: Split occur: C (in blue) splits into A and B. Arrows link matched regions

3.3.6.3 Occlusion detection

We use some measurements to define the consistency of these splits and merges. First, we filter out small objects that can not correspond to an entire person. Then, when a merge, or split, occur we calculate the age of the concerned object(s).

In fact, when these events occur on a single person, they do not last for a long period of time. Therefore, the regions splitting and merging should not be "old". Moreover, when two tracked people meet, they are usually tracked for a certain time before the encounter. There is also a certain amount of time between the merge and the split. Based on these measurements we can detect occlusions.

3.4 Evaluation and results

This section presents the evaluation of the presented tracking algorithms in various contexts. We first test our algorithm and show some limitations. Then we compare all algorithms on sequences with occlusions. We track vehicles for traffic monitoring and finally detect meetings in a video-surveillance context.

3.4.1 Tests on our method

Our algorithm is performing very well on LAB and MALL datasets. The system can track several objects simultaneously, is fast and accurate.

This first example shows the difference between the identification of regions and objects. Even when frame to frame matching generates wrong results, higher level analysis allows us to track a person correctly even in the presence of splits and merges, see figure 3.7.

The major difficulties come from occlusions. In many videos, several people interact simultaneously. These people occlude one another. Our system detects these occlusions and aims at correctly re-identify objects once the occlusion is over. This task is not easy, especially under the constraint that detected regions can either be a part of a person, a person or a group of person. The second example, see figure 3.8, shows a simple occlusion resulting in a split at frame 125. This type of cases is well handled by our system.

However, limitations of our method are reached in the next example, see figure 3.9. Around frame 170 and frame 250, one person is leaving when another enter the scene. Since the two persons are merged together, the system track the two persons as a single one and can not detect that one of them has left and was replaced by a new one.

We further note that the person with Id 6 switch to Id 2, in the same sequence. In fact, the person split into several objects and back to one. In such a case, the lower Id value is printed. Lower Id values correspond, in most cases, to older objects that tend to be more reliable. In fact, we know that

- Object 2 has a set of regions indexes: 5 - 13 - 17 - 19 - 20
- Object 6 has a set of regions indexes: 10 - 22
- Split of regions Id: 10 - 17, at frame 187 the object age is 46
- Merge of regions Id: 22 - 20, at frame 204 the object age is 62

3.4.2 Comparative Tests

First, it is interesting to note that there is a major difference between our method and the others:

- Our method tracks an object as soon as a corresponding region is matched in two consecutive frames. In other words, when a moving area is detected in more than 2 frames, we can already track it as an object. Therefore, our method adapts quickly to the scene but is more subject to noise.
- The existing methods require an object to be steady or have an homogeneous trajectory for around 10 to 15 frames, in practice, in order to validate its appearance as an object.

These tracking processes are not able to track objects in some videos. For example, only our method is able to track object in the MALL datasets. There are several reasons: objects are large, they generally do not stop, they do not always have homogeneous and straight trajectories, and they often are not fully in the image.

3.4.2.1 Occlusions

We achieve various tests when occlusions occur. We use our datasets and videos from PETS 2006. We test several videos in each dataset, see figure 3.11, 3.12, 3.10, 3.15, 3.16, 3.13, and 3.14. As we can see, our method handles various types of occlusions.

Figure 3.10 shows an example of complete occlusion. Our algorithm handles the occlusion. However, the other methods do not identify the second moving person before the occlusion. The first person is correctly tracked by all the method. We note that the bounding box of the tracked person is less accurate for the CCMSPF method due lighting changes generating noises.

Figure 3.11 and 3.12 shows three partial occlusions. Our algorithm correctly tracks the two persons on all occlusions. On the same figure, we note that CC offers poor results. Ids change from one person to the other, tracks are lost several times. MS handles the two last occlusions as well as MSPF and CCMSPF. However CCMSPF is, again, less precise. We note that these methods do not track the objects soon enough to cope with the first occlusion.

Then, we test all methods on the PETS 2006 dataset, see figure 3.13, 3.14, 3.15, and 3.16. The videos 7-4 and 7-3 are taken from two different locations in a train station.

Figure 3.13 and 3.14 show the results on video 7-4. Our method offers correct results and manages the occlusion. We note that none of the other methods successfully track the two persons. CC does not manage the occlusion and the one identified person loses its Id during the occlusion. MS, MSPF, and CCMSPF keep track of the person.

Figure 3.15 and 3.16 show another occlusion occurring on video 7-3. All methods can track the two objects before the occlusion. Our method, MS and CCMSPF handle the occlusion. CC and MSPF miss-match the persons.

It is interesting to note that the CC method relies a lot on the Kalmann Filter. When an object changes its trajectory, the method tend to detect the object on the same trajectory even if the real person stops or switches direction, as we can see on figure 3.11.

Finally this evaluation shows that our tracking algorithm can handle various types of occlusions.

3.4.3 Traffic monitoring: vehicle counting

In this section, we evaluate our tracking algorithm on a different task. We want to track vehicles on highways. There is ongoing research in this area of computer vision. The main idea is to detect automatically abnormal behaviors, such as accidents or traffic jams for example. Most systems are based on an object tracking process, and then vehicles trajectories are analyzed to detect abnormal events [100].

3.4.3.1 Task description

We test various algorithms on a simple task: counting vehicles. We use two videos [1], see figure 3.17. These videos are outdoor sequences filming motorways. The cameras are fixed on top of poles and allow us to see couple of hundreds meters of road. In order to count vehicles, we use our motion detection technique presented in chapter 2 and apply five tracking techniques to count all objects. Objects tracked for more than 30 frames are considered reliable.

In order to avoid false detection, we block the motion detection on the top left part of the video 2. In fact, we observe motion due to the video clock, as well as vehicles on the side road.

3.4.3.2 Results

The results, in the table 3.18, present the number of vehicles tracked using each algorithm. We also calculate the ground truth by hand. Then, we calculate the processing time of each algorithm.

Although, recall and precision would be better measurements for the evaluation, the number of tracked object already give us a reliable information regarding the number of tracked objects, since all tracking process are based on the same detection.

It is interesting to note that none of the methods is able to detect more than 93% of the vehicles because these videos are really challenging. Many occlusions occur due to the scene geometry. There is a lot of traffic, cars overtake one another, trucks are occluding cars, etc.

We also note that the second video is more challenging than the first one. We see a part of the road from the side and therefore complete occlusions are occurring more often. In

[1] ADACIS sarl and CETE sud-ouest provided the traffic sequences

particular, a truck can occlude one or two lanes. Moreover, the lighting conditions are not very good. The weather is cloudy and the road is dark. Thus, the motion detection process reaches its limitations when grey car have a color very similar to the road. Such vehicles are detected only when they are very close to the camera. On this video, we can observe some camera motion as well as high lighting changes due to clouds passing. However the camera motion is not large enough to generate too much noise and lighting changes are well handled by the motion detection process.

Our method offers interesting results and we are able to count 93% and 87% of the vehicles. Our method outperforms the other methods because they are not coping very well with the noise in the motion detection. We remind that the other methods must detect and track objects with homogeneous and smooth trajectories in order to validate these objects.

The motion detection is particularly noisy when vehicles are far away, due to their small sizes and the constant occlusions. Also, the objects size increases, or decreases, exponentially since they are going toward, or outward, the camera. Moreover, tracking many objects simultaneously is challenging and increases uncertainty.

Finally, this evaluation shows that our method is efficient and able to successfully track a dozen of objects simultaneously.

3.4.4 Video-surveillance: Meeting detection

This section presents a different type of application related to video-surveillance. We entered the VAST challenge 2009 [1] that is a contest linked with the Visual Analytics Science and Technology conference. Although this conference and challenge are in the video analytics research field, one task of the challenge need a computer vision system. The team working on this task was composed of three persons working in the computer vision field and two persons working in the data visualization field.

Introduction

The goal of the task is to discover meetings at locations captured by a security camera. The first goal is to provide a table of location, start time and duration of the meetings. Finally, we have to identify any events of potential interest in the video. Activities must be described as well as why the event is of interest. The data is composed of two videos of five hours each taken with a Pan-Tilt-Zoom (PTZ) camera. The outdoor scene is shot from a street corner and pans between four locations.

To solve the task, we build a system composed of two main parts: a computer vision system and a visualization system. The computer vision system uses an adaptive background

mixture model for motion detection. Then, objects are tracked based on regions and features. The results of this phase are passed as input to the visualization system. In the visualization, the position of an object in a given frame is mapped to a node, and nodes representing the same object in adjacent frames are connected with edges. Automatic filters eliminate common motions from the data. The visualization system helped us discover several meetings in the data.

In this section we only describe the computer vision system. This system is composed of three modules: video preprocessing, motion detection, and object tracking.

3.4.4.1 Video pre-processing

The camera films around 15 seconds each location then moves to the next one. Thus, we preprocess the video to divide it into its four locations by determining when motion on the overall image begins and ends. SVM classifiers, using libSVM package [22], verify that the camera has changed locations. For each frame, we extract an edge histogram descriptor, defined in the MPEG-7 standard, and provide it as input to the SVM classifier which outputs the camera position. An edge Histogram divides an image into a 4x4 grid and computes the histogram of five different edge orientations in each of the grid cells, leading to a 80-dimensional feature vector. This descriptor is appropriate for the task of determining the camera position, because the main edge orientation is independent of the changes in illumination and weather condition in the videos, as opposed to color or texture descriptors.

It is also interesting to note that the video encoding is not of good quality. We can see the video freezing often for several frames, especially when large motion occurs. Therefore, we detect still frames and filter them out when we are tracking objects. Then we avoid having objects detected at the exact same location for several frames. Figure 3.20 represents the functional diagram of the system.

3.4.4.2 Motion detection

Motion detection is achieved by using the Improve Gaussian Mixture Model described in chapter 2.

In order to cope with the camera location shifting, we generate four background models, one for each view. Having these four backgrounds allows us to improve the detection of the moving objects especially in the first seconds of video shooting a new location.

It is interesting to note that the motion detection process is also used to detect camera changing location. Camera motion is detected when large areas of motion are detected on the overall image, for several frames. When such an event occurs, the current background

model is not updated anymore and we wait for the motion to stop to detect the new view and use the next background model.

3.4.4.3 Object tracking

The object tracking is achieved as described in the section 3.3.

Then, the object tracking process generates a list of moving objects in each view. However, these objects can correspond to car or pedestrian. Our task aims at detecting meeting between people. Thus, we want to filter out all vehicles in order to give filtered information to the visualization system.

To improve the object list, we manually build one mask for each of the four locations, using the video. These masks can help identify the object in the scene. There are four different types of areas that can be encoded on a pixel of the mask: road where cars mainly appear, sidewalk where pedestrians mainly appear, crosswalk where both pedestrians and cars can appear, and building where nothing of interest should appear.

The computer vision system generates the input used by the visualizations. For every detected object, the system records the frames in which the object was observed and the object's descriptor used for the tracking process. Also, the module adds the camera location as well as the location type.

3.4.4.4 Results

Our computer vision system gives good results, considering the video data is challenging. We are tracking relatively small pedestrians, and the data is very large and moderately noisy. Additionally, we are constrained by a limitation of our technique where we can not distinguish two objects when they occlude each other. However, it is possible to detect these occlusions by finding major regions that fuse or divide.

The visualization system use objects trajectories and information to filter out irrelevant objects such as cars, noises detected on buildings, small objects, and objects with standard trajectories. At the end, an operator mainly observes objects stopped or wandering around the same location as well as objects having uncommon trajectories. The operator can see meetings between two objects stopping close to one another for some time and leaving.

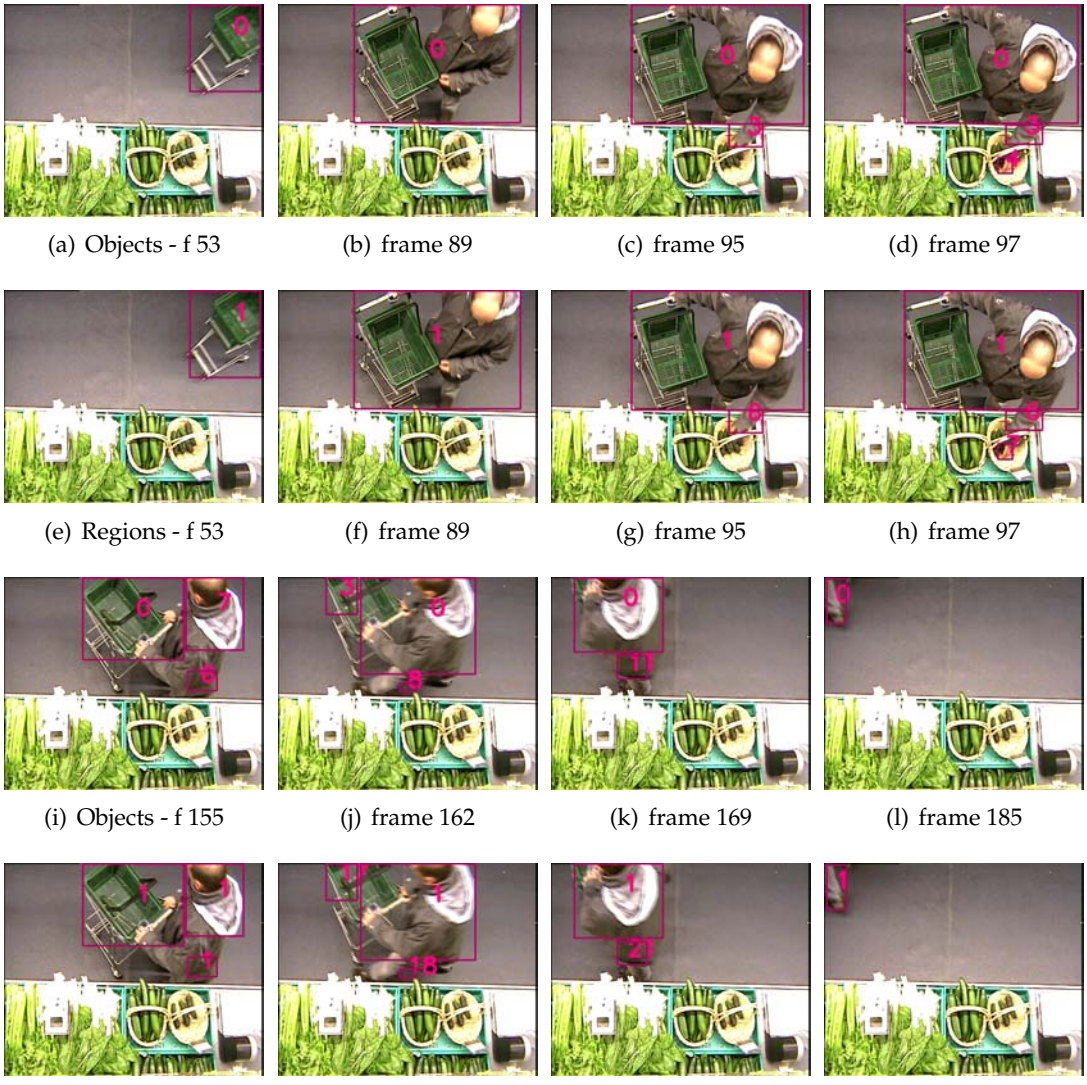
3.5 Conclusions

This chapter presents and compares several object tracking approaches. We propose a system based on multiple hypothesis that achieves tracking using multiple level analysis. Our

system can track several objects simultaneously, is able to track various types of objects (human, vehicles), is able to track objects in outdoor and complex scenes, and is robust to several types of occlusions.

To conclude on our system, having a method based on motion detection is both a major advantage and disadvantage. The advantage is that motion detection offers very accurate detection of the objects contour. The disadvantage comes from the fact that motion detection only gives us binary information: motion is detected of not. Therefore, when two objects are occluding each other, they are detected as a single one.

Finally, tracking outputs are information and identification of each moving object at every frame of the sequence. Based on this information, we can analyze the behavior of each moving object in the scene.



(a) Objects - f 53 (b) frame 89 (c) frame 95 (d) frame 97

(e) Regions - f 53 (f) frame 89 (g) frame 95 (h) frame 97

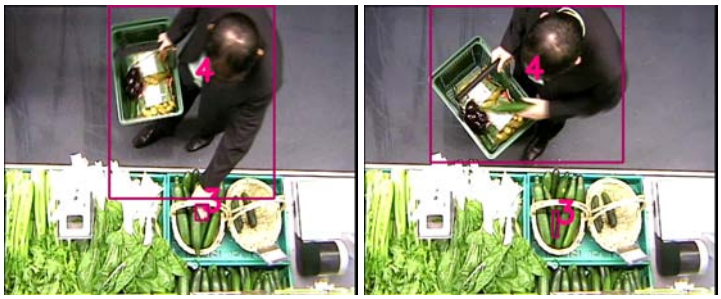
(i) Objects - f 155 (j) frame 162 (k) frame 169 (l) frame 185

(m) Regions - f 155 (n) frame 162 (o) frame 169 (p) frame 185

Figure 3.7: Tracking results for a video taken at the MALL. The first and third rows represent objects Ids, while the other rows correspond to regions

(a) frame 39 (b) frame 62 (c) frame 104

(d) frame 124 (e) frame 125 (f) frame 126

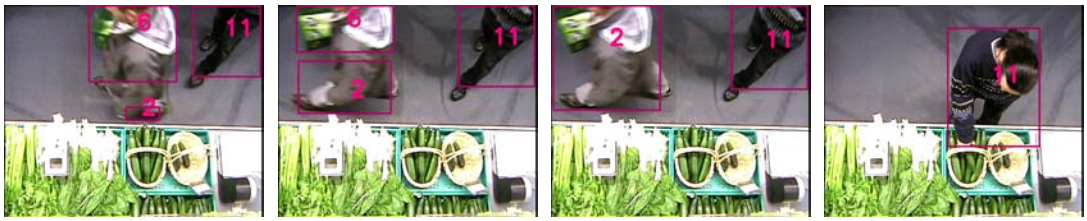


(g) frame 147 (h) frame 160

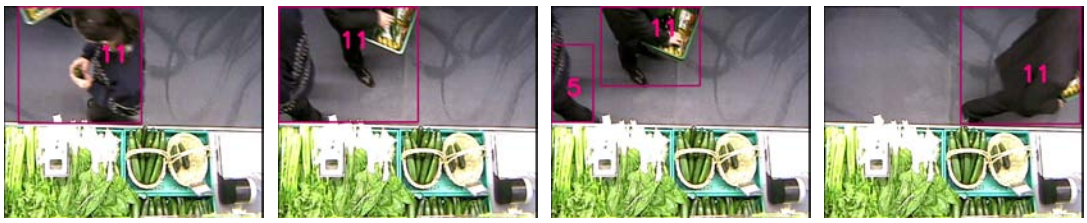
Figure 3.8: Tracking results for MALL 2 video 1, using our methods

(a) frame 105 (b) frame 124 (c) frame 140 (d) frame 141

(e) frame 172 (f) frame 180 (g) frame 195 (h) frame 198



(i) frame 199 (j) frame 204 (k) frame 205 (l) frame 221



(m) frame 236 (n) frame 248 (o) frame 251 (p) frame 278

Figure 3.9: Tracking results for MALL 2 video 3, using our methods

(a) Ours - frame 53 (b) Ours - frame 63 (c) Ours - frame 64 (d) Ours - frame 75 (e) Ours - frame 76

(f) CC - frame 55 (g) CC - frame 59 (h) CC - frame 60 (i) CC - frame 65 (j) CC - frame 75

(k) MS - frame 50 (l) MS - frame 60 (m) MS - frame 70 (n) MS - frame 75

(o) MSPF - frame 55 (p) MSPF - frame 60 (q) MSPF - frame 65 (r) MSPF - frame 70 (s) MSPF - frame 75

(t) CCMSPF - f 55 (u) CCMSPF - f 60 (v) CCMSPF - f 65 (w) CCMSPF - f 70 (x) CCMSPF - f 75

Figure 3.10: Tracking results for LAB 3 video 5, using various methods

(a) Ours - frame 115 (b) Ours - frame 116 (c) Ours - frame 123 (d) Ours - frame 164 (e) Ours - frame 165

(f) Ours - frame 196 (g) Ours - frame 204 (h) Ours - frame 205 (i) Ours - frame 220 (j) Ours - frame 221

(k) CC - frame 100 (l) CC - frame 117 (m) CC - frame 131 (n) CC - frame 132 (o) CC - frame 146

(p) CC - frame 169 (q) CC - frame 198 (r) CC - frame 217 (s) CC - frame 235 (t) CC - frame 244

(u) MS - frame 100 (v) MS - frame 120 (w) MS - frame 128 (x) MS - frame 129

(y) MS - frame 150 (z) MS - frame 184 () MS - frame 215 () MS - frame 240

Figure 3.11: Tracking results for LAB 3 video 4, using Our, CC, and MS methods

(a) MSPF - frame 121 (b) MSPF - frame 128 (c) MSPF - frame 129 (d) MSPF - frame 150 (e) MSPF - frame 167

(f) MSPF - frame 175 (g) MSPF - frame 180 (h) MSPF - frame 200 (i) MSPF - frame 215 (j) MSPF - frame 242

(k) CCMSPF - f 100 (l) CCMSPF - f 116 (m) CCMSPF - f 132 (n) CCMSPF - f 133 (o) CCMSPF - f 150

(p) CCMSPF - f 166 (q) CCMSPF - f 200 (r) CCMSPF - f 215 (s) CCMSPF - f 230 (t) CCMSPF - f 242

Figure 3.12: Tracking results for LAB 3 video 4, using MSPF and CCMSPF methods

(a) Ours - frame 227 (b) Ours - frame 253 (c) Ours - frame 254 (d) Ours - frame 277

(e) Ours - frame 287 (f) Ours - frame 288 (g) Ours - frame 289 (h) Ours - frame 321

(i) CC - frame 140 (j) CC - frame 227 (k) CC - frame 261 (l) CC - frame 262

(m) CC - frame 276 (n) CC - frame 296 (o) CC - frame 297 (p) CC - frame 328

(q) MS - frame 140 (r) MS - frame 227 (s) MS - frame 263

(t) MS - frame 274 (u) MS - frame 297 (v) MS - frame 330

Figure 3.13: Tracking results for PETS 2006 video7-4, using Our, CC, and MS methods

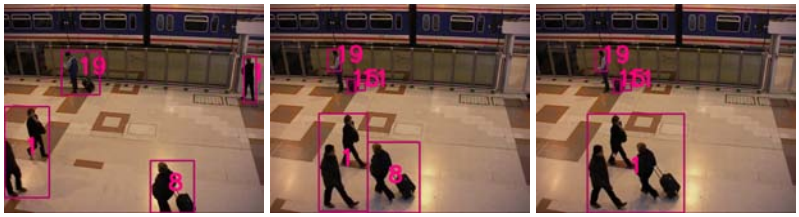
(a) MSPF - frame 162 (b) MSPF - frame 225 (c) MSPF - frame 263

(d) MSPF - frame 273 (e) MSPF - frame 297 (f) MSPF - frame 330

(g) CCMSPF - f 230 (h) CCMSPF - f 261 (i) CCMSPF - f 262 (j) CCMSPF - f 274

(k) CCMSPF - f 285 (l) CCMSPF - f 286 (m) CCMSPF - f 298 (n) CCMSPF - f 331

Figure 3.14: Tracking results for PETS 2006 video7-4, using MSPF, and CCMSPF methods



(a) Ours - frame 120 (b) Ours - frame 153 (c) Ours - frame 154

(d) Ours - frame 179 (e) Ours - frame 180 (f) Ours - frame 223

(g) CC - frame 137 (h) CC - frame 145 (i) CC - frame 147 (j) CC - frame 154

(k) CC - frame 164 (l) CC - frame 178 (m) CC - frame 184 (n) CC - frame 195

(o) MS - frame 115 (p) MS - frame 146 (q) MS - frame 147 (r) MS - frame 155

(s) MS - frame 164 (t) MS - frame 173 (u) MS - frame 184 (v) MS - frame 233

Figure 3.15: Tracking results for PETS 2006 video7-3, using Our, CC, and MS methods

(a) MSPF - frame 110 (b) MSPF - frame 146 (c) MSPF - frame 147 (d) MSPF - frame 155

(e) MSPF - frame 161 (f) MSPF - frame 176 (g) MSPF - frame 183 (h) MSPF - frame 197

(i) CCMSPF - f 110 (j) CCMSPF - f 147 (k) CCMSPF - f 157 (l) CCMSPF - f 168

(m) CCMSPF - f 179 (n) CCMSPF - f 218

Figure 3.16: Tracking results for PETS 2006 video7-3, using MSPF and CCMSPF methods

(a) video 1 (b) video 2

Figure 3.17: Images from video 1 and 2, used for traffic monitoring

	Number of vehicules		Execution time (seconds)	
	Video1	Video2	Video1 (3305 frames)	Video2 (2257 frames)
Ground truth	213	133	N.A.	N.A.
Our method	**198**	**116**	128.8	86.5
CC	173	107	**87.2**	**60.4**
MS	162	90	190.8	134.5
CC-MSPF	175	102	743.3	180.7
MSPF	106	65	1773.2	1306.1

Figure 3.18: Table relating the number of tracked objects and the execution time on two different sequences with five different methods

Figure 3.19: View of the four locations

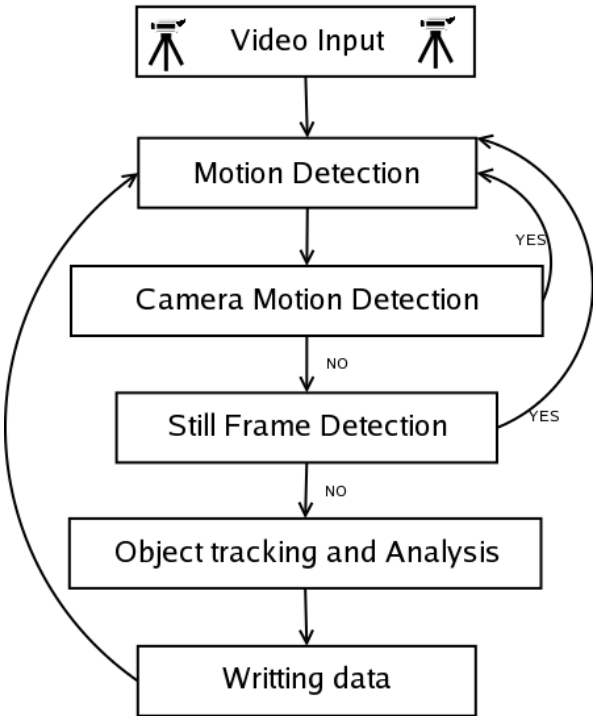


Figure 3.20: Diagram of the video-surveillance system

Chapter 4

Human activity analysis

4.1 Introduction

The object tracking system, previously presented, identifies moving objects in the scene at every frame. Using these data, we can look for measurements that can help determine the behavior of these objects. This section aims at analyzing objects behavior according to a behavior model. The different actions can be detected using simple measurements from the object tracking process or higher level information.

For our application, we are especially interested in detecting the interest of a customer into a product as well as actual interactions between a customer and products of the point of sale.

First, we present the behavior model composed of six states. Then, we focus on the interact state and present the various methods used to detect this state. We further recognize the states and evaluate the various methods presented. An overview of the behavior recognition process and the high-level analysis is presented figure 4.1

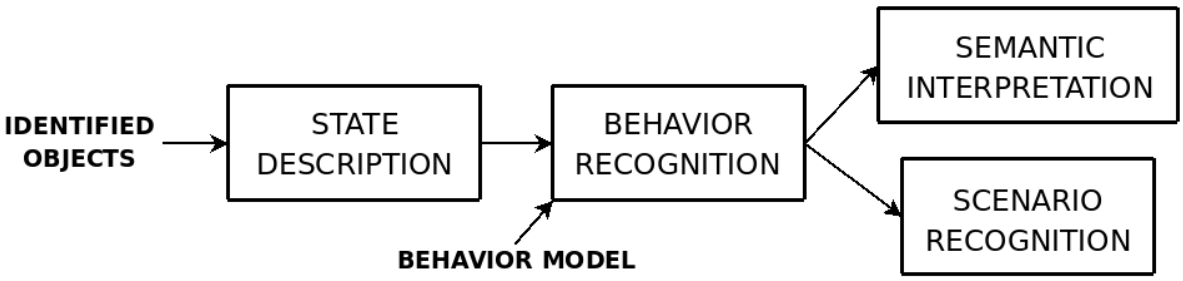


Figure 4.1: Diagram of the behavior recognition process

4.2 State of the art

This section present the previous works in human action recognition (HAR) and human behavior understanding (HBU) that are two areas of computer vision that are close one to another. HAR generally aims at recognizing simple actions that are achieved by one or two people, such as waving, jogging, picking up a phone, hugging, etc, in challenging contexts, like movies or video clips from the Internet. HBU is a more generic term and is often related to video-surveillance. HBU aims at detecting events occurring with several people, such as meetings, fights, or crowd analysis.

4.2.1 Human behavior understanding

Understanding behavior corresponds to the classification of time varying feature data, i.e., matching an unknown test sequence with a group of labeled reference sequences that represent typical behaviors. Then, behavior understanding is about learning reference behaviors from training samples and selecting training and matching methods that cope with small variations of the feature data [13]. We present major techniques for behavior understanding in the following sections.

4.2.1.1 Dynamic time warping (DTW)

DTW is a template based matching technique, widely used in speech recognition. This method is simple and robust, and can be used to match human movement patterns [139]. For example, [14] matches a test sequence with a deterministic sequence of states to recognize human gestures, using DTW. Even if the time scale between a test sequence and a reference sequence is inconsistent, DTW can still establish matching as long as the time ordering constraints are respected.

4.2.1.2 Finite state machine (FSM)

A FSM is made of states and transition functions between them. The transition functions are the most important feature of a FSM. The states represent the reference sequence that matches the test sequence. [157] analyzes the explicit structure of natural gesture with no learning involved. [19] builds a handcraft deterministic FSM to describe vehicle behaviors.

4.2.1.3 HMMs

HMM is a stochastic state machine that allows analysis of spatio-temporal varying data [17]. HMM is composed of two main steps: training and classification. Concerning training, a

number of state transitions must first be specified. Then the output probabilities are optimized so that the generated symbols correspond to the observed image features of the examples. During the matching process, the probability with which a particular HMM generates the test symbol sequence corresponding to the observed image feature is computed. HMM is a widely used method since it generally outperforms DTW. [135] is an example of use of the HMM for the recognition of sign language. [102] compares HMM to coupled hidden Markov models (CHMM) to model people behaviors and shows that CHMMs are more efficient and accurate than HMMs.

4.2.1.4 Time-delay neural network (TDNN)

TDNN is another approach to analyze time-varying data [163] [94]. In TDNN, delay units are added to a general static network, and some of the preceding values in a time varying sequence are used to predict the next value. Neural network is even more efficient when large amount of training data are available.

4.2.1.5 Syntactic techniques

Syntactic approaches come from the area of pattern recognition. For example, [16] uses a non-probabilistic grammar to recognize sequences of discrete behaviors. [59] presents a probabilistic syntactic approach that detects and recognizes temporally extended behaviors and interactions between multiple agents. Here, the recognition problem is divided in two levels. The lower level uses a standard probabilistic temporal behavior method, as HMMs. The second level is a stochastic context-free parser that uses the output of the lower level. The grammar and parser provide longer range temporal constraints, disambiguate uncertain low-level detection, and allow the inclusion of prior knowledge about the structure of temporal behaviors.

4.2.1.6 Non-deterministic finite automaton (NFA)

[151] uses NFA as a sequence analyzer. In fact, NFA is simple, instantaneous and purely non-deterministic. [151] recognizes multiple object behavior on behavior driven selective attention.

4.2.1.7 Self-organizing neural network (SONN)

Compare to the other methods, SONN is suited for behavior recognition when objects have unconstrained motion.

[63] describes objects motion by a sequence of flow vectors, each vector is representing the position and the velocity of the object in the image. Then, a statistical model of the object trajectories is created with two competitive learning networks connected with leaky neurons. [138] improves this model by introducing a feedback to the second network.

[104] applies a Kohonen self organizing feature map to find the flow vector distribution patterns and classify objects trajectories as normal or abnormal.

4.2.2 Human action recognition

Human action recognition aims at recognizing a set of specific actions in videos. Such videos can be extracted from movies, Internet clips, or video surveillance. Existing work can be separated in three categories

- Human model based methods use a full 3-D or 2-D model of human body parts. Action recognition is achieved using information on body parts pose and motion.
- Holistic methods use knowledge about the localization of humans in video and learn an action model from global body motion, without any notion of body parts
- Local feature methods only use descriptors of local regions in a video. No prior knowledge about the position of human or its limb is given.

Since our tracking system produces information regarding the localization and contour of the global human bodies without human model, only holistic methods can be applied on our system. Therefore, we present Holistic methods.

Holistic methods use the global body structure and its dynamics to represent human actions [114]. In fact, given a region of interest centered on the human body, global dynamics are discriminative enough to characterize human actions. Holistic approaches are much simpler than model based methods, since they only model global motion and appearance information. Thus, the computation is generally more efficient and robust.

We roughly divide holistic methods into two categories: methods based on shape masks or silhouette information and methods based on optical flow and shape information.

4.2.2.1 Shape mask and silhouette based methods

[161] is one of the first proposing such a method. The system computes a grid over the silhouette and calculates the ratio of foreground to background pixels. Then, training and testing is achieved using HMMs.

[13] uses shape masks from difference images to detect human actions and introduce the notion of temporal templates for action recognition. The system uses motion energy images (MEI) and motion history images (MHI). Specifically, MEIs are binary masks indicating regions of motion, and MHIs weight these regions according to the moment they occurred.

Recently, [158] proposes to recognize human actions based on accumulated motion image.

[12] and [43] create an action model based on space-time shapes from silhouette information. The silhouette is obtained by background subtraction. Then, features such as local saliency, action dynamics, shape structure, and orientation are extracted from the silhouette. These features are used to calculate an action descriptor on sets of 10 frames.

[166] uses contour information to obtain spatio-temporal shapes [12]. Actions are represented by sets of characteristic points (such as saddle, valley, ridge, peak, pit points) on the surface of the shape.

[154] represents actions sequences as vectors of minimum distance between silhouettes in the set of examples and in the sequence. Another measurement is added to silhouette information: Chamfer distance is calculated between silhouette examples and edge information contained in test sequences. Then, Classification is achieved with Bayes classifier with Gaussians modeling classes.

[167] computes a motion context descriptor over consistent regions of motion by using a polar grid. Each cell of the grid is described with a histogram over quantized SIFT [85]. Classification is achieved using support vector machine (SVM) [20] and probabilistic latent semantic analysis (PLSA) [49].

It is interesting to note that Silhouettes are a popular representation for surveillance applications [46], [53], and [130]. Since cameras are generally fixed, background subtraction can be applied to compute silhouette information. Action such as running, walking, carrying backpacks or large objects can be recognized. [118] uses a human tracker with camera motion estimation in order to cope with challenging data.

Silhouettes provide interesting information for action recognition. However, they are difficult to compute when clutter, camera motion, or occlusion occur. Moreover, they only describe the outer contours of a person and lose discriminative power for actions with self-occlusions.

4.2.2.2 Optical flow and shape based methods

[114] is the first to propose a human tracking framework with an action representation using spatio-temporal grids of optical flow magnitudes. By matching reference motion templates

of known periodic actions, such as walking, running, swimming, and skiing, the actions can be recognized.

[34] tracks soccer players and compute a descriptor on the players silhouette using blurred optical flow. The descriptor separates x and y flow measurements as well as positive and negative components. Further tests are achieved on tennis and ballet sequences.

[36] uses the same template and uses a two-layered AdaBoost [38] variant to classify the data. First, intermediate features are learned by selecting discriminative pixel flow values in small spatio-temporal blocks. The final classifier is then learned from a all previously accumulated intermediate features.

[123] uses spatio-temporal regularity flow information. Then, the system learns cuboid templates by aligning training samples via correlation. The classification is achieved by correlating test sequence with the learned template with generalized Fourier transform.

[68] computes spatio-temporal Haar features on optical flow components using an integral video structure. During learning, a discriminative set of features are chosen to optimally classify actions, which are represented as spatio-temporal cuboidal regions. The classification is achieved with a sliding window approach each position is classified as containing a particular action or not.

[86] presents a method purely based on shape information. This system tracks soccer or ice-hockey players and describe each frame with histograms of oriented gradients. Then, PCA is used to reduce dimensionality and HMMs classify actions.

[128] combines optical flow information and Gabor filter responses in a human-centric framework: both types of information are weighted and concatenated. PCA over all pixel values is applied to learn the most discriminative feature information

[79] proposes to localize drinking actions in movies. This system combines a set of appearance and motion features. Appearance is represented by histograms of oriented gradients and motion by histograms of optical flow. The system pre-filter possible action localizations with a human keypose detector trained on keyframes of the action.

[54] presents an approach based on multiple instance learning that cope with imperfect localizations of humans. The system uses histograms of oriented gradients, foreground segmentation, and motion history images as features. Results are presented on simple actions in crowded sequences and in challenging data recorded in a shopping mall. We note that this paper presents the only system detecting actions in a shopping setting.

Holistic approaches are suitable for action recognition in realistic video data. We further note that holistic representations are in general not invariant to camera view point and certain parts of the body might not be visible in the image. However, humans' localization reduces the complexity of detecting actions.

4.3 Proposed behavior recognition approaches

Introduction

Based on the object tracking process, see chapter 3, we want to characterize the behavior of these objects.

First, we present our behavior model that is composed of six states. These states represent the actions of interest for our application. Then, we focus on the "interact" state detection and description. The first approach detects the "interact" state deterministically. Then several descriptors are used to describe this state. The descriptors are based on local spatio-temporal templates or on interaction measurements. We further recognize all states and organize them in a finite state machine. We note that the descriptors of the "interact" state are recognized with support vector machines. A functional diagram is represented figure 4.1.

4.3.1 Behavior model definition

This section presents our model that defines interesting human behaviors while shopping. At a point of sale, customers walk around products, look at prices, pick up products, etc. We create six states that correspond to the current behavior of a person, or object. By recognizing these states at every frame, we generate the chain of states that can be used to describe a scenario that the person plays.

- Enter: A new person appears in the scene.
- Exit: The person leaves the scene.
- Interested: The person is close to products, i.e. possibly interested.
- Interacting: The person interacts with products, is grabbing products.
- Stand by: The person is in the scene and not close to any product area or image boundary. This person can be stopped or walking around products.
- Inactive: The person has left the scene.

4.3.2 Interaction detection and description

Introduction

We keep in mind that we want to build a real-world application. We first decide to create a simple software system where an operator can fill in the final system with a priori knowledge

about the scene. This software connects to a camera and allows a person to manually define the product areas within the image. The operator can also define the size of a person's hand

The second idea is to avoid a long training phase in the installation of our final system. We test various methods requiring training or not. At any time, we avoid long training phases by only using a few sequences to train our system.

In this section, we first present a deterministic method to detect the "Interact" state. Then, we deal with various probabilistic description of this state.

4.3.2.1 Deterministic detection

Since we know where products heaps are located, we classify detected regions depending on the location of their gravity center. If the gravity center of a region is located on a product area, the region is considered as products that are removed from the products heap, probably taken by a customer. Otherwise, the region should belong to a customer.

Detecting "products taken" corresponds to relevant information. However, such event occurs once the action is done and the customer is likely to be leaving.

Customers behavior is usually to think, look at the price, and then take a product. We want to know what they are interested in as soon as possible. Therefore, we detect customers occluding interest areas. Specifically, tracked object are considered as customer in the scene if they do not belong to a product area. At some point, a part of the customer can be detected as covering an interest area. When a surface that is larger than a theoretical hand size is detected, then a new event concerning the customer behavior is detected: the customer is about to grab a product. We remind that the hand size is known a priori through the initialization phase.

Finally we detect the "Interact" state when a customer is about to grab products, occluding a product area, or when a customer is close to a product area in which products are moved or removed: motion is detected in a product area.

4.3.2.2 Probabilistic description

The measurement presented above corresponds to direct interactions with products areas and is detected in a deterministic manner. However, we can represent interactions with products in a probabilistic way.

While grabbing a product, a person first reaches out with its arm, then grasps a product, and finally take the product. These different phases in the "product grabbing" event correspond to observable local motion of the person. Following the idea that similarity between various motions can be identified through spatio-temporal motion description, a corresponding descriptor has to be composed of sets of features sampled in space and time

[146] [34]. We further present various spatio-temporal descriptions of the objects motion to recognize "product grabbing event". These descriptors are based on motion history image (MHI), accumulated motion image (AMI), local motion context (LMC), interaction context (IC), and combined LMC and IC (MI).

Motion history image MHI is a temporal template used as model for actions [13]. MHI offers information concerning a person shape and the way it varies along a local period of time. We aggregate a sequence of foreground objects masks, scaled to a standard size of 120x120 pixels, see figure 4.2. MHI is computed as follows:

$$MHI(x,y) = \frac{1}{T}\sum_{t=1}^{T} I(x,y,t) \tag{4.1}$$

Where $I(x,y,t)$ is the pixel value of the image sequence I at position (x,y) and at time t. T is the time interval used to calculate the MHI, we choose $T = 15$.

We define two energy histograms by projecting MHI values along horizontal and vertical axis [158]. These energy histograms are calculated as follows:

$$EH_h(i) = \sum_{j=1}^{W} MHI(i,j),\ i = 1,\ldots,H \tag{4.2}$$

$$EH_v(i) = \sum_{i=1}^{H} MHI(i,j),\ j = 1,\ldots,W \tag{4.3}$$

Where H and W are relatively the height and width of the scaled image. We have $H = W = 120$. These two energy histograms are used as a 240 (120x2) dimensional descriptor to recognize interactions.

Accumulated motion image AMI [158] was inspired from MHI and Motion Energy Image (MEI) [13]. As we see in the previous section, MHI use the entire silhouette. However, only areas including changes are used to generate the AMI that is defined as follows:

$$AMI(x,y) = \frac{1}{T}\sum_{t=1}^{T} |D(x,y,t)| \tag{4.4}$$

Where $D(x, y, t) = I(x, y, t) - I(x, y, t - 1)$

We note that the image difference $D(x, y, t)$ is calculated between two scaled masks and we keep $T = 15$, see figure 4.2. We calculate the same energy histograms presented in the previous section that are used as descriptor (240 dimensions) to recognize interactions.

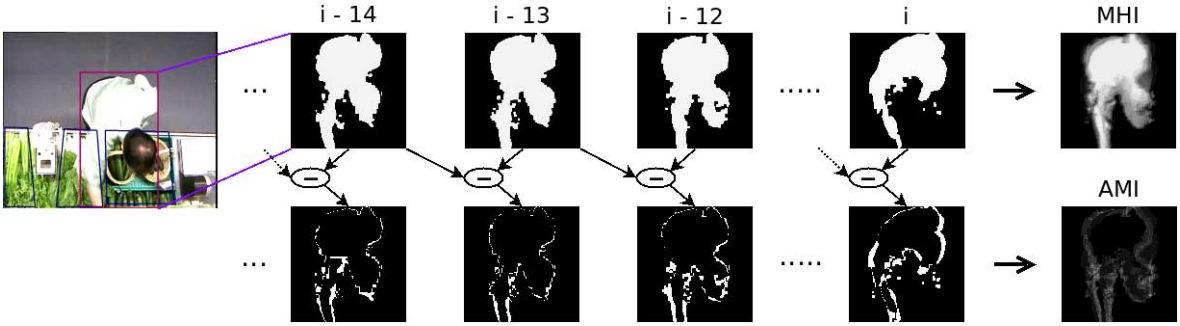


Figure 4.2: Diagram representing the MHI and AMI generation

Local motion context We then choose to describe motion using pixel-wise optical flow [34]. Since optical flow is not very accurate, we use histograms of features over image regions. Such a representation is tolerant to some level of noise, according to [146].

Local motion: first, each person's mask is scaled to a standard size of 120x120 pixels, while keeping aspect ratio. Then, the optical flow is computed using Lucas Kanade algorithm [87]. The result of this process is two matrices with values of motion vectors along x and y axis. We separate negative and positive values in the two matrices, and obtain four matrices before applying a Gaussian blur to reduce the effects of noises.

Silhouette: a fifth matrix, representing the person's silhouette, is computed from the scaled mask.

Data quantization: we reduce the dimensionality of these matrices to filter noises and save computation time. Each matrix is divided into a 2x2 grid. Each grid cell gets its values integrated over an 18-bin radial histogram (20 degrees per bin). Matrices are now represented by a 72 (2x2x18) dimensional vector.

Temporal context: to take into account temporal information, we use 15 frames around the current one and split them in three sets of 5 frames: past, current, and future. After applying Principal Component Analysis (PCA) on each set's descriptors, we keep the first 50 components for the current set, while we only keep the first 10 components for the past and future sets. We follow the intuition that local motion should be represented in better detail than more distant ones. The temporal context descriptor possesses then 70 (10+50+10) dimensions.

The final descriptor is composed of 430 (72x5+70) dimensions, see figure 4.3.

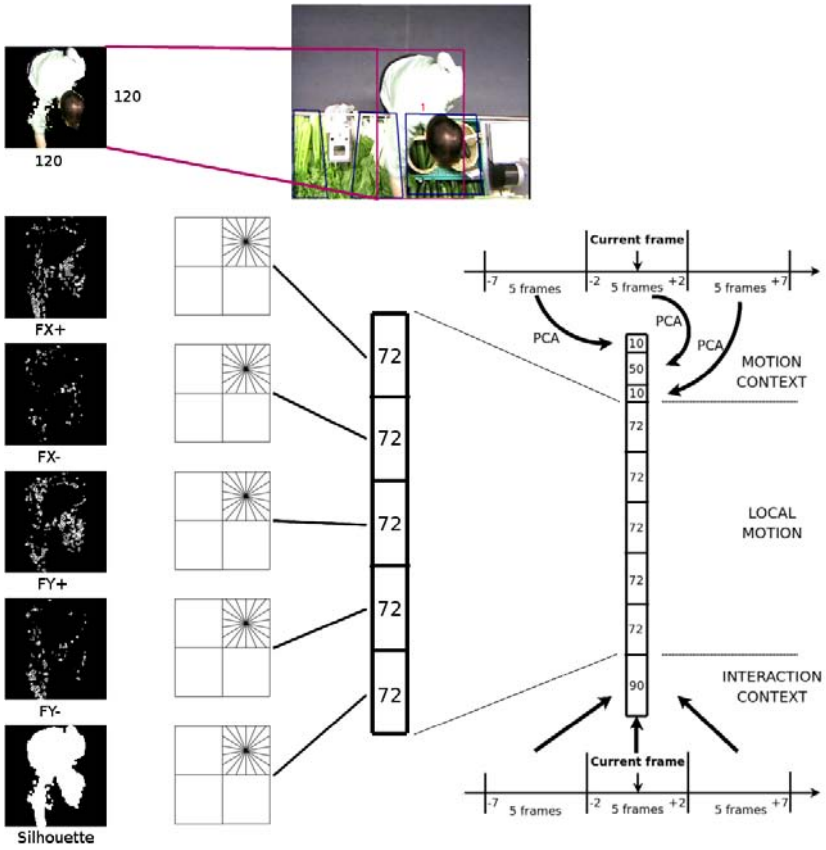


Figure 4.3: Diagram representing the combined LMC and IC descriptor

Interaction context This last descriptor is based on interactions with product areas. We use six measurements calculated as follow:

- The person's surface covering a product area.
- A Boolean that is true when this covering surface is bigger than a theoretical hand size or when a person is connected to a product area and there is motion detected on this area.
- The surface of the person.
- The height of its bounding box.
- The position of the bottom of the bounding box along y axis.
- The position of the top of the bounding box along y axis.

The first measurement increases when a customer is reaching out before taking a product. The second measurement detects motion in products areas. In fact, when a product is taken motion is detected where the product is missing. Furthermore, the measurements related to the height and position of the bounding box have meaningful variations as a person reaches out for products. The surface tends to increase as a person grasps a product, when products are big enough. These measurements fill the interaction context descriptor that possess 90 (6x15) dimensions, because we keep each measurement of the 15 frames around the current frame.

After running some tests, we decide to combine the local motion context and the interaction context description into one descriptor of 520 (430+90) dimensions, see figure 4.3.

4.3.3 Behavior recognition

4.3.3.1 States recognition

This section presents the behavior recognition process. Based on the behavior model, we detect the six states. Using object tracking and higher level information, for each frame, the state of each object has to be identified among the six predefined states:

- Enter is detected when a new person is detected and is connected to an image boundary.
- Exit is detected when a previously tracked person connects to an image boundary.
- Interested is detected when a person's contour connects a product area.
- Stand by is detected when a person is not connected to a product area or an image boundary. The person is further classified as moving or not.
- Inactive is detected when the system loses track of a person. This event mostly happens when a person leaves the scene, or is occluded by something in the scene or another person.
- Interact is detected deterministically, as in section 4.3.2.1, or using SVMs [22] on one of the presented descriptors, see section 4.3.2.2.

4.3.3.2 Support vector machine

SVM is a supervised learning method that classifies data. First, there are two phases in the data classification: training and testing. The data corresponds to several instances. An instance is composed of a "target value" and several "attributes". In our case, the target value is 0 or 1 depending if a product grabbing event, i.e. "Interact" state, occurs or not.

The attributes correspond to each value of our descriptor. SVMs create a model that predicts target value from attributes, by solving the following optimization problem.

$$min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{4.5}$$

with $y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$

Where y_i are target values, x_i are the attributes, ξ_i represent the error in the training set. The vector w and the scalar b are the parameters of the hyper-plane. $C > 0$ is the penalty parameter of the error term. Training vectors x_i are mapped into a higher dimensional space by the function ϕ. In this higher dimensional space, SVM finds a separating hyper-plane that maximizes the margin. The system uses a radial basis function (RBF) kernel:

$$K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j) = \exp(-\gamma \|x_i - x_j\|^2),\ \gamma > 0 \tag{4.6}$$

Although, different kernels exist, we choose RBF because it handles non-linear relations between attributes and target values, unlike linear kernel. RBF also has less hyper parameters and less numerical difficulties than polynomial or sigmoid kernels. Finally, two parameters must be identified C and γ. As proposed by [22], we use a grid to test various parameters values, for each dataset.

4.3.3.3 Finite state machine

A finite state machine is used to organize and prioritize the six states [19] [152].

The main advantage of FSM is that it is really flexible and allows adding or removing states. However, when adding a new state, state transitions must be added with caution.

The state machine we use is synchronous and deterministic. Synchronous means that the machine iterates over each new frame. Based on the previous state, the system calculates the new one by testing each transition condition. If a condition is satisfied, the system moves to the new state. Otherwise, the system stays in the same state. The machine is deterministic because for each state, there can not be more than one transition for each possible input. One FSM model the behavior of one person. We save the person's path through its FSM for higher level purposes. Figure 4.4 shows a possible path through the FSM.

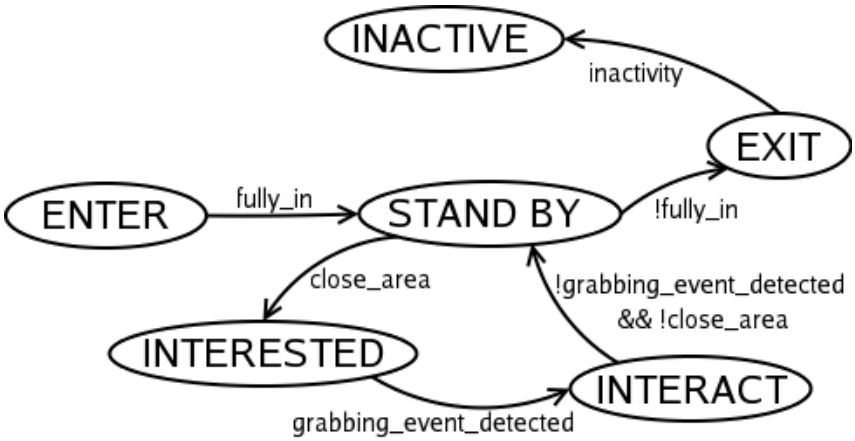


Figure 4.4: Diagram representing a path through the FSM. All transitions are not written to make it clear. We note that "&&" is a logic "AND" and "!" corresponds to "NOT"

4.4 Evaluation

4.4.1 Dataset description

We use various videos taken with the same camera, with 15 frames per second. Videos are grouped in datasets. Some are shot in our laboratory (LAB 1, LAB 2, and LAB 3). The others are taken in a shopping mall (MALL 1 and MALL 2).

The two first datasets (LAB 1 and MALL 1) possess five and six sequences respectively and contains a lot of interactions with products. Two and four different people are shopping respectively, see figure 4.5. However, there is only one person interacting with products at a time. Products taken by people have different shapes, colors, and sizes in the video sequences. Furthermore, all products are identical in the heaps.

LAB 2 presents actors that are not simultaneously in the scene and that are not interacting with products. Customers are just walking around the scene, see figure 4.6.

LAB 3 and MALL 2 are two datasets where multiple people interact together. Two to four people interact simultaneously in the scene, see figure 4.5.

4.4.2 Deterministic detection of the "Interact" state

In this section, we detect the "Interact" state when a person covers a product area with a surface area larger than a hand or when a person is next to a product area in which motion is detected.

(a) LAB 1 video 1 (b) LAB 3 video 2 (c) LAB 3 video 4

(d) MALL 1 video 4 (e) MALL 2 video 4 (f) MALL 2 video 5

Figure 4.5: Screenshots of various datasets

(a) frame 2 (b) frame 51 (c) frame 65 (d) frame 84 (e) frame 101

Figure 4.6: Screenshots of LAB 2 video 3

4.4.2.1 Comparison of both measurements

The result for the detection of motion in a product area is not very precise, see figure 4.8, 4.9, and 4.11. In fact, it is difficult to detect major changes when a product is taken from a heap of identical products, especially when the objects are small as in the LAB datasets. Furthermore, false negatives occur when the person occludes the taken object. However, it does not affect the results because the detected product merges with the person and then increases the occluding surface on the product area.

Customers covering a product area offers much better results, see figure 4.8, 4.9, and 4.11. However, false positives can occur due to shadows or when a person covers several interest areas at the same time. In such a case, the most occluded area is selected.

When these two measurements are combined, they offer a good interpretation of a picking up behavior in a specific area. Figure 4.11 gives correct results in a complex scene with multiple and simultaneous interactions.

Figure 4.7: Screenshots of LAB 1 video 1, on the top, and LAB 1 video 2, at the bottom

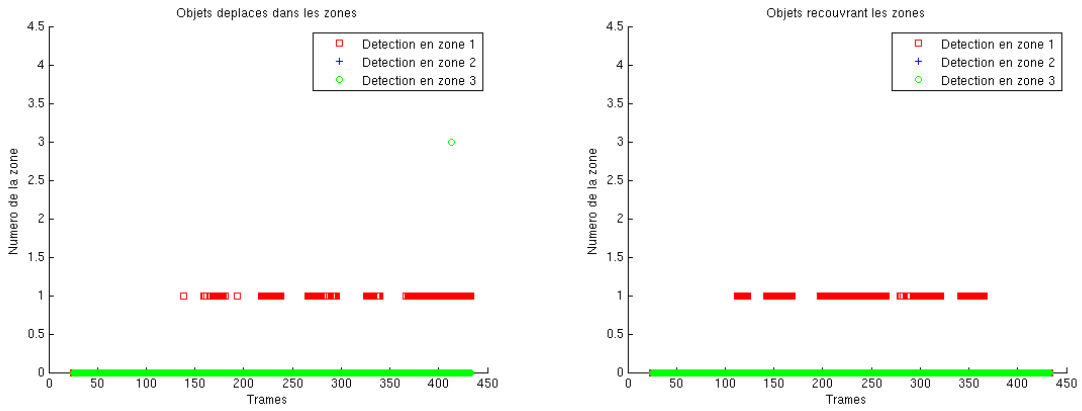


Figure 4.8: Diagrams representing motion detected in the corresponding product area, on the left, and an object covering a product area, on the right. The tested video is LAB 1 video 1, see figure 4.7

4.4.2.2 Recall - precision on datasets

We further calculate the recall and precision for the "Interact" state on several datasets, see figure 4.12. We note that the MALL datasets offer better results than LAB datasets because the camera is closer to the products. Therefore, products appear larger in the video.

We also note that the datasets with several customers interacting simultaneously offer results as good as datasets with only one person interacting at a time. Therefore, our determin-

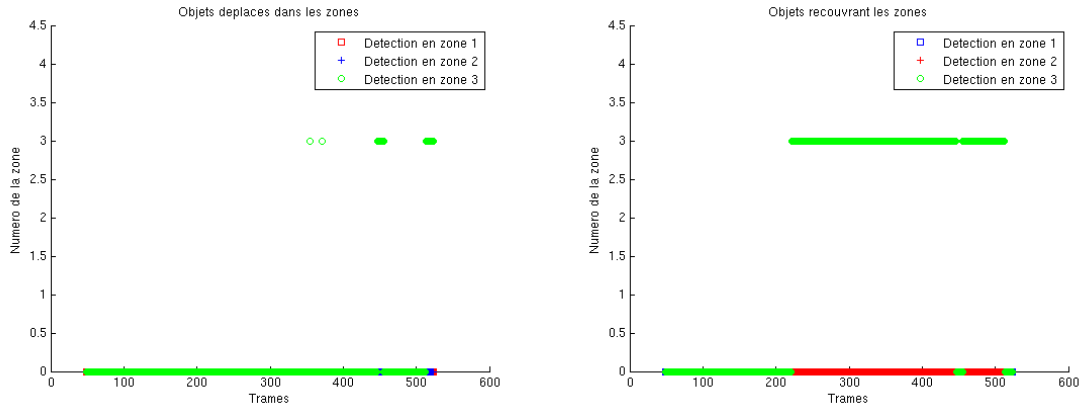


Figure 4.9: Diagrams representing motion detected in the corresponding product area, on the left, and an object covering a product area, on the right. The tested video is LAB 1 video 2, see figure 4.7

(a) frame 14 (b) frame 19 (c) frame 22

(d) frame 37 (e) frame 41 (f) frame 45

Figure 4.10: Screenshots of a video sequence with two customers interacting simultaneously

istic detection of the "Interact" state is robust to multiple people interacting simultaneously in the scene.

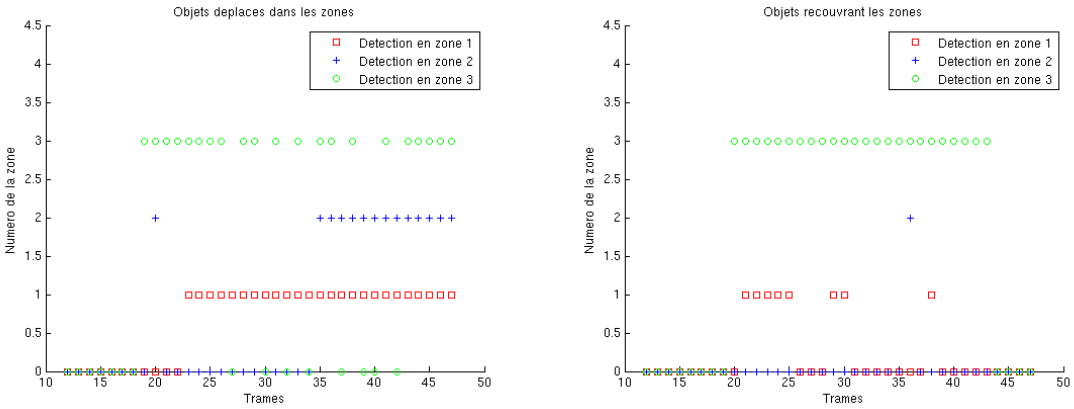


Figure 4.11: Diagrams representing motion detected in the corresponding product area, on the left, and an object covering a product area, on the right. The tests are achieved on the video presented on figure 4.10

Dataset	Video	Frames	Precision	Recall
LAB1	1	545	0,8558	0,6742
	2	672	0,7172	0,6698
	3	704	0,812	0,662
	4	771	0,5349	0,4978
	5	518	0,8174	1
	mean		**0,7475**	**0,7008**
MALL1	1	327	0,9005	0,795
	2	444	0,7692	0,9184
	3	434	0,5778	0,9512
	4	336	0,9326	0,9326
	5	164	0,8571	0,9231
	6	232	0,8444	1
	mean		**0,8136**	**0,9201**
LAB3 MP	1	212	0,9815	0,9217
	2	211	0,9789	0,7209
	3	300	0,7643	0,673
	4	303	0,7733	0,6591
	5	259	0,7311	0,58
	mean		**0,8458**	**0,7109**
MALL2 MP	1	215	0,9595	0,9467
	2	208	1	1
	3	735	0,7611	0,8566
	4	153	0,7429	0,8966
	5	382	0,7203	0,7103
	6	504	0,9136	0,8862
	mean		**0,8496**	**0,8827**

Figure 4.12: Precision - recall table for the deterministic detection of the "Interact" state on various datasets

4.4.2.3 Varying hand size

Finally, we test this method with varying hand sizes, see figure 4.13. We already note that MALL dataset offers better results than LAB.

The hand size is larger on MALL than LAB, due to the camera location. Moreover, the variation of the hand size has less effect on MALL 1 dataset than on LAB 1 dataset. Recall increases with the hand size and precision tends to decrease.

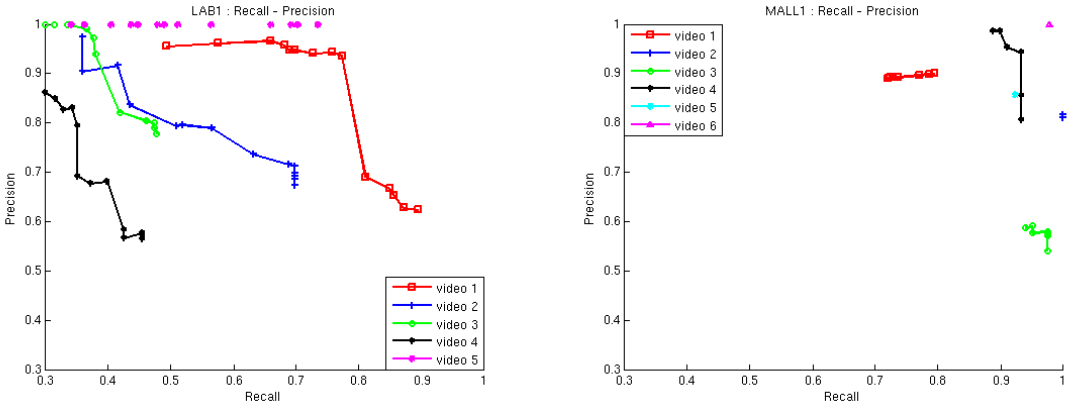


Figure 4.13: Recall - precision graph for the "Interact" state for the dataset LAB 1, on the left, and MALL 1, on the right. Hand size varies from 10% to 500%, on the left, and from 50% to 150%, on the right.

4.4.3 States detection

This section presents recognition results for every state, see figure 4.14. We detect the state of each object, at each frame, and then compare this state to the ground truth. We then calculate the percentage of correctly labeled frames for three datasets.

We note that even though "Interact" state is better detected on MALL datasets, the overall results of states detection results are better for LAB datasets. In fact, the other states are often not well detected on MALL datasets due to the camera location that does not allows to see people entirely in the scene. For example, a person can be detected as "Exiting" the scene because he is connected to an image boundary, while he is just walking around. As we see on figure 4.5, a person is looking interested in products and is connected to an image boundary, at the same time.

We further note that LAB 2 offers better results than LAB 1. Since LAB 2 is composed of simple actions, we can see that "Stand by", "Entering", and "Exiting" states are easier to detect than "Interact" for the LAB datasets.

Dataset	Video	Frames	Correctness
MALL 1	1	327	70,03%
	2	444	74,77%
	3	434	66,13%
	4	336	70,83%
	5	164	76,22%
	6	232	79,74%
	mean		**72,95%**
LAB 1	1	545	85,87%
	2	672	74,40%
	3	704	76,28%
	4	771	60,57%
	5	518	92,66%
	mean		**76,13%**
LAB 2	1	476	87,61%
	2	342	82,46%
	3	143	91,61%
	4	303	95,71%
	mean		**89,35%**

Figure 4.14: Percentage of correctly labeled state for various dataset

4.4.4 Probabilistic recognition of the "Interact" state

In order to recognize the "Interact" state, we use a cross validation process. In other words, to recognize events on a video, we use all the other sequences of the dataset as training and then calculate recall and precision on the detection of the state, see figure 4.15 and 4.16. It is interesting to note that using this process, we only use a few minutes of video as training with a few actors.

Dataset	Video	Frames	MHI R	MHI P	AMI R	AMI P	LMC R	LMC P	IC R	IC P	MI R	MI P
MALL1	1	327	0,4787	0,3261	0,4894	0,4646	0,5102	0,3937	0,5306	0,5977	0,8163	0,6667
	2	444	0,2178	0,2716	0,1386	0,5833	0,5149	0,4815	0,9307	0,9592	0,9505	1
	3	434	0,4919	0,7625	0,5403	0,7128	0,6822	0,869	0,6667	0,7368	0,7525	0,5802
	4	336	0,6386	0,5699	0,5783	0,6857	0,1148	0,1591	0,6905	0,9063	0,7976	0,8701
	5	164	0	0	0,125	0,0833	1	0,3333	0,5	1	0,5	1
	6	232	0,7797	0,7797	0,5085	0,4688	0,8852	0,6429	0,8475	0,9434	0,8983	0,9815
	mean		**0,4345**	**0,4516**	**0,3967**	**0,4998**	**0,6179**	**0,4799**	**0,6943**	**0,8572**	**0,7859**	**0,8498**
LAB1	1	545	0,1898	0,3514	0,6423	0,5946	0,0092	1	0,7299	0,7692	0,8321	0,8085
	2	672	0,1441	0,2133	0,036	0,1739	0,2697	0,3333	0,6585	0,648	0,6748	0,6288
	3	704	0,2581	0,48	0,4247	0,4031	0,5185	0,332	0,6774	0,9333	0,7473	0,9392
	4	771	0,0854	0,4242	0,2561	0,2979	0,153	0,5185	0,7203	0,7687	0,7552	0,7347
	5	518	0,2364	0,1711	0,3091	0,4146	0,9153	0,7013	0,9818	0,75	0,9818	0,7941
	mean		**0,1828**	**0,328**	**0.3336**	**0.3768**	**0,3731**	**0,577**	**0,7536**	**0,7738**	**0,7982**	**0,7811**

Figure 4.15: Recall - precision table for the "Interact" state using various descriptors on two datasets

Dataset	Video	Frames	Recall IC	Precision IC	Recall MI	Precision MI
MALL2 MP	1	215	0,8929	0,8333	0,8214	0,902
	2	208	0,6462	0,8235	0,7846	0,8947
	3	735	0,7151	0,7278	0,9101	0,72
	4	153	1	0,5106	0,5417	0,9286
	5	382	0,8333	0,8404	0,6583	0,8404
	6	504	0,7273	0,8571	0,8561	0,8828
	mean		**0,8025**	**0,7655**	**0,762**	**0,8614**
LAB3 MP	1	212	0,7282	0,8929	0,8738	0,9375
	2	211	0,9063	0,8969	0,7604	0,9012
	3	300	0,784	0,7538	0,856	0,7643
	4	303	0,7561	0,5636	0,7317	0,6383
	5	259	0,8158	0,6596	0,8026	0,6854
	mean		**0,7981**	**0,7534**	**0,8049**	**0,7853**

Figure 4.16: Recall - precision table for the "Interact" state using the two descriptors offering the best results, on multiple people datasets

4.4.4.1 Motion history image

MHI is tested on LAB 1 and MALL 1 datasets, see figure 4.15. MHI offers poor results, especially on LAB 1 sequences. In fact, the silhouette of a person does not present significant changes when a person picks up products in this dataset, due to the camera location.

MALL 1 offers better results due to the camera location that allows seeing a major difference on the silhouette of a person picking up products. However, we realize that the appearance of the customer is not necessarily preserved from one sequence to another. This phenomenon is due to the fact that some videos show customers with shopping cart or basket, detected as foreground.

4.4.4.2 Accumulated motion image

AMI is tested on LAB 1 and MALL 1 datasets, see figure 4.15, and offers much better results than MHI for LAB sequences. AMI better model the changes occurring when a person picks up a product with such a camera configuration.

AMI give similar results as MHI for MALL 1. AMI is also sensitive to the appearance changing of the customer, within the dataset sequences.

4.4.4.3 Local motion context

LMC is also tested on MALL 1 and LAB 1, see figure 4.15, and offers much better results for both datasets.

The quantized optical flow combined with the silhouette and temporal context presents a more accurate description that better model a person's motion while picking up a product.

Once again, we note that results on MALL 1 are better than LAB 1. The camera configuration allows us to see the customer from closer and with a better angle, see figure 4.5.

4.4.4.4 Interaction context

IC offers even better results than LMC on all datasets, see figure 4.15. We note that MALL datasets performs slightly better than LAB datasets. The difference is much smaller than the difference with the other methods.

We then compare results on multiple people datasets, see figure 4.16. Results are as good as on previous datasets. The IC descriptor is robust to multiple people interacting together.

4.4.4.5 Combined local motion context and interaction context

We then decide to combine the two descriptors offering the best results to test several datasets, see figure 4.15 and 4.16.

We compare result using only the IC descriptor and combined LMC and IC (MI) descriptor. MI performs as good as IC for precision, but offers better results for recall on the first datasets.

On multiple people datasets, MI performs slightly better than IC on average. In fact, LMC description tends to be noisier, due to occlusions. IC remains robust in these situations and the recognition rates in multiple people datasets are as good as in the first datasets.

4.4.5 Camera location

The camera position is an important factor for the overall quality of the system. The case we study is simple: taking object on a table. Thus, our choice of camera position is flexible. Knowing that the application will be used for different kinds of racks, we want to optimize this position. The basic question to solve is: "What do we observe?" "The customers or the products?". Although our most relevant observations come from the products, the customers offer a lot of meaningful information.

Various tests are made, with various camera locations. We note that when people or products are seen from a closer view point, the results are improved due to the resolution increase. It is also important to see the entire customer while he is picking up products. If a person is not fully in the image, the detection of "Stand by", "Enter", and "Exit" states are noisier.

For example, MALL performs better than LAB on recall and precision for the Interact state due to the position of the camera, located directly above the products and closer on MALL than on LAB, see figure 4.5. Thus, a position close to the products performs better

on Interact state recognition. However, having the camera too close to the products make us lose information about the customers, since they are only detected when they are near the products.

Another major point is that when people cover large areas of the image, the field of view is reduced and the noises generated by the camera become more significant. Auto-focus generates blurs; auto-white-balance and auto-iris generate light changes. In fact, when an object that significantly changes the image contrast appears, pixels values of the entire image are modified, by these camera functions. It is important to fix these functions or to effectively compensate them.

4.5 Conclusions

In this chapter, we present the behavior analysis phase of our system. We first define a behavior model composed of six states. These six states are organized in a finite state machine that models the behavior of each tracked object in the scene.

Then, we focus on the Interact state that model interaction between customers and products of the point of sale. This state corresponds to a product grabbing event and is detected using various methods. First we detect interaction using a deterministic method. Then, we build various descriptors: MHI, AMI, LMC, IC, and MI. These descriptors are used to detect the "Interact" state with SVMs.

The evaluation shows us that the deterministic method offers good results for a final system that would not use a training phase. Then, probabilistic methods offer promising results by combining LMC and IC descriptors. We note that we do not require a long training to obtain good results with this method.

Chapter 5

Interpretation and scenario recognition

5.1 Introduction

This chapter presents two high-level analysis processes. These processes are based on the behavior analysis module, presented in chapter 4, that detects the state of action of each person in the scene along the sequence, see figure 4.1.

First, a semantic interpretation is generated that describes the actions achieved by the actors in the video. Specifically, sentences are generated when state changes occur. This interpretation produces semantic information that is suitable for non-specialist.

Then, we recognize three different scenarios that summarize the actions of each actor, in a video sequence. Such information is useful for statistical analysis about the number and interest of customers in specific areas. Moreover, these statistics can be used as a measure for media content efficiency.

5.2 State of the art

Semantic description of behavior aims at generating sentences in natural language to describe actions taking place in a video. Many applications require a description of object behavior in natural language, suitable for non-specialist operator [82].

There are two main categories of behavior description methods: statistical models and formalized reasoning.

5.2.1 Statistical models

Bayesian network model is a representative statistical method [55] [120]. This model interprets events and behaviors by analyzing time sequences and statistical modeling. [119] uses a two-layer agent-based Bayesian network to describe interactions between several objects. However, these methods use motion concepts based on low-level recognition. In order to use high-level concepts, such as events or scenarios, these methods require high-level reasoning based on a large amount of prior knowledge.

5.2.2 Formalized reasoning

These methods require symbol systems, which represent behavior patterns, and reasoning methods, such as prediction logic, to recognize and classify events [99]. [71] and [70] propose to generate natural language descriptions of human behaviors in videos. First, head region of a human is detected in each frame. The 3-D pose and position of the head are estimated using a model-based approach. Then, head trajectory is divided into segments of monotonous motion. For each segment, the degree of change of pose and position and the relative position to other objects are calculated. Meanwhile, verbs and syntactic elements are matched to the object behaviors and text is generated using a machine translation technique.

[74] presents a method that uses fuzzy membership functions. These functions associate verbs with quantitative measurements obtained by analyzing the image sequence. In this method, each occurrence is defined by three predicates: precondition, monotonicity condition, and post-condition.

The main disadvantage of formalized reasoning is that it does not handle uncertainty of events [81].

Description of behavior presents several issues: how to represent semantic concepts, how to map motion characteristics to semantic concepts, how to choose a good representation to interpret the meaning of the scene.

5.3 Semantic interpretation

This section aims at describing people's behavior in natural language. Since the behavior analysis system detects the current state of action of a person, we want to generate sentences that summarize the actions and events achieved by each person in a video.

It is interesting to note that there is a semantic gap between video analysis information and conceptual information described in natural language.

In our approach each state, detected in the behavior analysis phase, corresponds to a concept of human action. We use case frames to express these actions. Case frames consist

of syntactic components of a sentence. Then, we build a state transition model based on a hierarchy of actions. Such a model allows dynamic state changes and the generation of sentences along a video sequence.

5.3.1 Case frames

A case frame is a frame expression that represents relationships between cases in a sentence. Fillmore [37] classifies cases into eight categories: agent, locus, source, predicate, etc. In our study, we use simple case frames composed of three categories as illustrated in the following example:

[AG: "Person 1", PRED: "is interested in", LOC: "area 2"]

Where AG, PRED and LOC are the agent, the predicate, and the locus of the described action, respectively. These three cases allow us to describe all the actions we are interested in.

5.3.2 Hierarchy of action

Sentences are built using a hierarchy of actions (HoA). In a HoA, each node is represented by a case frame.

Figure 5.1 presents an example of hierarchy of action. In this example, a parent node [PRED: be, LOC: 0, ...] derives two child-nodes.

- One node is the redefinition of the verb: verb "is interested in" is derived from verb "be", when the measurement *Interested* is detected.
- The second node is the redefinition of the locus. The locus "in the scene" is derived from "0", when the measurement *Fully in* is detected. Information about the location of the action is added to the locus.

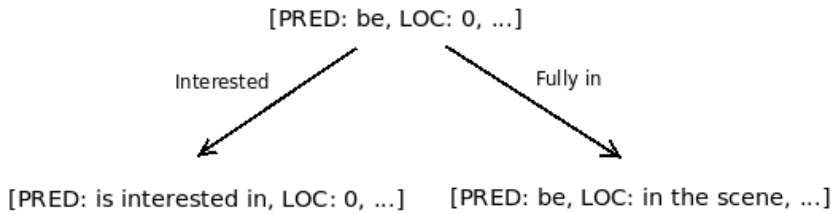


Figure 5.1: Example of hierarchy of action

In particular, by going through the hierarchy of action, the case frame is refined until a final node is reached. Final nodes do not have children and contains all the components of

the sentence we are looking for, see figure 5.2. The final nodes predicates of our HoA are listed as follow:

- "is walking around"
- "is stopped"
- "enters the scene"
- "exits the scene"
- "is interested in"
- "interacts with"
- "is gone"

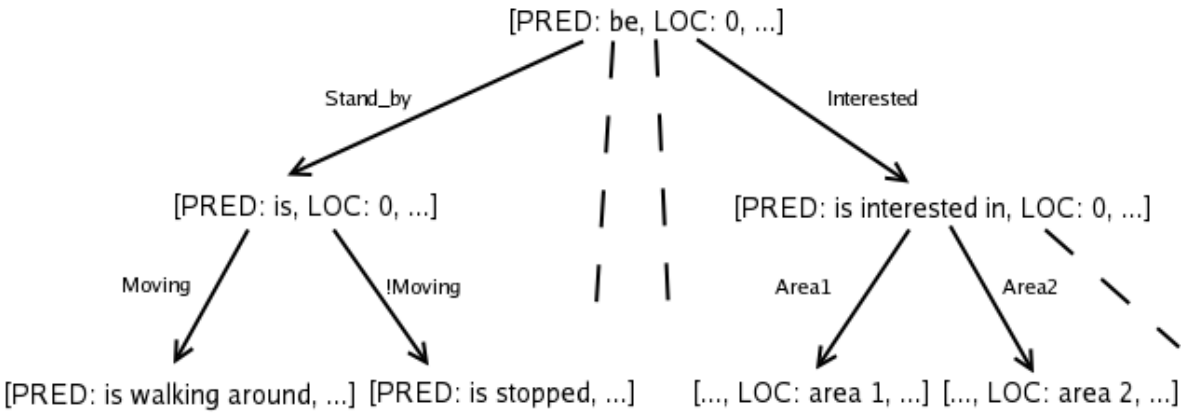


Figure 5.2: Sample of the hierarchy of action

However, to follow rapidly changing states and save processing time, we use a different model: a state transition diagram [71].

5.3.3 State transition diagram

We generate states for each type of verb and each locus and construct a State Transition Diagram (STD). As most of the analysis is completed in the previous phase, with the FSM, the STD is simplified, see figure 5.3. The STD has 11 states with no children that correspond to the 11 possible sentences that can be printed.

We explain, in the algorithm that follows, how case frames are generated from the STD.

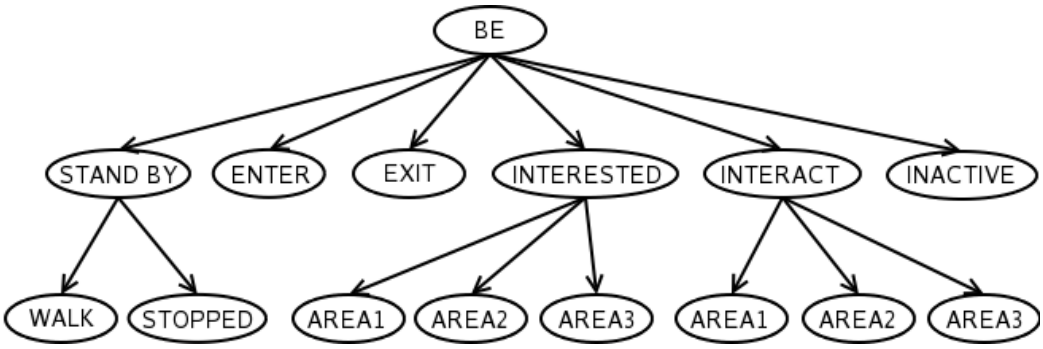


Figure 5.3: State transition diagram for three areas of interest, or product areas

1 Let s be a current state. For a new position of the tracked person, semantic primitives are evaluated downward from the top of the STD to determine the new state s'.

2 If state s and state s' are identical, no case frame is generated.

3 Otherwise, a case frame associated to the new state s' is generated.

Specifically, when a state change occurs, a case frame is generated and a sentence is printed.

It is interesting to note that we print sentences using present tense or progressive form, because the sentence is usually printed at the beginning of an action that last for several frames.

5.3.4 Filtering results

After generating sentences at every state change, we notice that the amount of information is too large. Furthermore, when a state change is detected for only one frame, meaningless sentences are generated. Therefore, we filter sentences by checking that the person remains in the same state for a couple of frames, to print meaningful sentences. We avoid writing two sentences when a person state is stationary and switches for one frame to another state before returning to the original state, for example.

We further filter out small objects or objects detected for a few frame. These objects often correspond to noises detected when a person enters the scene, for example.

5.3.5 Evaluation

Semantic interpretation produces interesting results, see figure 5.9, 5.10, 5.11, 5.12, 5.13, 5.14, and 5.15.

Since this interpretation relies completely on the state recognition, a few errors occur. In particular, videos from MALL datasets generate errors due to miss-detections of the "Exit" state. In fact, the camera is very close by the product and when a person is standing by the products, this person is usually connected to an image boundary. Then "Exit" state is detected instead of "Stand by", see figure 5.13 frame 34 and figure 5.14 frame 39.

Moreover, the filtering of small object results in the non-detection of the "Enter" state. Since an object entering the scene is not fully detected, the object is filtered out because it is too small, see figure 5.10, 5.11, 5.13, 5.14, and 5.15.

5.4 Scenario recognition

Chapter 4 identifies the state of tracked objects at every frame of the video. These states correspond to actions achieved by the tracked people. In this section, we aim at defining and recognizing specific high-level scenarios. These scenarios correspond to a series of actions and summarize the achievement of a customer in the scene. Specifically, we want to differentiate people passing by from people interested or interacting with products.

For our application, it is interesting to detect various scenarios that describe different customer behaviors. Depending on the scenario occurring, the media communicates differently with the customers.

We build a temporal structure that represents a series of human actions. This model, which detects the occurrence of actions, is described using a constraint network based on Allen's theory. Constraints are imposed by the video analysis measurements and the previous state of the network. A network is built to describe each scenario. By checking the validity of each network we determine the scenario played by the customer.

5.4.1 Temporal relationships definition

As we see in chapter 4, the FSM is a method that allows determining the current state of action of a person based on its previous state and on the video analysis measurements. Thus, we use Allen's theory that allows us to define more complex scenarios with extra temporal relationships between states.

Allen's interval algebra [6] defines 13 relationships between two time intervals: equal (e), before (b), meet (m), overlap (o), during (d), start (s), finish (f), and the inverses relations, ib, im, io, id, is, if, see figure 5.4. We use these relationships as a base to describe the scenarios.

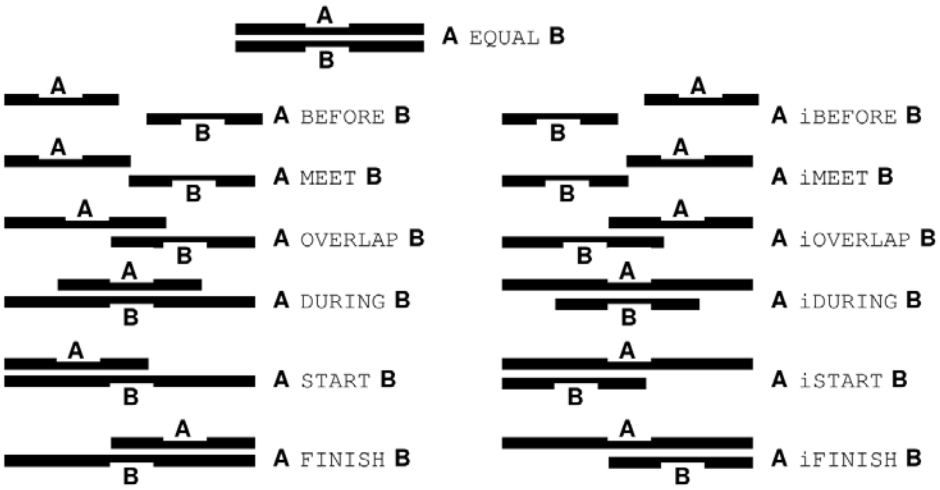


Figure 5.4: The 13 time relationships between two time intervals [6]

5.4.2 Scenarios description

We use "Interval algebra constraint network" [5], or "IA-network" [77] [111], to represent temporal structure. The network variables are time intervals and arcs, which are binary temporal constraints between intervals.

In our approach, we use the states of the FSM as the time intervals. In fact, each detected state of the FSM is an event that last for a certain interval of time. However, we can not use all the possibilities of Allen's theory, because all the states are mutually independent in the deterministic FSM. Therefore we use only four relations, as temporal constraints: before, meet, and their inverse ibefore and imeet.

Using these relations, we build three different constraint networks that represent the following scenarios:

1 Person walking by

2 Person walking by and being close to a products

3 Person walking by and interacting with products

The corresponding networks are shown in figures 5.5, 5.6, and 5.7. *m* and *b* on the transitions represent the relations meet and before, *im* and *ib* are the inverse relations respectively. We note that *A* meet *B* means that the event *B* starts as soon as event *A* finishes.

The first scenario is simple and can be thought of as a sequence of states: Enter - Stand by - Exit - Inactive.

As we can see on figure 5.5 the relationship between “Enter” and “Inactive” are meet and before, meaning that “Enter” meets “Inactive” or “Enter” occurs before “Inactive”. These relationships are the only ones that are authorized between the two states, for this network.

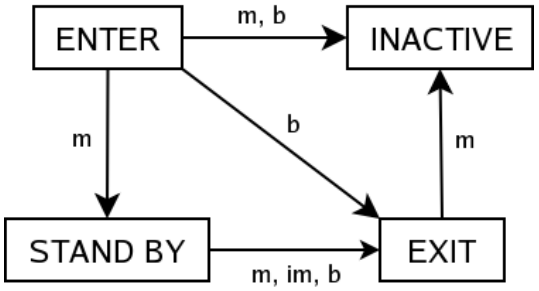


Figure 5.5: Scenario 1 constraint network

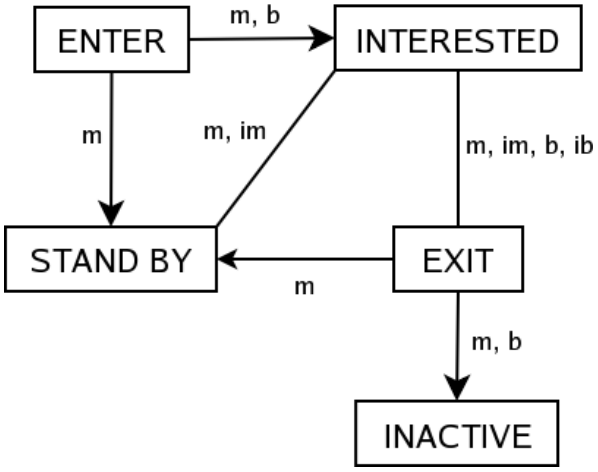


Figure 5.6: Scenario 2 constraint network

5.4.3 Scenario recognition

This section presents how the three scenarios are recognized. For each tracked person, we build the three networks. As a person passes from one state to another in the FSM, we check the validity of each transition in the networks. As soon as two scenarios are no longer valid, the system prints a sentence suggesting that the person plays the only valid scenario. Once the person leaves the scene, or is “Inactive”, we check the validity of each scenario. Then the system prints the more likely scenario that was achieved by the tracked person, as well as its

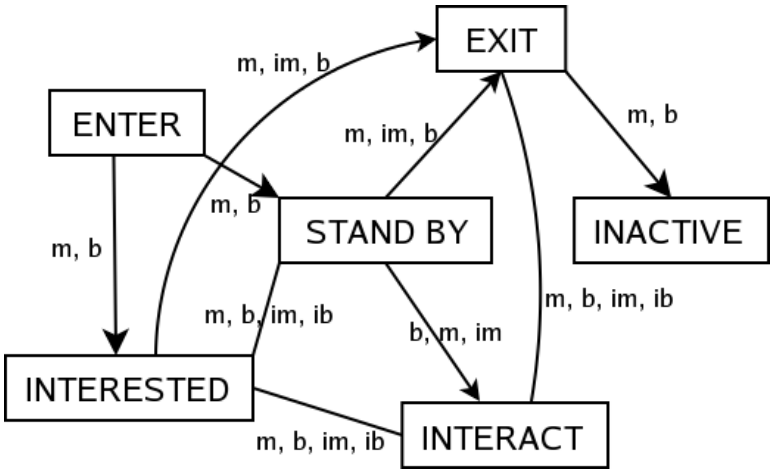


Figure 5.7: Scenario 3 constraint network

validity score. This score relates the number of valid transitions divided by the total number of transitions.

5.4.4 filtering results

As in section 5.3, we filter the results to make them clear and meaningful. The system does not print information regarding small objects and objects that are tracked on a few frames. These objects are likely to correspond to noises.

5.4.5 Evaluation

Scenario recognition offers very good results. For every video of the datasets MALL 1, LAB 1, LAB 2, MALL 2, and LAB 3, the scenarios played by the tracked objects are correctly identified, even if states are not perfectly recognized. However, a few sequences are not properly modeled. In fact, we see in chapter 3 that the tracking process can generate errors when a person leaves the scene while another one enter. Even if the scenario is correctly recognized, the identity of the two persons is interchanged. Furthermore, when several customers are occluding each other, the tracking process can not separate them during the occlusions. Thus, only one tracked object's scenario is identified, see figure 5.15.

Figure 5.8 shows the percentage of correct state transitions for all datasets. We note that MALL datasets offers slightly worse results. We note the same phenomenon for multiple person datasets.

We detect the scenarios in two different ways. First we detect scenarios on-line, and suggest a scenario as soon as the two others are invalid. It is interesting to note that this phase does not always offer perfect results, see figure 5.11, 5.12, 5.13, and 5.14. The second

method checks the entire path, once the person leaves the scene, and look for the closest scenario. This second process offers correct results for every video tested, see figure 5.9, 5.10, 5.11, 5.12, 5.13, 5.14, and 5.15.

Dataset	Video	Frames	Scenario	Dataset	Video	Frames	Scenario
MALL1	1	327	98,65%	**MALL2 MP**	1	215	96,70%
	2	444	98,68%		2	208	95,71%
	3	434	100%		3	735	96,98%
	4	336	99,65%		4	153	98,97%
	5	164	96,46%		5	382	98,33%
	6	232	98,34%		6	504	96,49%
	mean		**98,63%**		*mean*		**97,20%**
LAB1	1	545	100%	**LAB3 MP**	1	212	96,55%
	2	672	98,96%		2	211	98,90%
	3	704	100%		3	300	98,47%
	4	771	100%		4	303	97,62%
	5	518	100%		5	259	98,81%
	mean		**99,79%**		*mean*		**98,07%**
LAB2	1	476	95.82%				
	2	342	98,60%				
	3	143	100%				
	4	303	100%				
	mean		**98,61%**				

Figure 5.8: Table representing the validity scores for various videos

5.5 Conclusions

This chapter presents two high-level analysis processes. First, a semantic interpretation is generated that describes the actions achieved by the actors in the video. Then, we recognize three different scenarios that summarize the actions of each actor. Both processes are based on the behavior analysis module, presented in chapter 4, that detects the state of action of each person in the scene.

The semantic interpretation is based on these states and generates sentences when state changes occur. This interpretation produces semantic information that is suitable for non-specialist. Furthermore, the interpretation offers good results that are directly depending on

the state detected in the behavior analysis phase. We note that the interpretation filters the states to improve the results.

The scenario recognition generates a summary of a video, by detecting each person and categorizing its behavior. Such information is useful for statistical analysis about the number and interest of customers in specific areas. Moreover, these statistics can be used as a measure for media content efficiency. The scenario recognition offers very good results since all tracked object are correctly labeled. However, a few objects are not properly identified due to the object tracking system limitations.

We finally note that the full system can analyse 6 to 10 frames per seconds, for an image resolution of 704X576 or 640x480, respectively.

(a) frame 31 (b) frame 92

(c) frame 114 (d) frame 414

Frame 1 : No one is in the scene

Frame 25 : The person 1 enters the scene

Frame 32 : The person 1 is walking around

Frame 74 : The person 1 is interested in area 2

Frame 103 : The person 1 is interested in area 1

Frame 113 : The person 1 interacts with area 1

Frame 124 : The person 1 is interested in area 1

Frame 143 : The person 1 interacts with area 1

Frame 320 : The person 1 is interested in area 1

Frame 345 : The person 1 interacts with area 1

Frame 399 : The person 1 is interested in area 2

Frame 413 : The person 1 is interested in area 3

Frame 420 : The person 1 is walking around

Frame 429 : The person 1 exits the scene

Frame 434 : No one is in the scene

Scenarios :

Frame 113 : Person 1 plays scenario 3

Frame 434 : Person 1 had scenario 3 valid all along

Figure 5.9: Semantic interpretation and scenario recognition results for LAB 1 video 1

(a) frame 51 (b) frame 65 (c) frame 101

Frame 1 : No one is in the scene

Frame 54 : The person 1 is walking around

Frame 101 : The person 1 exits the scene

Frame 115 : No one is in the scene

Scenarios :

Frame 101 : Person 1 plays scenario 1

Frame 115 : Person 1 had scenario 1 valid all along

Figure 5.10: Results for LAB 2 video 3

(a) frame 44 (b) frame 64 (c) frame 96

(d) frame 118 (e) frame 184

Frame 1 : No one is in the scene

Frame 44 : The person 0 is interested in area 1

Frame 62 : The person 0 interacts with area 2

Frame 68 : The person 0 is interested in area 1

Frame 78 : The person 0 interacts with area 1

Frame 98 : The person 2 is interested in area 3

Frame 107 : The person 0 is interested in area 1

Frame 113 : The person 0 interacts with area 1

Frame 116 : The person 2 interacts with area 3

Frame 126 : The person 0 is interested in area 1

Frame 136 : The person 2 is interested in area 3

Frame 149 : The person 2 interacts with area 3

Frame 155 : The person 2 is interested in area 3

Frame 179 : The person 2 interacts with area 3

Frame 193 : The person 2 exits the scene

Frame 194 : The person 2 is gone

Frame 194 : The person 0 exits the scene

Frame 200 : No one is in the scene

Frame 75 : There is no perfectly valid scenario for the person : 0

Frame 136 : The person 2 plays scenario 3

Frame 190 : Scenario 3 was valid for the object : 0 - Correctness : 162 / 165

Frame 194 : Person 2 had scenario 3 valid all along

Figure 5.11: Results for LAB 3 video 2

(a) frame 14 (b) frame 19 (c) frame 22

(d) frame 37 (e) frame 41 (f) frame 45

Frame 1 : No one is in the scene

Frame 12 : The person 1 is stopped

Frame 13 : The person 1 enters the scene

Frame 15 : The person 1 is interested in area 3

Frame 16 : The person 3 enters the scene

Frame 19 : The person 3 is interested in area 1

Frame 20 : The person 1 interacts with area 3

Frame 21 : The person 3 interacts with area 1

Frame 37 : The person 3 interacts with area 2

Frame 38 : The person 3 interacts with area 1

Frame 41 : The person 3 exits the scene

Frame 43 : The person 3 is gone

Frame 44 : The person 1 exits the scene

Frame 48 : No one is in the scene

Scenarios :

Frame 12 : Person 1 has no perfectly valid scenario

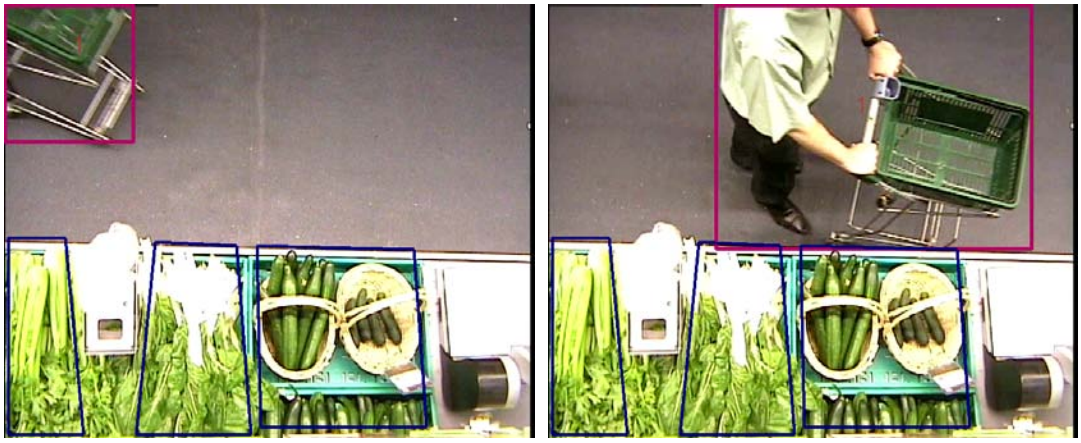
Frame 21 : Person 3 plays scenario 3

Frame 43 : Person 3 had scenario 3 valid all along

Frame 48 : Scenario 3 was valid for the person 1

Correctness : 33 / 35

Figure 5.12: Results for a video sequence with two people interacting simultaneously



(a) frame 38 (b) frame 70

(c) frame 149 (d) frame 204

Frame 2 : No one is in the scene

Frame 32 : The person 1 is walking around

Frame 34 : The person 1 exits the scene

Frame 82 : The person 1 is interested in area 3

Frame 93 : The person 1 interacts with area 3

Frame 188 : The person 1 is interested in area 3

Frame 200 : The person 1 exits the scene

Frame 208 : No one is in the scene

Scenarios :

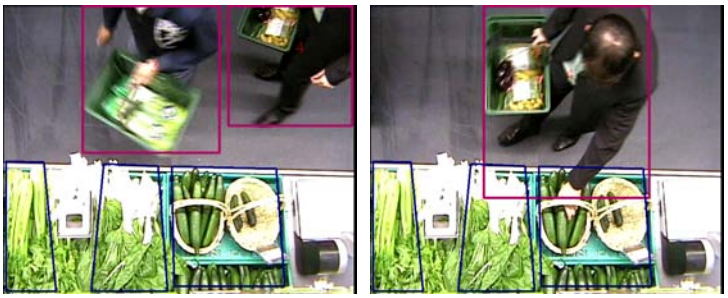
Frame 82 : Person 1 has no perfectly valid scenario

Frame 208 : Scenario 3 was valid for the person 1

Correctness : 178 / 181

Figure 5.13: Results for MALL 1 video 6

(a) frame 60 (b) frame 74 (c) frame 111



(d) frame 129 (e) frame 147

Frame 2 : No one is in the scene

Frame 39 : The person 1 exits the scene

Frame 58 : The person 1 interacts with area 3

Frame 70 : The person 1 is interested in area 3

Frame 80 : The person 1 interacts with area 3

Frame 126 : The person 1 exits the scene

Frame 128 : The person 4 enters the scene

Frame 139 : The person 4 interacts with area 3

Frame 147 : The person 1 is gone

Frame 157 : The person 4 is interested in area 2

Frame 164 : The person 4 exits the scene

Frame 191 : The person 4 is gone

Frame 200 : No one is in the scene

Frame 79 : Person 1 has no perfectly valid scenario

Frame 147 : Scenario 3 was valid for the person 1

Correctness : 114 / 118

Frame 177 : Person 4 has no perfectly valid scenario

Frame 191 : Scenario 3 was valid for the person 4

Correctness : 62 / 64

Figure 5.14: Results for MALL 2 video 1

(a) frame 45 (b) frame 58 (c) frame 97

Frame 2 : No one is in the scene

Frame 47 : The person 1 is interested in area 3

Frame 55 : The person 1 interacts with area 3

Frame 72 : The person 1 plays scenario 3

Frame 97 : The person 1 exits the scene

Frame 119 : Scenario 3 was valid for the person 1

Frame 121 : No one is in the scene

Correctness : 96 / 97

Figure 5.15: Results for MALL 2 video 4

Chapter 6

Conclusions and perspectives

This thesis presents our work achieved in various areas of computer vision. Our goal is to build a real-time system analyzing human behavior in a shopping setting. The system is composed of various modules on different level.

The low-level layer is composed of motion detection and object tracking that are presented in chapter 2 and chapter 3, respectively. The mid-level layer is the human activity analysis phase, see chapter 4. Finally, the high-level layer is composed of two separated processes: semantic interpretation and scenario recognition presented in chapter 5.

6.1 Motion detection

The chapter 2, about motion detection, presents and evaluates several methods.

First, we present frame differencing, a techniques without background model, which is very fast. However, detection results are poor.

Then temporal averaging, which uses a background model, remains fast and offers much better results. However there are several limitations due to camera motion and lighting changes.

To solve these issues, we test approaches using a multi-modal model of the background, such as Bayes decision rules (BDR) classification [80], Gaussian mixture model [136], and our method: improved Gaussian mixture model (iGMM).

iGMM and BDR offer accurate results and handle camera motion, light changes, shadows, and stopped objects. However, iGMM is about 5 times faster than BDR.

Motion detection techniques have been studied intensively for more than 10 years now and many methods were presented. However, there are still some interesting methods from data analysis that are adapted to the context of motion detection. For example, global model of the background are built using principal component analysis [149].

6.2 Object tracking

The chapter 3 presents and compares various object tracking approaches.

First we present four existing algorithms based on connected components, mean-shift, particle filtering, and combined connected component and particle filtering.

Then, we present our system based on multiple hypothesis that achieve tracking using multiple level analysis.

After evaluating the 5 methods, we conclude that our system can track several rigid or non-rigid objects, is able to track objects in outdoor and complex scenes, and is robust to several types of occlusions.

Since our method is based on the motion detection technique presented in chapter 2, the advantages and limitations of the tracking are linked to the motion detection technique used. The advantage is that motion detection offers very accurate detection of the object contour and the main limitation is that occluding objects are merged together and are not distinguishable.

As motion detection, object tracking has been studied intensively already. Most of the existing systems are adapted to specific cases and proposing an outstanding method coping with various types of scene is a tough challenge. Mixing existing methods can be an interesting work to improve genericity and precision.

6.3 Human activity analysis

The chapter 4 presents the behavior analysis phase of our system. We first define a behavior model composed of six states. These six states are organized in a finite state machine that models the behavior of each tracked object in the scene.

Then, we focus on the Interact state that model interaction between customers and products of the point of sale. This state corresponds to a product grabbing event and is detected using various methods. First we detect interactions using a deterministic method. Then, we build various descriptors: motion history image, accumulated motion image, local motion context, interaction context, and combined local motion context and interaction context. These descriptors are used to detect the “Interact” state with SVMs.

The evaluation shows us that the deterministic method offers good results for a final system that would not use a training phase. Then, the probabilistic method offering the best results is the combination of LMC and IC descriptors. We note that we do not require a long training to obtain good results with this method.

Human behavior analysis is a younger field of research and there are many open issues remaining. Several contests are being held to enable the detection of more complex actions.

For example, the conference CVPR 2011 [3] proposes a challenge where the goal is to recognize behavior such as "people getting in a vehicle", "people loading or unloading a vehicle", etc. Other contests aim at detecting or retrieving actions in movies [133] [78].

6.4 Interpretation and scenario recognition

The last chapter presents two high-level analysis processes. First, a semantic interpretation is generated that describes the actions achieved by the actors in the video. Then, we recognize three different scenarios that summarize the actions of each actor. Both processes are based on the behavior analysis module, presented in chapter 4, that detects the state of action of each person in the scene.

The semantic interpretation is based on these states and generates sentences when state changes occur. This interpretation produces semantic information that is suitable for non-specialist. The precision of the results directly depends on the state detected in the behavior analysis phase.

The scenario recognition generates a summary of a video, by categorizing the behavior of each detected person. The scenario recognition offers very good results since all tracked object are correctly labeled. However, a few objects are not properly identified due to the object tracking system limitations.

These two high-level processes are strongly linked to our behavior analysis system. Semantic interpretation of behavior and scenario recognition are sub-parts of behavior analysis. Nowadays, semantic analysis is studied in various contexts: scene analysis, image retrieval, etc.

6.5 Project perspectives

Concerning the project, there are several possibilities of improvement.

A first step would be to perform tests on a bigger scale: set up the system to work all day long for several days in a raw and in various locations. To set up such a system, there are many decisions to be taken concerning the content of the digital media. There are several possible scenarios: playing a specific video linked to a specific product when a person interact with it, play various videos and analyze their impact on customer, etc. Realizing such test would definitely present new issues or axes of improvements.

Improvement could be achieved on the motion detection level by parallelizing the motion detection process. Updating the background and testing if pixels belong or not to the background are two process that are performed on pixels independently. Therefore, these two processes can be parallelized.

The tracking system could be further tested by modifying the regions descriptors. We can imagine using a Kalman filter to predict the next position of the image gravity centre for example. We could also build a color histogram and add it to the region descriptor.

Other behaviors could be recognized. Further methods can be implemented to analyze object trajectories, for example. Concerning the "Interact" state recognition, various descriptors based on local features have been recently tested for action recognition in movies and could be tested for our application. Detecting and matching feature points in a local temporal context and describing their displacements can be an interesting descriptor, for example.

More complex scenarios could be described, such as counting the number of products picked up, or detecting people interacting together.

Conclusions et perspectives

Cette thèse présente notre travail réalisé dans plusieurs champs de la vision par ordinateur. Le but est de créer un système qui analyse les comportements humains dans un point de vente en temps réel. Ce système est composé de divers modules sur trois niveaux.

La couche de bas niveau est composée de la détection de mouvement et du suivi d'objet, présentés dans le chapitre 2 et 3. La couche de niveau moyen analyse les activités humaines, voir chapitre 4. Finalement, la couche de haut-niveau est composée de deux procédés séparés: l'analyse sémantique et la reconnaissance de scénarios, voir chapitre 5.

Détection de mouvement

Le chapitre 2 présente et évalue plusieurs méthodes de détection de mouvement.

D'abord, nous présentons la différence inter-image, une technique sans modèle de l'arrière-plan, qui est très rapide. Cependant, les résultats sont de mauvaise qualité.

Puis, la moyenne temporelle, qui utilise un modèle de l'arrière-plan, offre de biens meilleurs résultats tout en restant rapide. Cependant, les mouvements de caméras et les changements de luminosité limitent cette méthode.

Afin de résoudre ces problèmes, nous testons des approches utilisant des modèles multi-modaux de l'arrière-plan, tels que la classification par règles de décision Bayesienne (BDR) [80], le modèle de mélange de mixture Gaussienne [136], et notre méthode: le modèle de mixture Gaussienne amélioré (iGMM).

iGMM et BDR offrent des résultats très précis et sont robustes aux mouvements de caméras, changements de luminosité, ombres, et objets arrêtés dans la scène. Cependant, iGMM est environ 5 fois plus rapide que BDR.

La détection de mouvement a été largement étudiée depuis plus d'une dizaine d'année et beaucoup de méthodes ont été proposées. Cependant, il existe toujours des méthodes d'analyse de données pouvant être adaptées à la détection de mouvement. Par exemple, des modèles globaux de l'arrière-plan sont créés avec des analyses en composante principale [149].

Suivi d'objets

Le chapitre 3 présente et compare plusieurs approches pour le suivi d'objets.

D'abord, nous présentons quatre algorithmes existants basés sur les composants connectés, mean-shift, les filtres à particules, et les composants connectés combinés avec des filtres à particules.

Puis, nous proposons un sytème basé sur plusieurs hypothèses, qui suit des objets sur plusieurs niveaux d'analyse.

Après avoir évalué les cinq méthodes, nous concluons que notre système peut suivre des objets rigides et non-rigides, suivre des objets dans des scènes extérieures et complexes et peut gérer divers types d'occultations.

Etant donné que notre méthode est basée sur la détection de mouvement présentée dans le chapitre 2, les avantages et limites du suivi sont liés à la technique de détection. L'avantage est que la détection offre des résultats très précis pour le contour des objets et la limite générale est que les objets s'occultant les uns les autres sont fusionnés et sont indistinguables.

Tout comme la détection de mouvement, le suivi d'objet a été largement étudié. La plupart des systèmes sont adaptés à des cas spécifiques et peu d'entre eux peuvent faire face à des types de scènes très différent. Combiner des méthodes ensemble peut être intéressant pour améliorer la généricité et la précision.

Analyse d'activité humaine

Le chapitre 4 présente la phase d'analyse de comportement du système. D'abord, un modèle est défini, basé sur six états. Ces six états sont organisés dans une machine à état, qui modélise le comportement de chaque objet suivi dans la scène.

Puis, nous nous focalisons sur l'état "Interact", qui détecte les interactions entre les clients et les produits du point de vente. Cet état correspond à un évènement représentant la prise d'un produit et peut être détecté de diverses manières. D'abord, nous détectons l'interaction de manière déterministe. Puis, nous créons plusieurs descripteurs: Image d'historique du mouvement, Image de mouvement accumulé, contexte de mouvement local (LMC), contexte d'interaction (IC) et la combinaison du contexte d'interaction et du contexte de mouvement local. Ces descripteurs sont utilisés pour détecter l'état "Interact" avec des machines à vecteurs de support.

L'évaluation montre que la méthode déterministe offre de bons résultats pour un système sans phase d'apprentissage. Puis, la méthode probabiliste offre les meilleurs résultats avec la combinaison des descripteurs LMC et IC. On remarque, que l'on utilise une phase d'apprentissage très courte pour obtenir ces bons résultats.

L'analyse de comportement humain est un champ de recherche plus jeune et offre encore beaucoup de problèmes à résoudre. Il existe notamment des challenges ayant pour but de détecter des actions de plus en plus complexes. Par exemple, la conférence CVPR 2011 [3] propose un challenge dont le but est de reconnaitre des comportements tels que "personne entrant/sortant d'un véhicule", "personne chargeant/déchargeant un véhicule", etc. D'autre chalenge existent pour détecter des actions dans des films [133] [78].

Interprétation et reconnaissance de scénario

Le dernier chapitre présente deux procédés de haut niveau. D'abord, une interprétation sémantique est générée décrivant les actions réalisées par les acteurs dans la vidéo. Puis trois scenarios résumant les actions de chaque acteur sont reconnus. Ces deux traitements sont basés sur le module d'analyse de comportement qui détecte l'état d'action de chaque personne en chaque instant.

L'interprétation sémantique est basée sur ces états et génère des phrases lorsqu'un changement d'état est noté. Cette interprétation produit des informations sémantiques utilisables par des non-spécialistes. Les résultats dépendent directement de la phase de détection des états.

La reconnaissance de scenario génère un résumé de la vidéo en catégorisant le comportement de chaque personne détectée. La reconnaissance de scénario offre de très bons résultats tant que les objets sont correctement identifiés. Cependant, certains objets peuvent êtres mal identifiés lorsque l'on atteint les limites du système de suivi.

Ces deux procédés de haut niveau sont fortement liés à l'analyse de comportement. De nos jours, l'analyse sémantique est étudiée dans divers contextes: l'analyse de scène, la récupération d'image, etc.

Perspectives du projet

En ce qui concerne le projet, il y a plusieurs axes d'amélioration.

Une première étape consisterait à faire des tests à plus grande échelle: faire marcher le système pendant plusieurs jours consécutifs dans plusieurs endroits. Un tel prototype nécessite de prendre plusieurs décisions concernant le contenu à diffuser sur les écrans. Plusieurs scénarios sont possibles: diffuser une vidéo liée à un produit lorsqu'une personne interagit avec ce produit, diffuser plusieurs vidéos et analyser leur impact sur l'interaction avec les produits. De tels tests montreraient probablement des nouvelles difficultés et des axes d'améliorations.

Des améliorations sont possibles sur la détection de mouvement en parallélisant le procédé de détection de mouvement.

Le suivi d'objets peut être testé en modifiant les descripteurs. On peut utiliser un filtre de Kalman pour prédire la position suivante du centre de gravité de chaque objet, par exemple. Un histogramme de couleur peut être aussi utilisé comme descripteur.

D'autres comportements pourraient être reconnus. Des méthodes analysant la trajectoire des objets peuvent être implémentées. Concernant l'état "Interact", des descripteurs basés sur des "local features" ont été récemment testés pour détecter des actions dans des films et pourraient offrir de bons résultats pour notre application.

Des scenarios plus complexes pourraient être décrit afin d'estimer le nombre de produit pris ou détecter les personnes interagissant ensemble.

Bibliography

[1] *VAST Symposium Challenge*, 2009. 89

[2] *PETS: Performance Evaluation of Tracking and Surveillance*, 2010. 19, 23

[3] *CVPR Workshop: Activity Recognition Competition*, 2011. 149, 153

[4] J. K. Aggarwal and Q. Cai. Human motion analysis: a review. pages 90–102, 1997. 19, 24

[5] J.F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983. 135

[6] J.F. Allen. Towards a general theory of action and time. *Artificial intelligence*, 23(2):123–154, 1984. 15, 134, 135

[7] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002. 76

[8] JL Barron, D.J. Fleet, and SS Beauchemin. Performance of optical flow techniques. *International journal of computer vision*, 12(1):43–77, 1994. 30

[9] A. Baumberg and D. Hogg. Learning deformable models for tracking the human body. *Motion-Based Recognition*, 3:39–60. 72

[10] JE Bennett and JC Wakefield. Markov chain Monte Carlo for nonlinear hierarchical models. *Markov Chain Monte Carlo in Practice (London: Chapman & Hall)*, pages 339–357, 1996. 74

[11] D. Beymer and K. Konolige. Real-time tracking of multiple people using continuous detection. In *IEEE Frame Rate Workshop*. Citeseer, 1999. 72

[12] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. 2005. 18, 23, 109

[13] A.F. Bobick and J.W. Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(3):257–267, 2002. 106, 109, 113

[14] AF Bobick and AD Wilson. A state-based technique to the representation and recognition of gesture. *IEEE Trans. Pattern Anal. Machine Intell*, 19:1325–1337, 1997. 106

[15] TE Boult, 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 *Visual Surveillance, 1999. Second IEEE Workshop on,(VS'99)*, pages 48–55. IEEE, 2002. 31

[16] M. Brand. Understanding manipulation in video. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 94–99. IEEE, 2002. 107

[17] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *cvpr*, page 994. Published by the IEEE Computer Society, 1997. 106

[18] C. Bregler. Learning and recognizing human dynamics in video sequences. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 568–574. IEEE, 2002. 73

[19] F. Bremond and G. Medioni. Scenario recognition in airborne video imagery. In *DARPA Image Understanding Workshop 1998*, pages 211–216. Citeseer, 1998. 106, 117

[20] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998. 109

[21] Hilary Buxton. Learning and understanding dynamic scene activity: a review. *Image and Vision Computing*, 21(1):125 – 136, 2003. 19, 24

[22] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. 2001. 90, 116, 117

[23] J.C. Cheng and J.M.F. Moura. Capture and representation of human walking in live video sequences. *Multimedia, IEEE Transactions on*, 1(2):144–156, 2002. 73

[24] S.C.S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. 2003. 19, 24

[25] Cliris. Cliris. 18, 23

[26] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, 6(4):271–288, 1998. 72

[27] D. Comaniciu and V. Ramesh. Mean shift and optimal prediction for efficient object tracking. In *2000 International Conference on Image Processing, 2000. Proceedings*, pages 70–73, 2000. 75

[28] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1337–1342, 2003. 30

[29] R. Cutler and L. Davis. View-based detection and analysis of periodic motion. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, volume 1, pages 495–500. IEEE, 2002. 30

[30] Q. Delamarre and O. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999*, pages 716–721, 1999. 73

[31] Q. Delamarre and O. Faugeras. 3D Articulated Models and Multiview Tracking with Physical Forces* 1. *Computer Vision and Image Understanding*, 81(3):328–357, 2001. 73, 74

[32] P.M. Djuric, J.H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M.F. Bugallo, and J. Miguez. Particle filtering. *Signal Processing Magazine, IEEE*, 20(5):19–38, 2003. 76

[33] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001. 77

[34] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. 2003. 110, 113, 114

[35] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. *Computer Vision—ECCV 2000*, pages 751–767, 2000. 32

[36] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. pages 1–8, 2008. 110

[37] C.J. Fillmore. THE CASE FOR CASE. 1967. 131

[38] Y. Freund, R. Schapire, and N. Abe. A short introduction to boosting. *JOURNAL-JAPANESE SOCIETY FOR ARTIFICIAL INTELLIGENCE*, 14:771–780, 1999. 110

[39] A. Galata, N. Johnson, and D. Hogg. Learning variable-length Markov models of behavior. *Computer Vision and Image Understanding*, 81(3):398–413, 2001. 72

[40] W.F. Gardner and D.T. Lawton. Interactive model-based vehicle tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(11):1115–1121, 2002. 74

[41] D. M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82 – 98, 1999. 19, 24

[42] NJ Gordon, DJ Salmond, and AFM Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings F*, volume 140, pages 107–113. IET, 2002. 76

[43] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(12):2247–2253, 2007. 109

[44] M. Haag and H.H. Nagel. Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences. *International Journal of Computer Vision*, 35(3):295–319, 1999. 74

[45] B. Han, D. Comaniciu, and L. Davis. Sequential kernel density approximation through mode propagation: applications to background modeling. In *Proc. ACCV*, volume 2004. Citeseer, 2004. 31

[46] I. Haritaoglu, D. Harwood, and L.S. Davis. real-time surveillance of people and their activities. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):809–830, 2002. 109

[47] MH Hayes. Statistical digital signal processing and modeling. 1996, 1996. 31

[48] M. Heikkila and M. Pietikainen. A texture-based method for modeling the background and detecting moving objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(4):657–662, 2006. 31

[49] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999. 109

[50] L. Hongche, T. Hong, M. Herman, and R. Chellappa. Accuracy vs. efficiency trade-offs in optical flow algorithms. *Computer Vision and Image Understanding*, 72(3):271–86, 1998. 30

[51] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 30

[52] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE ICCV*, volume 99, pages 1–19. Citeseer, 1999. 53

[53] Weiming Hu, Tieniu Tan, Liang Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, 34(3):334–352, 2004. 19, 24, 71, 109

[54] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, and T.S. Huang. Action detection in complex scenes with spatial and temporal ambiguities. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 128–135. IEEE, 2010. 19, 24, 110

[55] T. Huang, D. Koller, J. Malik, G. Ogasawara, B. Rao, S. Russell, and J. Weber. Automatic symbolic traffic scene analysis using belief networks. In *Proceedings of the National Conference on Artificial Intelligence*, pages 966–966. JOHN WILEY & SONS LTD, 1995. 130

[56] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *Computer Vision—ECCV'96*, pages 343–356, 1996. 72

[57] M. Isard and A. Blake. Condensation—conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998. 72, 74

[58] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. *Computer Vision—ECCV'98*, page 893, 1998. 74

[59] YA Ivanov and AF Bobick. Recognition of visual activities and interactions by stochastic parsing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):852–872, 2002. 107

[60] R. Jain and H.H. Nagel. On the analysis of accumulative difference pictures from image sequences of real world scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):206–214, 2009. 29

[61] D.S. Jang and H.I. Choi. Active models for tracking moving objects. *Pattern Recognition*, 33(7):1135–1146, 2000. 72

[62] G. Jing, C.E. Siong, and D. Rajan. Foreground motion detection by difference-based spatial temporal entropy image. In *TENCON 2004. 2004 IEEE Region 10 Conference*, pages 379–382. IEEE, 2005. 29

[63] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14(8):609–615, 1996. 108

[64] S.X. Ju, M.J. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated image motion. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, pages 38–44. IEEE, 2002. 73

[65] M.Y. Jun'ichi Hoshino and H. Saito. A match moving technique for merging CG cloth and human movie sequences. *The Journal of Visualization and Computer Animation*, 12(1):23–29, 2001. 74

[66] P. KaewTraKulPong and R. Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Proc. European Workshop Advanced Video Based Surveillance Systems*, volume 1. Citeseer, 2001. 51, 53

[67] IA Karaulova, PM Hall, and A.D. Marshall. A hierarchical model of dynamics for tracking people with a single video camera. In *British Machine Vision Conference*, volume 1, pages 352–361. Citeseer, 2000. 73

[68] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. 2005. 110

[69] A. Klaser. *Apprentissage pour la reconnaissance d'actions humaines en vidéo*. PhD thesis, 2010. 19, 24

[70] A. Kojima, M. Izumi, T. Tamura, and K. Fukunaga. Generating natural language description of human behavior from video images. In *icpr*, page 4728. Published by the IEEE Computer Society, 2000. 130

[71] A. Kojima, T. Tamura, and K. Fukunaga. Natural language description of human activities from video images based on concept hierarchy of actions. *International Journal of Computer Vision*, 50(2):171–184, 2002. 130, 132

[72] D. Koller, K. Daniilidis, and H.H. Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *International Journal of Computer Vision*, 10(3):257–281, 1993. 31, 74

[73] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell. Towards robust automatic traffic scene analysis in real-time. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pages 126–131. IEEE, 2002. 72

[74] H. Kollnig, H. Nagel, and M. Otte. Association of motion verbs with vehicle movements extracted from dense optical flow fields. *Computer Vision—ECCV'94*, pages 338–347, 1994. 130

[75] H. Kollnig and H.H. Nagel. 3D pose estimation by directly matching polyhedral models to gray value gradients. *International Journal of Computer Vision*, 23(3):283–302, 1997. 74

[76] T. Korhonen, P. Pertil, and A. Visa. Particle filtering in high clutter environment. In *Proceedings of the 2005 Finnish Signal Processing Symposium. FINSIG*, 2005. 76

[77] V. Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI magazine*, 13(1):32, 1992. 135

[78] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 19, 23, 149, 153

[79] I. Laptev and P. Pérez. Retrieving actions in movies. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007. 110

[80] L. Li, W. Huang, I.Y.H. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *Proceedings of the eleventh ACM international conference on Multimedia*, page 10. ACM, 2003. 40, 41, 59, 147, 151

[81] Z.Q. Liu, L.T. Bruton, J.C. Bezdek, J.M. Keller, S. Dance, N.R. Bartley, and C. Zhang. Dynamic image sequence analysis using fuzzy measures. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 31(4):557–572, 2002. 130

[82] J. Lou, Q. Liu, T. Tan, and W. Hu. Semantic interpretation of object activities in a surveillance system. In *International Conference on Pattern Recognition*, volume 16, pages 777–780. Citeseer, 2002. 129

[83] J. Lou, H. Yang, WM Hu, and T. Tan. Visual vehicle tracking using an improved EKF. In *Proc. Asian Conf. Computer Vision*, pages 296–301. Citeseer, 2002. 74

[84] D.G. Lowe. Fitting parameterized three-dimensional models to images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(5):441–450, 2002. 74

[85] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 109

[86] W.L. Lu and J.J. Little. Simultaneous tracking and action recognition using the pca-hog descriptor. In *Computer and Robot Vision, 2006. The 3rd Canadian Conference on*, page 6. IEEE, 2006. 110

[87] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International joint conference on artificial intelligence*, volume 3, pages 674–679. Citeseer, 1981. 30, 114

[88] Y.F. Ma and H.J. Zhang. Detecting motion object by spatio-temporal entropy. 2001. 29

[89] DJC MacKay. Introduction to monte carlo methods. *Learning in graphical models*, pages 175–204, 1998. 72

[90] J. Malik and S. Russell. A machine vision based surveillance system for California roads. 1995. 71

[91] J. Malik and S. Russell. Traffic Surveillance and Detection Technology Development: New Traffic Sensor Technology. *Info: Research Reports, California Partners for Advanced Transit and Highways (PATH), Institute of Transportation Studies, UC Berkeley*, 1997. 71, 72

[92] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004. 81

[93] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42–56, 2000. 30, 71

[94] U. Meier, R. Stiefelhagen, J. Yang, and A. Waibel. Towards unrestricted lip reading. *International Journal of Pattern Recognition and Artificial Intelligence*, 14(5):571–586, 2000. 107

[95] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231 – 268, 2001. 19, 24

[96] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90 – 126, 2006. Special Issue on Modeling People: Vision-based understanding of a person's shape, appearance, movement and behaviour. 19, 24

[97] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 786–793. IEEE, 2002. 33

[98] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349–361, 2002. 72

[99] M. Mohnhaupt and B. Neumann. On the use of motion concepts for top-down control in traffic scenes. *Computer Vision—ECCV 90*, pages 598–600, 1990. 130

[100] B.T. Morris and M.M. Trivedi. A survey of vision-based trajectory learning and analysis for surveillance. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(8):1114–1127, 2008. 19, 24, 88

[101] S.A. Niyogi and E.H. Adelson. Analyzing and recognizing walking figures in XYT. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 469–474. IEEE, 2002. 73

[102] N.M. Oliver, B. Rosario, and A.P. Pentland. A Bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2002. 33, 107

[103] E.J. Ong and S. Gong. A dynamic 3D human model using hybrid 2D-3D representations in hierarchical pca space. In *BMVC*, 1999. 73

[104] J. Owens and A. Hunter. Application of the self-organizing map to trajectory classification. *vs*, page 77, 2000. 108

[105] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(3):266–280, 2002. 72

[106] CA Pau and A. Barber. Traffic sensor using a color vision method. *Proc. SPIE—Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS*, 2902:156–165, 1996. 72

[107] A. Pece and A. Worrall. A statistically-based Newton method for pose refinement. *Image and Vision Computing*, 16(8):541–544, 1998. 74

[108] A.E.C. Pece and A.D. Worrall. Tracking without feature detection. In *Proc. IEEE Int. Workshop Performance Evaluation of Tracking and Surveillance*, pages 29–37. Citeseer. 74

[109] N. Peterfreund. Robust tracking of position and velocity with Kalman snakes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(6):564–569, 2002. 72

[110] M. Piccardi. Background subtraction techniques: a review. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 4, pages 3099–3104. Ieee, 2005. 19, 24

[111] CS Pinhanez and AF Bobick. Human action detection using pnf propagation of temporal constraints. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 898–904. IEEE, 2002. 135

[112] R. Plankers and P. Fua. Articulated soft objects for video-based body modeling. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 394–401. IEEE, 2002. 73

[113] R. Pless. Spatio-temporal background models for outdoor surveillance. *EURASIP Journal on Applied Signal Processing*, 2005:2281–2291, 2005. 19, 24

[114] R. Polana and R. Nelson. Low level recognition of human motion. In *Proc. IEEE Workshop on Nonrigid and Articulate Motion*, pages 77–82. Citeseer, 1994. 72, 108, 109

[115] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010. 19, 24

[116] Quividi. Quividi. 18, 23

[117] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 31

[118] E. Ramasso, C. Panagiotakis, M. Rombaut, D. Pellerin, and G. Tziritas. Human shape-motion analysis in athletics videos for coarse to fine action/activity recognition using transferable belief model. *Electronic Letters on Computer Vision and Image Analysis*, 7(4):32–50, 2009. 109

[119] P. Remagnino, T. Tan, and K. Baker. Multi-agent visual surveillance of dynamic scenes. *Image and Vision Computing*, 16(8):529–532, 1998. 130

[120] P. Remagnino, T. Tan, and K. Baker. Agent orientated annotation in model based visual surveillance. In *Computer Vision, 1998. Sixth International Conference on*, pages 857–862. IEEE, 2002. 130

[121] C. Ridder, O. Munkelt, and H. Kirchner. Adaptive background estimation and foreground detection using kalman-filtering. In *Proceedings of International Conference on recent Advances in Mechatronics*, pages 193–199. Citeseer, 1995. 31

[122] J. Rittscher, J. Kato, S. Joga, and A. Blake. A probabilistic background model for tracking. *Computer Vision—ECCV 2000*, pages 336–350, 2000. 31

[123] M.D. Rodriguez, J. Ahmed, and M. Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 19, 24, 110

[124] K. Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP-Image Understanding*, 59(1):94–115, 1994. 73

[125] R. Rosales and S. Sclaroff. 3D trajectory recovery for tracking multiple objects and trajectory guided recognition of actions. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 2002. 72

[126] P. Rosin. Thresholding for change detection. In *Computer Vision, 1998. Sixth International Conference on*, pages 274–279. IEEE, 2002. 43

[127] B. Schiele. Model-free tracking of cars and people based on color regions. *Image and Vision Computing*, 24(11):1172–1178, 2006. 72

[128] K. Schindler and L. Van Gool. Action Snippets: How many frames does human action recognition require? In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 110

[129] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004. 18, 19, 23, 24

[130] Andrew Senior. An introduction to automatic video surveillance. In Andrew Senior, editor, *Protecting Privacy in Video Surveillance*, pages 1–9. Springer London, 2009. 10.1007/978-1-84882-301-3_1.109

[131] K. Shafique and M. Shah. A non-iterative greedy algorithm for multi-frame point correspondence. *IEEE Trans. Pattern Anal. Mach. Intell*, 2005. 72

[132] H. Sidenbladh, M. Black, and D. Fleet. Stochastic tracking of 3D human figures using 2D image motion. *Computer Vision—ECCV 2000*, pages 702–718, 2000. 73

[133] A.F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and TRECVid. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 321–330. ACM, 2006. 19, 23, 149, 153

[134] C. Sminchisescu and B. Triggs. Covariance scaled sampling for monocular 3D body tracking. 2001. 73

[135] T. Starner, J. Weaver, and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(12):1371–1375, 2002. 107

[136] C. Stauffer and WEL Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 2002. 31, 48, 49, 51, 53, 55, 59, 60, 147, 151

[137] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and JM Buhmann. Topology free hidden markov models: Application to background modeling. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 294–301. IEEE, 2002. 32

[138] N. Sumpter and A. Bulpitt. Learning spatio-temporal patterns for predicting object behaviour. *Image and Vision Computing*, 18(9):697–704, 2000. 108

[139] K. Takahashi, S. Seki, E. Kojima, and R. Oka. Recognition of dexterous manipulations from time-varying images. In *Motion of Non-Rigid and Articulated Objects, 1994., Proceedings of the 1994 IEEE Workshop on*, pages 23–28. IEEE, 2002. 106

[140] T. Tan, G. Sullivan, and K. Baker. Pose determination and recognition of vehicles in traffic scenes. *Computer Vision—ECCV'94*, pages 501–506, 1994. 74

[141] T.N. Tan and K.D. Baker. Efficient image gradient based vehicle localization. *Image Processing, IEEE Transactions on*, 9(8):1343–1356, 2002. 74

[142] TN Tan, GD Sullivan, and KD Baker. Fast vehicle localisation and recognition without line extraction and matching. In *Proc. 5th British Machine Vision Conference*, pages 85–94, 1994. 74

[143] TN Tan, GD Sullivan, and KD Baker. Model-based localisation and recognition of road vehicles. *International Journal of Computer Vision*, 27(1):5–25, 1998. 74

[144] Y.L. Tian and A. Hampapur. Robust salient motion detection with complex background for real-time video surveillance. In *Motion and Video Computing, 2005. WACV/MOTIONS'05 Volume 2. IEEE Workshop on*, volume 2, pages 30–35. IEEE, 2007. 32

[145] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *iccv*, page 255. Published by the IEEE Computer Society, 1999. 19, 24, 31, 32

[146] D. Tran and A. Sorokin. Human activity recognition with metric learning. *Computer Vision–ECCV 2008*, pages 548–561, 2008. 113, 114

[147] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine Recognition of Human Activities: A Survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, September 2008. 19, 24

[148] C.J. Veenman, M.J.T. Reinders, and E. Backer. Resolving motion correspondence for densely moving points. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(1):54–72, 2002. 72

[149] N. Verbeke. *Sujet de these: Suivi d'objets en mouvement dans une séquence vidéo*. PhD thesis, 2007. 19, 24, 28, 147, 151

[150] S. Wachter and H.H. Nagel. Tracking of persons in monocular image sequences. In *Nonrigid and Articulated Motion Workshop, 1997. Proceedings., IEEE*, pages 2–9. IEEE, 2002. 73, 74

[151] Toshikazu Wada and Takashi Matsuyama. Multiobject behavior recognition by event driven selective attention method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:873–887, 2000. 107

[152] F. Wagner, R. Schmuki, and T. Wagner. *Modeling software with finite state machines: a practical approach*. Auerbach Publications, 2006. 117

[153] L. Wang, W. Hu, and T. Tan. Recent developments in human motion analysis. *Pattern recognition*, 36(3):585–601, 2003. 19, 24

[154] D. Weinland and E. Boyer. Action recognition using exemplar-based embedding. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–7. IEEE, 2008. 109

[155] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *Computer Vision and Image Understanding*, 104(2-3):249–257, 2006. 18, 23

[156] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. Technical report, New York, NY, USA, February 2011. 19, 24

[157] A.D. Wilson, A.F. Bobick, and J. Cassell. Temporal classification of natural gesture and application to video coding. In *cvpr*, page 948. Published by the IEEE Computer Society, 1997. 106

[158] K. Wonjun, L. Jaeho, K. Minjin, O. Daeyoung, and K. Changick. Human action recognition using ordinal measure of accumulated motion. *EURASIP Journal on Advances in Signal Processing*, 2010, 2010. 109, 113

[159] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785, 2002. 30, 31, 71

[160] Y. Wu and T.S. Huang. A co-inference approach to robust visual tracking. 2001. 72

[161] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 379–385. IEEE, 2002. 108

[162] H. Yang, J. Lou, H. Sun, W. Hu, and T. Tan. Efficient and robust vehicle localization. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 2, pages 355–358. IEEE, 2002. 74

[163] M.H. Yang and N. Ahuja. Extraction and classification of visual motion patterns for hand gesture recognition. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 892–897. IEEE, 2002. 107

[164] T. Yang, S.Z. Li, Q. Pan, and J. Li. Real-time and accurate segmentation of moving objects in dynamic scene. In *Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*, pages 136–143. ACM, 2004. 30

[165] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm Computing Surveys (CSUR)*, 38(4):13, 2006. 19, 24

[166] A. Yilmaz and M. Shah. Actions sketch: A novel action representation. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 984–989. IEEE, 2005. 109

[167] Z. Zhang, Y. Hu, S. Chan, and L.T. Chia. Motion context: A new representation for human action recognition. *Computer Vision–ECCV 2008*, pages 817–829, 2008. 109

[168] T. Zhao, T. Wang, and H.Y. Shum. Learning a highly structured motion model for 3d human tracking. In *Proc. Asian Conf. Computer Vision*, volume 1, pages 144–149. Citeseer, 2002. 73

[169] Y. Zhou, H. Nicolas, and J. Benois-Pineau. A multi-resolution particle filter tracking in a multi-camera environment. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 4065–4068. IEEE, 2010. 72

[170] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006. 51, 52