UAB
Universitat Autònoma
de Barcelona

Many-to-Many High Order Matching. Applications to Tracking and Object Segmentation

A dissertation submitted by **Jose C. Rubio Ballester** at Universitat Autonoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, July 2012

Director

Dr. Joan Serrat
Dept. Ciencies de la Computació & Centre de Visió per Computador
Universitat Autonoma de Barcelona

This document was typeset by the author using $\LaTeX\,2_{\varepsilon}$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autonoma de Barcelona.

ISBN:

Printed by

Als meus pares, Jose i Leonor,
a la meva germana Belén,
i a Naila

Aknowledgements

Completing a PhD has probably been the most challenging and rewarding activity of my life. I have been fortunate to be surrounded by fantastic people that gave me their support in the best and worse moments of this long journey. I believe that without their encouragement and advise this dissertation could not have been possible. It has been a great privilege to spend these four years in the Computer Vision Center at Universitat Autónoma de Barcelona, and its members will always remain dear to me.

I am truly indebted and thankful to Joan Serrat for being an excellent mentor and supervisor. I am grateful not only for his advice and contributions to this work, but also for his patience and perseverance, that has enhanced my research career significantly. I also would like to thank Antonio López for being and excellent group director, and for his counsel and constructive ideas.

It is difficult to overstate my gratitude to professor Nikos Paragios, for granting me the opportunity to collaborate and share an office in the MAS Laboratorie in École Centrale Paris. I spent five wonderful months under his tutorship, and his inspiring advice was extremely helpful. I would like to thank all the members of the MAS Laboratorie, and particularly to Aris Sotiras and Olivier Teboul for their valuable counsel and entertaining conversations, which made even more pleasant my short stage in Paris.

The members of the ADAS group also deserve my sincerest thanks for their assistance and friendship. Especially, I would like to thank my colleagues and friends David Vazquez and Javier Marin for their sense of humor, and for their encouragement in times of joy and sorrow. The fellow students and members of the Computer Vision Center in Barcelona have greatly contributed to my experience as a PhD student. I am particularly grateful to my friends Pep Gonfaus, Jaume Gibert and Albert Gordo, which have always given me their support, as well as sharing my preoccupations and interests throughout all these years.

I have no words to express how happy I am to have met Naila Murray in the first stages of my PhD. Her company and affection, her constant support and encouragement has greatly contributed to the success of this work, as well as my personal maturation and progression in life.

Finally, I would like to thank my family and friends from Valencia and Barcelona, which have patiently bore my complaints for four years. I am deeply grateful to my parents Jose and Leonor, as well as my sister Belén. Your love and support made possible the completion of this PhD.

Resum

La correspondencia de caracter´stiques és un problema fonamental de la Visió per Computador, que té múltiples aplicacions com el seguiment, la classificació i recuperació d'imatges, el reconeixement de formes i la visió estereoscopica. En molts ambits, és útil per representer l'estructura local de les caracter´stiques en correspondencia, per augmentar la precissió o per fer les correspondencies invariants a certes transformacions (afins, homografies, etc...). No obstant aixo, la codificació d'aquest coneixement requereix complicar el model mitjançant l'establiment de relacions d'ordre alt entre els elements del model, i per tant l'augment de la complexitat del problema d'optimització.

La importancia de les correspondencies molts-a-molts es de vegades ignorada en la literatura. La majoria dels metodes es limiten a realizar correspondencies un-a-un, generalment validant en conjunts de dades sintetiques, o no realistes. En un entorn real, amb variacions d'escala, il.luminació i orientació de l'objecte d'interés, i amb la presencia d'oclusions, desordre, i observacions sorolloses, les relacions molts-a-molts son necessaries per aconseguir resultats satisfactoris. Com a conseqüencia, trovar la correspondencia molts-a-molts més probable, implica un procés complicat d'optimització combinatoria.

En aquest treball dissenyem i demostrem algorismes de correspondencia que calculen associacions molts-a-molts, i que poden ser aplicats a diversos problemes dif´cils de resoldre. El nostre objectiu és fer ús de representacios d'ordre alt per millorar el poder expressiu de la correspondencia, alhora que ferm possible el procés d'inferencia o l'optimització d'aquests models. Al llarg de la tesi, hem utilitzat eficaçment els models grafics com la nostra representació preferida, ja que proporcionen un marc probabil´stic elegant per abordar problemes de predicció estructurada.

Hem introdüit un algorisme de seguiment bassat en correspondencies que es porten a terme entre els fotogrames d'una sequencia de v´deo, per tal de resoldre el problema de segument de fars de cotxes durant la nit. També generalitzem aquest mateix algorisme per resoldre el problema de l'associació de dades aplicat a different escenaris de seguiment. Hem demostrat l'eficacia d'aquest enfoc en seqüencies de v´deo reals i demostrem que el nostre algorisme de seguiment es pot utilitzar per millorar la precisió d'un sistema de classificació de fars de cotxes.

A la segona part d'aquest treball, pasem desde correspondencies no denses (punts) cap a correspondencies denses (regions), i introdüim una nova representació jerarquica d'imatges. Seguidament, fem ús d'aquest model per desenvolupar correspondencies molts-a-molts d'ordre alt entre parelles d'imatges. Demostrem que l'ús de models d'ordre alt en comparació amb altres models mes senzills no només millora l'exactitud dels resultats, sinó també la velocitat de convergencia de l'algorisme d'inferencia.

Finalment, seguim explotant la idea de correspondencia de regions per dissenyar un algorisme de co-segmentació completament no supervisat, que és capaç de competir amb altres metodes supervisats de l'estat-de-l'art. El nostre metode supera inconvenients t´pics d'alguns treballs passats, com evitar la necesitat d'aparences variades al fons de les imatges. La correspondencia de regions en aquest cas s'aplica per explotar eficaçment la informació compartida entre les imatges. També extenem aquest treball per dur a terme co-segmentació de v´deos, sent la primera vegada que s'aborda aquest problema.

Abstract

Feature matching is a fundamental problem in Computer Vision, having multiple applications such as tracking, image classification and retrieval, shape recognition and stereo fusion. In numerous domains, it is useful to represent the local structure of the matching features to increase the matching accuracy or to make the correspondence invariant to certain transformations (affine, homography, etc...). However, encoding this knowledge requires complicating the model by establishing high-order relationships between the model elements, and therefore increasing the complexity of the optimization problem.

The importance of many-to-many matching is sometimes dismissed in the literature. Most methods are restricted to perform one-to-one matching, and are usually validated on synthetic, or non-realistic datasets. In a real challenging environment, with scale, pose and illumination variations of the object of interest, as well as the presence of occlusions, clutter, and noisy observations, many-to-many matching is necessary to achieve satisfactory results. As a consequence, finding the most likely many-to-many correspondence often involves a challenging combinatorial optimization process.

In this work, we design and demonstrate matching algorithms that compute many-to-many correspondences, applied to several challenging problems. Our goal is to make use of high-order representations to improve the expressive power of the matching, at the same time that we make feasible the process of inference or optimization of such models. We effectively use graphical models as our preferred representation because they provide an elegant probabilistic framework to tackle structured prediction problems.

We introduce a matching-based tracking algorithm which performs matching between frames of a video sequence in order to solve the difficult problem of headlight tracking at night-time. We also generalise this algorithm to solve the problem of data association applied to various tracking scenarios. We demonstrate the effectiveness of such approach in real video sequences and we show that our tracking algorithm can be used to improve the accuracy of a headlight classification system.

In the second part of this work, we move from *single* (point) matching to *dense* (region) matching and we introduce a new hierarchical image representation. We make use of such model to develop a high-order many-to-many matching between pairs of images. We show that the use of high-order models in comparison to simpler models improves not only the accuracy of the results, but also the convergence speed of the inference algorithm.

Finally, we keep exploiting the idea of region matching to design a fully unsupervised image co-segmentation algorithm that is able to perform competitively with state-of-the-art supervised methods. Our method also overcomes the typical drawbacks of some of the past works, such as avoiding the necessity of variate appearances on the image backgrounds. The region matching in this case is applied to effectively exploit inter-image information. We also extend this work to perform co-segmentation of videos, being the first time that such problem is addressed, as a way to perform video object segmentation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Matching in Computer Vision

Humans possess an outstanding ability to constantly analyze the visual information available in the environment they interact with. Several activities we carry out during our lives are strongly related to our visual capabilities: manipulating devices, interpreting gestures, recognizing objects, avoiding obstacles, etc. The amount of information to which humans are exposed is massive, both in terms of volume and complexity of the semantic content that can be inferred from the visual data. In Computer Vision research, we are interested in designing algorithms and models that allow a computer to autonomously analyze the available visual information in a similar manner as human beings do.

As humans, we observe the world as a sequence of images that capture photometric characteristics of the scene, as well as the movement of the objects. Moreover, we use a binocular vision system to extract depth information. The first process requires certain correlation of the visual stimuli over time to perceive the motion, and the second combines the information of each eye through a process called *binocular summation*. Although this is an over-simplistic explanation of how the visual information is acquired through the human vision system, it is interesting to note that these processes involve putting into correspondence information from several sources.

In this dissertation we address the problem of matching features from different information sources (e.g different images). This problem arises in multiple vision tasks. For instance, it has been applied to building recognition and retrieval systems, by establishing a similarity measure based on feature matching [77]. It has been extensively applied to tracking by establishing correspondences between the targets among several frames [27, 100], or to stereo vision to relate points from left and right images in order to calculate the disparity map [106]. Matching simply means finding a mapping, or a correspondence, from one set of features (points, regions, edges, etc...) to another set of features. Although the theoretical contribution of this work is clearly independent on the type of application considered, we focus on region correspondence applied to tracking, scene understanding and segmentation. In Computer Vision, point matching has been almost entirely approached as a graph matching problem since the 1970's [21], and in this work we also take this course. These pioneering studies present point sets abstracted as graphs were the nodes correspond to the points, while the edges model the spatial configuration of such points, imposing structural constraints that help disambiguate the correspondence problem. However, these structural constraints do not directly apply in the case of deformable objects, and the training and optimization of structural models is a difficult challenge. This discourages in some cases the use of structural matching in favour of appearance-only matching [3].

The problem of matching can be found in the literature under several names. Point Pattern Matching (PPM), or equivalently, Point Set Matching (PSM) generally refers to the problem of matching a

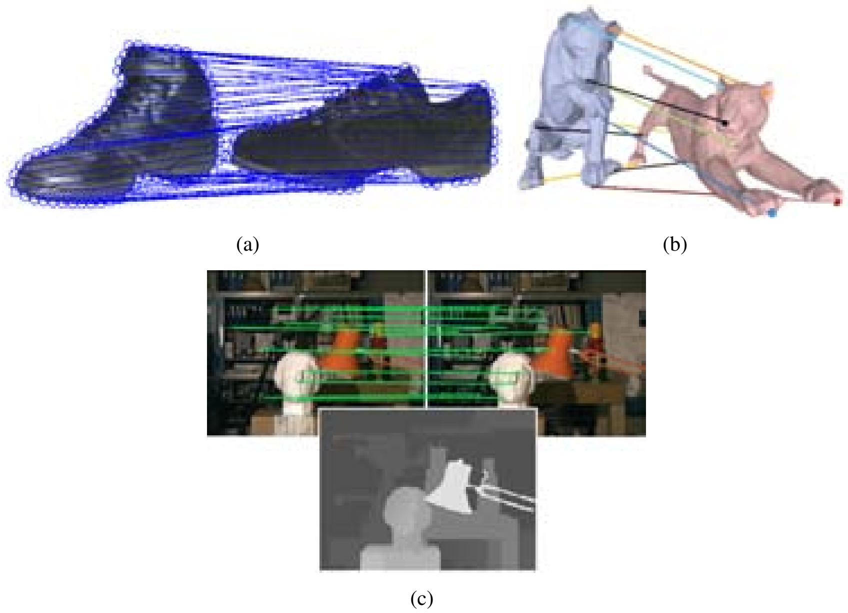


Figure 1.1: Examples of typical matching applications. In (a) the spectral method of [30] applied to silhouette matching. In (b), dense surface matching between 3D model meshes [109]. In (c), an example of matching for stereo-vision.

“query” set of points against a *database* of points or some subset of it, which has experimented some type of stochastic noise or transformation so that the original structure of the query has been significantly deformed. The classic terminology of PPM involved finding correspondences between query and database sets, where the points are represented exclusively by their location in the 2D-space. This is the case of contour matching and point registration. More recently, when applying matching to images, the sets of points to match usually have associated attributes that represent relevant features of the scene objects. The necessity for a representation which is robust against noise and variations of the node arrangements, has encouraged the researchers to use graph representations as well as properly choosing invariant features as attributes. These recent works do not strictly follow the query-database model, but just match sets of features, usually extracted from different images or object representations. Within this approach, typical applications include object retrieval, feature matching, stereo vision, or shape registration (See Figure 1.1).

1.2 Matching with Graphical Models

The core of this thesis is composed by three topics: posing the problem of feature matching as one of inference on graphical models, exploring the advantages and limitations of many-to-many matching over one-to-one correspondences, and using high-order graphs to model the scene using a rich and descriptive graph representation.

In the formulation we follow in the major part of this work, the matching problem is posed as a Maximum a Posteriori (MAP) inference on a Graphical Model. This means obtaining an estimate of certain unobserved quantity, such that it maximizes the posterior probability distribution representative of the problem. A detailed explanation of the MAP estimate can be found in Section 2.2.4. Using graphical models to solve this problem has several advantages:

- Provides an intuitive formalization of a probability distribution over many variables.
- Allows modeling the local structure present in spatial data.
- There exist efficient inference and optimization algorithms.
- It can handle fairly large-scale problems (thousands of variables).

The use of Graphical Models has been widely used in the past to represent the structure of a scene [91], and to perform matching between the elements within [14, 97]. Probabilistic Graphical Models have been very popular for a long time in the field of computer vision and pattern recognition at all levels. These techniques allow us to reason probabilistically about the values of several variables, given observations. There are three current avenues of research that are modernizing Graphical Models and specifically Markov Random Fields (MRFs). The first involves new inference algorithms and optimization techniques to perform approximate inference on MRFs (e.g improvements over Belief Propagation, Graph Cuts). These algorithms make the inference tractable and improve its usefulness by means of new optimization strategies [79,93] or by proposing new structures of the graph itself [14,68]. The second line of research involves improving the expressive power of MRFs with high-order models with greater expressive power at the expense of tractability of the inference algorithm. The third thread of research is concerned with exploiting training data to learn the potential functions as well as the optimal structure of the graph.

Many-to-many matching

Previous work on graph matching has typically focused on one-to-one assignments [53,97]. This means that one node in one graph can only be assigned to (at most) one node in the other graph. This can be a very restrictive assumption since it implies that both graphs agree in the level of abstraction, scale, and that the nodes cannot undergo splittings or merges from one graph to the other. For instance, consider the problem of finding correspondences between the pixels of two photographs of an object, that are taken at different resolutions. Each pixel of the low-resolution image will correspond to several pixels of the high-resolution image. As we will see along this dissertation, several applications and scenarios can only be correctly modeled following a many-to-many matching approach. One of the main difficulties of finding such correspondences is the fact that several subsets of nodes in one graph could be assigned to several subsets in the other graph, and this gives rise to a combinatorial challenge.

In the literature, the problem of matching has mainly been addressed in two modalities. The first is presented as a multi-dimensional labeling problem, where the variables represent source nodes and the labels represent the destination node indexes [14, 15]. The second approach uses binary variables to represent the possible node correspondences, labeling them as *active* or *inactive* [97, 109]. Figure 1.2 shows examples of both cases. In most of the matching formulation presented in this work we follow the second approach, since it better suits our necessities of many to many matching. Note that expressing one-to-many, or many-to-one correspondences using this representation is straightforward, simply by activating several variables/correspondences that share one of the points as source or destination (See Figure 1.2 (c)). The multi-label approach can express many-to-one correspondences, but since the variables can only have one label, expressing one-to-many matchings is not trivial. To do so, some authors perform forward-backward matching, or expand the variable dimensionality using values that enumerate sub-sets of destination points. However, these solutions are usually ad-hoc and not as attractive as using boolean variables.

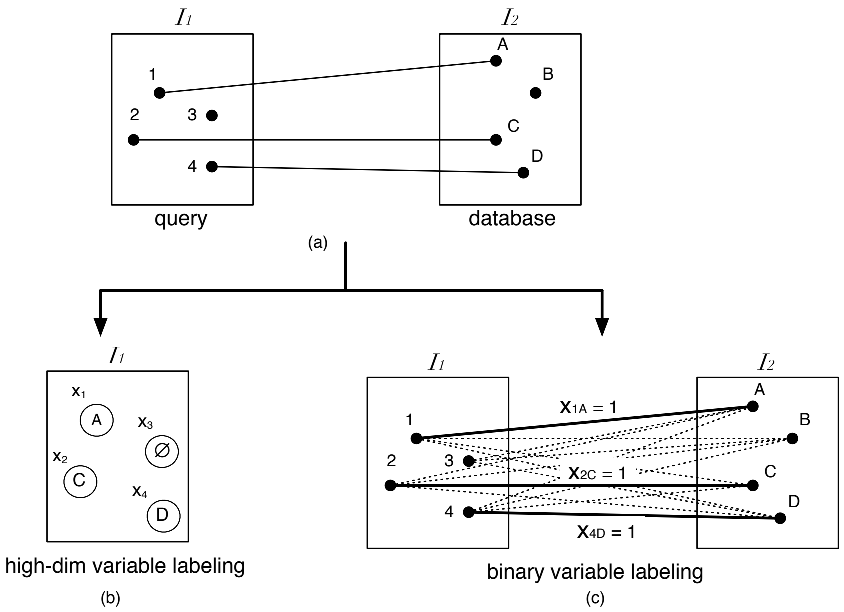


Figure 1.2: Toy examples of the two typical modalities of matching formulations. In (a), the problem of matching 4 source points (1-4) to 4 destination points (A-D) between two images I_1 and I_2. In (b) the multi-dimensional node labeling approach. In this case, only four variables corresponding to the points of image I_1 are present. Note how the variable labels correspond to the points of I_2, being ∅ in case of a non existing match. In (c), the binary approach, where each line represents one of the 16 variables. The bold lines represent active variables, while the dotted lines represent inactive variables.

High-order matching

Natural images often exhibit complex spatial structures (e.g a 3D scene), that cannot be captured using simple relationships between attributes of putative corresponding elements. In the context of graph matching, these relationships are often simplified to pairwise distances between neighboring elements, which provides a matching framework that relates pairs of graphs by an *isometry*, or a transformation that preserves distances. For example, if feature 1 is assigned to feature A and feature 2 is assigned to feature B, then the distance from 1 to 2 should be similar to the distance from A to B. However, a pairwise constraint is not enough to represent other kinds of invariances, such as scale and rotation, that can only be modeled using tri-wise constraints (e.g. preserving angles between triplets of points), or an homography transformation, that requires four-connected nodes. We believe that matching complex scene structures requires high dimensional representations, not only to make the matching robust against several transformations, but also to encode the prior knowledge of the problem with high descriptive power. This strongly motivates one of the contributions of this dissertation, which consists of profiting from these descriptive elements using high-order graphical models, and exploiting the structural information of the scene to, ultimately, improve the matching performance. In practice, high order matching is achieved by formulating an objective function whose terms depend on several (more than two) variables simultaneously.

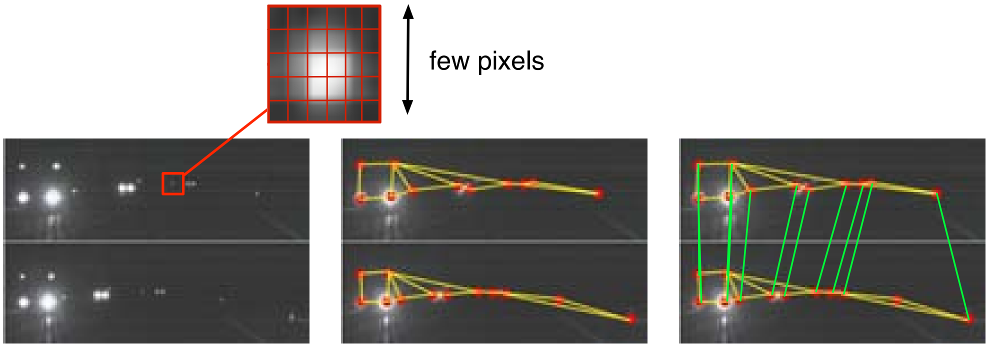


Figure 1.3: Multiple Target Tracking for Intelligent Headlights Control. Left: set of input frames. Center: detect interesting points, and build graph representation. Right: perform graph-matching. The yellow edges represent geometric constrains between the graph nodes, while the green lines represent the estimated correspondences between nodes on different frames. Note the small size of some of the blobs, which makes difficult to extract discriminative appearance features.

1.3 Problems addressed and Contributions

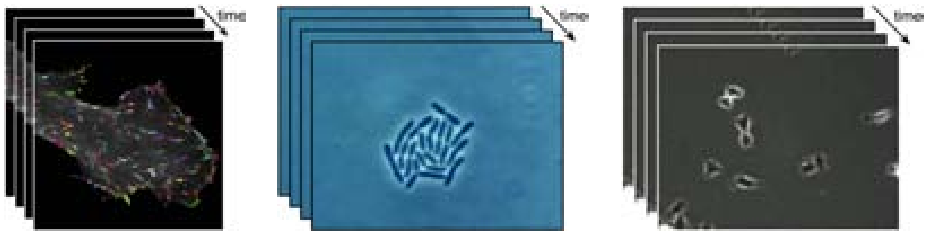
In this work we address several matching problems using a methodology conceived under the common subject of *inference on graphical models*. In the first part of the thesis, we tackle the problem of matching for *multiple target tracking and data association*, motivated by an application of intelligent headlights' control (see Figure 1.3). We address in a consistent manner both the problem of tracklet creation and data association. In the second part of the thesis we extend our framework to dense matching in the context of scene understanding, combining segmentation and matching to extract certain semantic information from the scene, with applications to region matching and co-segmentation (See Figure 1.5).

1.3.1 Multiple Target Tracking and Data Association

Along the first half of this dissertation we develop a matching-based Multiple Target Tracking (MTT) system. The original motivation of this work was to aid a classifier of vehicle headlights applying MTT. Such system would incorporate temporal (tracking) information into the classification framework, with the goal of improving the classification results. This is specially interesting in the case of tiny targets which are represented in the images with few pixels and lack any discriminative appearance features, as illustrated in Figure 1.3.

We start by generating a set of local tracks (trajectory fragments). Our method goes beyond the one-to-one assumption by formulating the tracking problem as a many-to-many probabilistic graph matching. This allows accounting for target occlusions, merges and splittings, which are due to the intersection of the target trajectories, or because groups of far away targets are imaged as a unique blob.

The process of assigning measurements to the observed targets is called *data association*. This process has to cope with certain level of ambiguity and uncertainty spanning from missing or incomplete observations of targets. One of the main tasks of data association is to handle targets appearing and disappearing, merging and splitting, at the same time that is able to validate the data (which measurement is true), and assign the validated measurement to the correct target. Following the task of tracklet creation, we generalise our framework to several tracking scenarios, classified by the relationship between the target behavior and its measurements. The goal is to simultaneously deal with the following



(a) (b) (c)

Figure 1.4: Examples frames of live cell tracking applications. Many-to-many matching is specially interesting to explicitly model the process of cell mitosis. At the left, fluorescent proteins (lung epithelial cells) [82]. At the center a video of dividing bacteria. At the right, live cell video produced by the Large-Scale Digital Cell Analysis System [25].

realistic conditions:

1. All or most objects share a very similar appearance, or it is difficult to effectively extract discriminative characteristics from it, as can be seen in Figure 1.4.
2. Objects are imaged at close positions so there is a data association problem which becomes worse when the number of targets is high.
3. The objects to be tracked may lack observations for a short or long interval, for instance because they are not well detected or are being temporally occluded by another non-target object.
4. Their observations may overlap in the images because the objects are very close to each other, or because the image results from a 2D projection from the 3D scene, giving rise to the merging and subsequently splitting of tracks.

This later condition poses the additional problem of maintaining the objects identity when their observations undergo a merge and split.

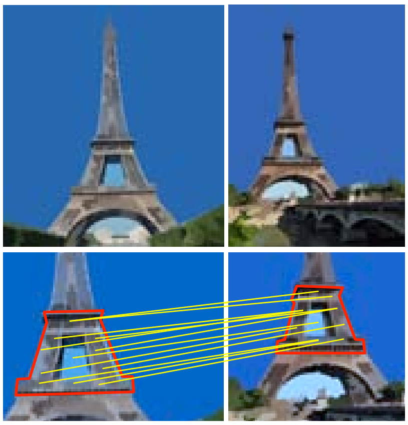
To show that our tracking and data association method generalises to other problems under similar assumptions, we provide results on various applications like headlight tracking or *live cell* tracking (Figure 1.4). This last case is very convenient to illustrate the advantages of many-to-many matching applied to tracking, since we can explicitly model the process of cell mitosis, which is seldom taken into account in the tracking literature.

Our proposed framework is composed by two sequential processes: tracklet creation and data association. The first consists of inferring tracklets (small tracks) within a small window of frames, and the second groups these tracklets into long tracks. We propose a novel two-layered probabilistic graphical model, and formulate the problem as a MAP inference on a MRF. The main contributions presented in this part of the dissertation are the following:

- Our method efficiently handles *observations* merging, splitting and occluding, as well as *targets* dividing (cell mitosis) through a novel graph-based data association algorithm.
- The method is able to fulfill the real-time requirements of the motivational headlight-control application at the same time that improves the accuracy of the original classification system.

1.3.2 Region Matching

Most matching methods are restricted to work with sparsely sampled sets of points conveniently chosen according to their discriminative power (e.g edge-points, local maxima in the scale-space). Even though


(a)

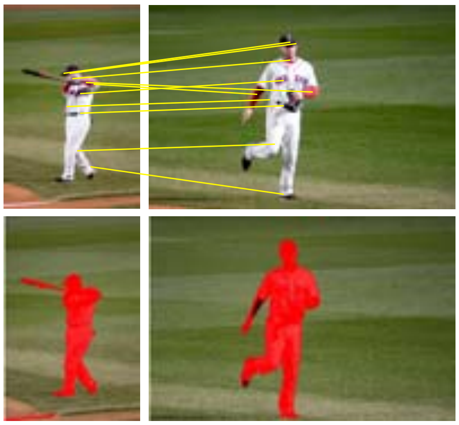

(b)

Figure 1.5: Examples of results of our region-matching contributions. In (a), dense matching of regions between two images of the Eiffel tower. In (b) foreground/background separation in the iCoseg database.

having a sparse set of interest points identified by local descriptors can be effective, we believe that going beyond this locality using a dense coverage of features is preferable, since the local ambiguity of the sparse points is hard to be solved: the larger image regions to match, the more reliable the correspondences [18]. A naive approach to dense matching would be simply to increase the number of interest points, which is not feasible due to the high computational cost. Instead, we achieve dense matching by exploiting the grouping provided by bottom-up segmentation. Our major goal is to move from point matching towards dense feature matching, with the aim of constructing richer representations of the scene elements, which we foresee as a potential improvement when applied to category-level object matching, retrieval and recognition.

We present an image region matching algorithm which establishes correspondences between regions from two segmented images. We develop an abstract graph-based representation which enciphers the image into a hierarchical graph, exploiting the scene properties at two levels: (1) the invariant similarity and spatial consistency of the image semantic objects is encoded in a graph of commute times, and (2) the cluttered regions belonging to each of the semantic objects are represented with a shape descriptor. We formulate region matching as a high order energy optimization problem, maximizing the spatial consistency and appearance similarity.

The main contributions of this part of the dissertation are the following:

- We explicitly handle many-to-many region matching, which is a difficult and critical issue due to the instability of the segmentation under slight image changes.
- Our method is robust against large variations of illumination and viewpoint.

Finally, we demonstrate the matching approach applied to images of world famous buildings, captured under different lighting and viewpoint conditions.

1.3.3 Image and Video Co-Segmentation

Co-segmentation is defined as jointly partitioning multiple images depicting the same or similar objects, into foreground and background. Figure 1.6 shows examples of results on the iCoseg database. Our method is built around the assumption that the foreground appearance does not change across the different images, independently of the background. Recent co-segmentation works require the background to change significantly between different images in order to perform well. Our method overcomes this difficulty by explicitly modeling the foreground and background appearance distributions.

The availability of several sources of information (images) suggests that applying matching to put into correspondence the scene elements will be advantageous to understand the semantics in the scene and, ultimately, facilitate the task of distinguishing between the foreground and background elements. It is important to remark that we are interested in *object* co-segmentation: that is, our method segments objects, not *stuff* [5]. As we will see later, this has considerably influenced our decisions when developing our method.

Our main goal is to improve previous co-segmentation methods in two aspects: (1) increase the segmentation accuracy and (2) avoid a high level of supervision (this is requiring annotated training data, or other kinds of human supervision such as hand-made scribbles [**?**]). We report results on iCoseg, a challenging dataset that presents extreme variability in camera viewpoint, illumination and object deformations and poses. We also show that our method is robust against large intra-class variability in the MSRC database.

We present a multiple-scale multiple-image generative model, which jointly estimates the foreground and background appearance distributions from several images, in a non-supervised manner. Region matching is applied to exploit inter-image information by establishing correspondences between the common objects that appear in the scene.

Several contributions are presented in this part of the dissertation:

- We suggest a novel non-supervised approach to the problem of co-segmentation.
- In contrast to other co-segmentation methods, our approach does not require the images to have similar foregrounds and different backgrounds to function properly.
- Computing many-to-many associations of regions allow further applications, like recognition of object parts across images.
- We apply the idea of co-segmentation to video which, to the best of our knowledge, has never been introduced before in the literature.

1.4 Thesis Origins and Evolution

This dissertation originated in the wake of a project developed by Antonio López for Volkswagen. The project goal was to automate the control of vehicle headlights at night-time, using a mounted camera and a classifier to label the scene blobs as vehicles or non-vehicles. Even though the results were fairly satisfactory, once the project was finished we decided to improve its performance by including a tracking module in the classification system. Due to a growing interest of the vision research community in graphical models, and the specific characteristics of the problem (feature-less blobs), we chose graph matching and MRFs as our basic tools. After several months of intense work, we managed to overcome the classification results thanks to the tracking system, and published two conference papers [85, 86] and one journal paper [89].

The second part of the thesis was greatly influenced by my short stage in École Centrale Paris, in the MAS Laboratorie with professor Nikos Paragios. From that valuable experience, I jealously guard two memories: the beauty of Paris, and the challenging working environment which I confronted with courage and enthusiasm. We focused in the problem of dense matching on pairs of images, following the thread of many-to-many high-order matching using graphical models. Several interesting ideas flourished during those five months, which we applied to the problem of region matching. However, we later felt that several other applications could benefit from the concept of dense matching, which motivated the last contributions of this dissertation. The problem of co-segmentation seemed specially

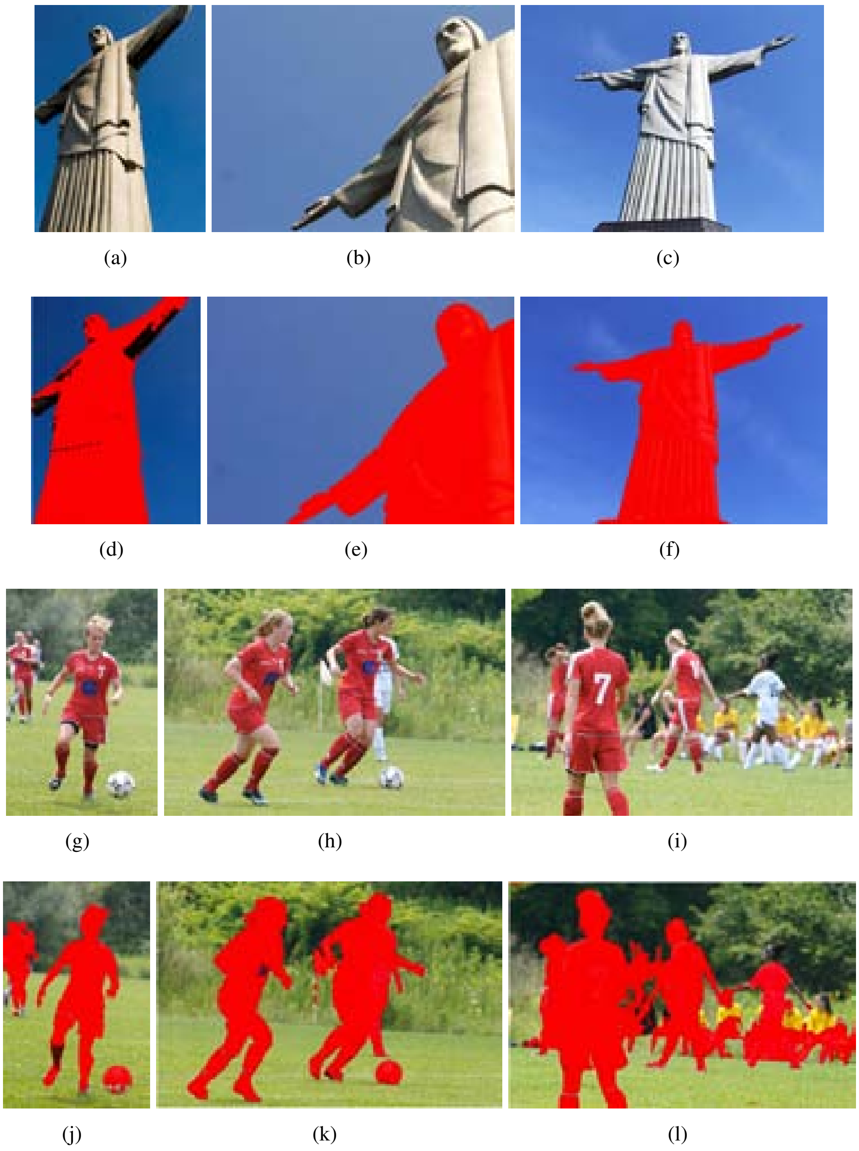


Figure 1.6: Examples of results on the iCoseg database. In (a-c) three input images of the class *Christ the Redeemer*, and in (d-f) their corresponding results. In (g-i) input images of the class *Woman Soccer Players*, and in (j-l) their co-segmentation results. Note that the yellow players in image (l) are left out because are not considered as part of the common foreground, due to the chromatic differences with the rest foreground elements.

suitable, and our first experiments provided promising results. Eventually, we developed a new non-supervised co-segmentation method that could compete with the state-of-the-art supervised methods, finally publishing two papers [87, 88], one of those in the CVPR 2012. Our latest contribution was to extend the same concept to sequences of images. We devised a new application of video segmentation that exploits examples from different video sources, and we proposed a novel application which we call *Video co-Segmentation*. This idea has been recently submitted to the Asian Conference on Computer Vision 2012.

1.5 Thesis outline

Chapter 2 lays the relevant foundations of the methodology used along the rest of the dissertation, covering basic formulation about graphical models and inference algorithms. In Chapter 3 we introduce the motivational application of headlight control, and we formulate the problem of tracking as a probabilistic graph matching. Chapter 4 extends this formulation to perform not only tracklet creation but also data association and creation of long stable tracks, always under the umbrella of Graphical Models. In Chapter 5 we move from the feature-less point matching of previous chapters, towards dense feature-attributed region matching, combining segmentation and matching to build an hierarchical representation of the scene correspondence. In Chapter 6 we mature the idea of region matching and apply it to the challenging problem of co-segmentation and part-based co-recognition. Chapter 7 extends this formulation from images to video co-segmentation. Chapter 8 presents the conclusions of this thesis, and avenues for future work.

The majority of chapters in this thesis are written following a similar structure. The reason is that they have been built on the basis of our papers published at journals and conferences (see the List of Publications chapter). First, an introductory section presents the problem and the related work. Then the method and formulation is detailed in the central pages, followed by experimental validation and discussion. Each chapter closes with a conclusion section and future work.

Chapter 2

Foundations

2.1 Introduction

Along this dissertation we have used a similar method to address different problems. Generally, we are first concerned about defining an objective function that represents the problem, and then we optimize this function to obtain a set of values (a priori unknown) that are a valid solution to this problem. For instance in the context of graph matching, we would define first our unknown elements (variables) which usually are the correspondences between the two sets of points, and also encode within our objective function the *contextual constraints* of our problem (e.g preserve distances between neighboring elements). After optimizing the function we would obtain the set of most likely correspondences that fulfill the constraints. Usually our solution will not be optimal. Instead, we pursue a reasonable balance between the accuracy of the solution and its computational feasibility.

In this work we make use of high-order Graphical Models applied to classic Computer Vision problems such as feature matching or image segmentation. In the following, we give insights on the potential benefits of high-order models and explore some of the alternatives that make tractable the optimization of such models.

2.2 Graphical Models

Graphical Models (GMs) are the marriage between probability theory and graph theory. More formally, GMs are diagrammatic representations which encode complex probability distributions in a high-dimensional space. One of their main advantages against other probabilistic representations, is that they provide a simple way to express structural properties of a probabilistic model, providing an intuitive way of modeling context-dependent entities such as image pixels and correlated features.

One typical classification of these graphical representations can be established looking at the edges between variables. The ones that use directed graphs are called Bayesian Networks, while the ones using undirected graphs are called Markov Networks. The main difference is that in the former the edges express probabilistic conditional dependence, whereas in the later the edges just establish a degree of compatibility between the latent variables they relate. In this thesis we are interested in the factorization induced by Markov Networks, and specifically we focus on Markov Random Fields and Conditional Random Fields.

When applying probabilistic GMs, we are generally interested on inferring certain quantities identified in the context of a problem. We can represent these unknown values as a set of discrete random variables $\mathbf{x}$. Lets assume that some other of those quantities are observed *evidence*, represented by a set of instantiated variables **y**. All the variables will be subscripted by integers (e.g, $s = 1, x_s$ denotes the single variable x_1), and sets of variables will be subscripted by integer sets (e.g., where $A = \{1, 2, 3\}, \mathbf{x}_A$ denotes the set $\{x_1, x_2, x_3\}$). Usually, we are interested on finding the most likely state of the vector of variables **x**, or Maximum a Posteriori (*MAP*), given the *evidence* or observation vector **y**. Formally,

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}), \tag{2.1}$$

where the *posterior* factorizes into *likelihood* and *prior*, as follows:

$$p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x}). \tag{2.2}$$

A probabilistic GM comprises a set $\mathcal{V}$ of nodes, and a set $\mathcal{E}$ of edges. Each node represents a random variable $x \in \mathbf{x}$, and the links between them represent probabilistic relationships between these variables. In this manner, the graph captures the way in which the joint distribution of random variables decomposes into a product of factors.

In this thesis we have made extensive use of undirected graphical models. However, for the sake of completeness we found convenient to add a brief notion on directed graphs.

2.2.1 Bayesian Networks

In the context of Bayesian Networks, each directed edge in the graph represents a statistical relationship between a pair of variables. This defines an independence structure: the state of one node depends only on the state of the parent nodes. We can now state in general terms the relationship between a given directed graph and the corresponding distribution over the variables. For a graph with K nodes, the joint distribution is given by:

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|pa_k), \tag{2.3}$$

where pa_k denotes the set of parents of x_k, and $\mathbf{x} = \{x_1, ..., x_k\}$. See Figure 2.1.(a).

2.2.2 Markov Random Fields

Recall the definition of the discrete random variable vector **x** and its corresponding observations **y** introduced in Section 2.2. We assume there is some statistical dependency between x_i and its observation y_i, which we call *unary potential* and we write as $\phi_i(x_i)$. Note that the "observed" nodes y_i are *embedded* into functions of the "hidden" nodes x_i, and thus we write $\phi_i(x_i)$ as a short-hand for $\phi_i(x_i, y_i)$. We also assume certain structure on the elements x_i, represented by the compatibility function $\psi_{ij}(x_i, x_j)$, where ψ_{ij} only connects nearby positions. Since there is no implicit "parenting" relationship between neighboring nodes, we write $\psi_{ij}(x_i, x_j)$ instead of $p(x_i|x_j)$. This is the typical configuration of a pairwise MRF, because the potential functions depend on a maximum of two unknown variables. Figure 2.1.(b) shows an example of pairwise MRF. To generalise to high order potentials, we need to introduce the concept of *clique*.

A *clique* is defined as a subset of the nodes in a graph such that there exists a link between all pairs of nodes in the subset. A *maximal clique* is a clique such that is not possible to include any other node from the graph in the set without it ceasing to be a clique [13]. If we denote the variables in a clique C by $\mathbf{x}_C$, the joint distribution expressed by the MRF can be written as a product of *potential functions* $\psi_C(\mathbf{x}_c)$, over the cliques of the graph:

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{C}\psi_C(\mathbf{x}_C), \qquad Z = \sum_{\mathbf{x}}\prod_{C}\psi_C(\mathbf{x}_C), \tag{2.4}$$

where Z is called *partition function*, and acts as a normalization constant that ensures that the distribution $p(\mathbf{x})$ is correctly normalized. Moreover, since $p(\mathbf{x}) > 0$ has to be satisfied, the potential functions are restricted to be strictly positive. Following the Hammersley-Clifford theorem [39], this is a necessary and sufficient condition under which such distribution can be factorized over the cliques of the graph, and represented by a Gibbs measure:

$$\psi_C(\mathbf{x}_C) = \exp\{-V_C(\mathbf{x}_C)\} \tag{2.5}$$

where $V(\mathbf{x}_C)$ is called a *potential function*. The joint distribution is defined as the product of potentials, so the total energy is obtained by adding the energies of each of the maximal cliques:

$$E(\mathbf{x}) = \sum_{c} V_C(\mathbf{x}_C), \tag{2.6}$$

which is called the energy of the system, or objective function. Minimizing this function is equivalent to maximizing the joint probability $p(\mathbf{x})$.Along this dissertation, we indistinctly refer to maximization of the posterior or minimization of the energy, depending on the given problem formalism and methodology.

Conditional Random Fields

We often have access to measurements that correspond to variables that are part of the model. This means that the pairwise (or higher) potential functions also depend on the set of observations **y**. In this case the partition function as well as the potentials become a function of the observations, and we can directly model the conditional probability distribution as:

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z(\mathbf{y})}\prod_{C}\psi_C(\mathbf{x}_C, \mathbf{y}), \qquad Z(\mathbf{y}) = \sum_{\mathbf{x}}\prod_{C}\psi_C(\mathbf{x}_C, \mathbf{y}), \tag{2.7}$$

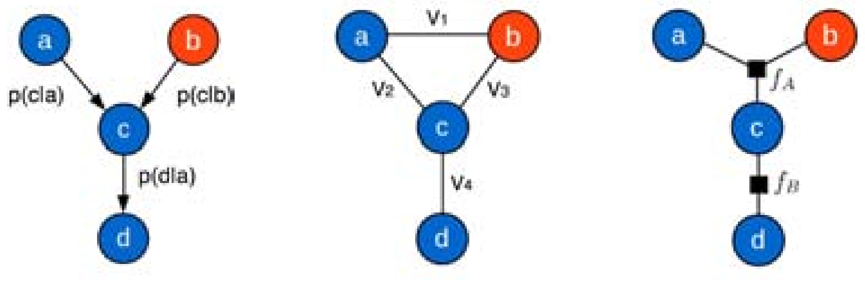
In such a configuration we say that the graphical model is a Conditional Random Field (CRF). However in the literature the distinction between MRF and CRF is sometimes omitted, and the models are commonly referred to as MRFs. Note the observed variables are not treated as random variables in the model, and are sometimes excluded from the formulation. For instance, we can refer to the conditional pairwise potential $V(x_1, x_2, y_1, y_2)$ as $V(x_1, x_2)$.

2.2.3 Factor Graphs

Being Bayesian Networks and Markov Random Fields the most commonly used graphical models, a message passing inference algorithm can be cast in a simple form using a more general type of graphical model: factor graphs [52]. A *factor graph* is a bipartite graph consisting of *variable* nodes $i \in \mathcal{V}$ and *factor* nodes $I \in \mathcal{F}$, a family of random variables $(x_i)_{i\in\mathcal{V}}$, a family of *factors* $(f_I)_{I\in\mathcal{F}}$, and a set of edges $E \in \mathcal{V} \times \mathcal{F}$. A variable node i is connected to a factor node I by an undirected edge, if the factor f_I depends on the variable x_i. The probability distribution encoded by the graph is given by:

$$p(\mathbf{x}_{\mathcal{V}}) = \frac{1}{Z}\prod_{I\in\mathcal{F}} f_I(\mathbf{x}_{N_I}), \qquad Z = \sum_{\mathbf{x}_{\mathcal{V}}}\prod_{I\in\mathcal{F}} f_I(\mathbf{x}_{N_I}), \tag{2.8}$$

where N_I contains all the variables on which the factor f_I depends. Note that there is a direct equivalence between the factors f_I and the potential functions introduced in Eq. (2.7).



(a) Directed Graphical Model / Bayesian Network

(b) Undirected Graphical Model / Markov Random Field

(c) Factor Graph

Figure 2.1: Examples of three types of Graphical Models. Blue nodes denote hidden variables and red nodes denote observed variables.

A Markov Random Field can be represented by a Factor Graph by introducing a factor for each clique potential. Factor graphs naturally express the factorization structure of probability distributions, which is a convenient feature for approximate inference algorithms that exploit this factorization. We can distinguish between two types of undirected graphical models: (1) those which factors connect only pairs of nodes, called *pairwise Markov Random Fields* and (2), those which factors connect more than two nodes, called *high-order Markov Random Fields*.

2.2.4 Inference

Inference in a Graphical Model is the process of computing information about the *hidden* variables $\mathbf{x}$, given the observed variables $\mathbf{y}$. Depending on the information one is interested in, two types of estimation are available:

1. compute the configuration $\mathbf{x}^*$ that maximizes the posterior probability $p(\mathbf{x}|\mathbf{y})$, which is called *maximum-a-posteriori* estimation (MAP).
2. compute the *marginal distribution* over a set of hidden variables A, as $p(\mathbf{x}_A|\mathbf{y})$.

While all experiments in this dissertation are based on MAP inference, computing marginals may be important for future applications, and should be taken into account when choosing the preferred inference algorithm. In general, inference in discrete graphical models is NP hard. Nevertheless, there are efficient algorithms to perform inference in specific types of graphs (See Table 2.2). Belief Propagation (BP) is an algorithm that efficiently computes exact marginal and posterior probabilities by message-passing on a factor graph. Each node passes messages to its neighbors: variables to factors, and factors to variables. The outgoing messages are functions of the incoming messages at each node, and this iterative process leads to either *convergence* or goes on *ad in nitum*. If the graph has no cycles, that is, a tree-structured graph, the update schedule of BP will converge towards a unique fixed point, and the variable beliefs will be exact. However, for many problems of practical interest the structure of the graphs is very likely to have loops. In such cases, we need to resort to approximation methods.

It is convenient to discriminate two types of messages: the ones sent from factors to variables $\mu_{f\to i}(x_i)$, and the ones sent from variables to factors $\mu_{i\to f}(x_i)$. See figure 2.2 for an example. The output messages, given in terms of incoming messages are defined by the following update equations:

$$\mu_{j\to f}(x_j) \propto \prod_{s\in ne(x_j)\setminus f} \mu_{f_s\to j}(x_j), \tag{2.9}$$

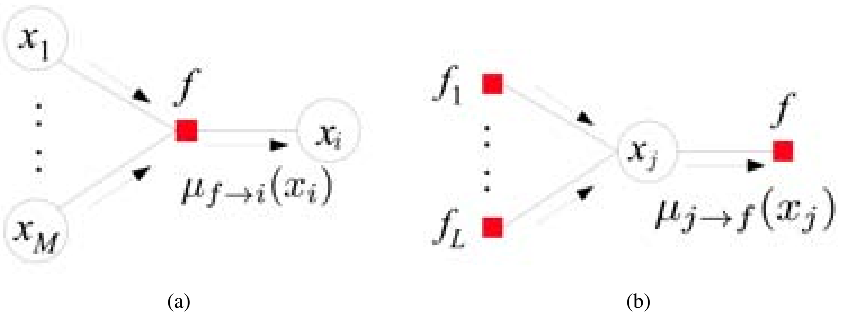


Figure 2.2: Examples of factor graphs showing the message propagation. In (a), the messages that go from factors f to variables x_i. In (b), messages from variables x_j to factors f.

$$\mu_{f\to i}(x_i) \propto \sum_{x_{ne(f)\setminus i}} \psi(x_{ne(f)}) \prod_{j\in ne(f)\setminus i} \mu_{j\to f}(x_j), \tag{2.10}$$

where $ne(f)$ denotes the neighbor variables of factor f, and $ne(x)$ denotes the neighbor factors of variable x. If BP converges to some fixed point, the variable marginals (beliefs) can be calculated from the fixed point messages.

Approximate Inference

One simple approach to perform approximate inference in a graph with loops is to apply the message-passing algorithm in the graph with loops, which is called *Loopy Belief Propagation*. Ignoring the existence of cycles may lead to unstable and incoherent probabilities of the network nodes, and the message may circulate indefinitely around the loops [76]. However, there has been significant empirical success when applying loopy belief propagation in several applications [73].

Wainwright et al. introduced a message-passing algorithm with better convergence properties than BP, called Tree Re-weighted Belief Propagation. This algorithm attempts to find a convex combination of tree-structured distributions that yields a tighter bound, and in some case yields an exact MAP configuration for the original graph with cycles.

Another class of inference techniques related to MAP estimation based on *graph-cuts* has become popular. The idea is that good MAP approximations can be obtained in certain types of Markov Random Fields by solving min-cut/maximum-flow graph problems. The main disadvantage in this case is the restriction on the type of MRFs, whose pairwise potentials should be a *metric* or a *semi-metric*. For any three labels α, β, γ,

$$E(\alpha,\beta) = 0 \Leftrightarrow \alpha = \beta, \tag{2.11}$$

$$E(\alpha,\beta) = E(\beta,\alpha) \geq 0, \tag{2.12}$$

$$E(\alpha,\beta) \leq E(\alpha,\gamma) + E(\gamma,\beta), \tag{2.13}$$

if E satisfies 2.11 it is called a *metric*. If E satisfies only 2.12, 2.13 it is called a *semi-metric*. In the case of minimizing an arbitrary function of binary variables, the necessary condition for the function to be minimized via graph-cuts is that its pairwise terms satisfy

$$\psi(0,0) + \psi(1,1) \leq \psi(0,1) + \psi(1,0). \tag{2.14}$$

If the function satisfies the above condition, we say it is *sub-modular*. Table 2.2 shows a list of different inference algorithms, with some of its characteristics.

2.3 High Order Graphs

Message-based computations are very expensive in the context of high order graphs since, in the general case, they have a complexity which is exponential in the size of the largest clique. For this reason, most applications of Loopy Belief Propagation are based on simple pairwise MRF models. However, in this dissertation, we put especial emphasis in the necessity of encoding high level structural dependencies among our problem variables. A recent work by Rother et. al. [83], introduces a mechanism to transform a high order problems into an equivalent quadratic functions, given that such high order terms present a sufficient sparse structure. We have applied this technique at several points of our research with very satisfactory results. Moreover, the binary variable representation of the matching problem proves to be specially adequate, since the grouping of these variables naturally lead to very sparse potential functions. In the following, we introduce the mechanism to perform the transformation.

Minimizing sparse high order potentials

Lets consider a factor ψ of three binary variables $\{x_a, x_b, x_c\}$, which we would like to reduce to pairwise. Our function should assign a low cost (high probability) to a small subset of variable realizations which are *favored*, while the rest are *penalized*. Minimizing this cost is equivalent to finding the most probable configuration of variable values. In our example we consider only one configuration with low cost: for instance, the realization $\{x_a = 1, x_b = 1, x_c = 1\}$. The objective function is expressed as

$$\psi(x_a, x_b, x_c) = \begin{cases} \theta_0 & \text{if } x_a = 1, x_b = 1, x_c = 1 \\ \theta_1 & \text{otherwise} \end{cases} \tag{2.15}$$

where $\theta_0 \leq \theta_1$ are the costs for each configuration. This function can be transformed to pairwise by adding a $(t+1)$ state switching variable z, where t is the number of clique configurations to *favor*. In our case only one configuration is allowed $(t = 1)$, and z has two possible states $\{0, 1\}$. The function minimization is then expressed as,

$$\min_{x_a, x_b, x_c} \psi(x_a, x_b, x_c) = \min_{x_a, x_b, x_c, z} f(z) + \sum_{i \in (a,b,c)} g_i(x_i, z) \tag{2.16}$$

where f is a unary potential, defined as: $f(0) = \theta_0$ and $f(1) = \theta_1$. The pairwise function $g_i(z, x_i)$ assigns a low cost if the value of x_i is consistent with the configuration indexed by the variable z, and a high cost otherwise. Table 2.1 shows the values taken by the g function. The left table shows the values of g_a depending on x_a and z. In the right table, we fix the z value to zero and show the values of the three g functions. Notice that the value $z = 0$ indexes the only configuration (1,1,1) which we are interested of assigning a low cost θ_0, while $z = 1$ indexes the rest of configurations which get a higher cost θ_1.

Table 2.1: g function

(x_a, z)	g_a
(0, 0)	∞
(0, 1)	0
(1, 0)	0
(1, 1)	0

$x_a, x_b, x_c, z = 0$	g_a, g_b, g_c
(0, 0, 0)	∞, ∞, ∞
(0, 0, 1)	$\infty, \infty, 0$
(0, 1, 0)	$\infty, 0, \infty$
(0, 1, 1)	$\infty, 0, 0$
(1, 0, 0)	$0, \infty, \infty$
(1, 0, 1)	$0, \infty, 0$
(1, 1, 0)	$0, 0, \infty$
(1, 1, 1)	0, 0, 0

Whenever the variables $\{x_a, x_b, x_c\}$ have values coherent with the configuration indexed by z (e.g all set to 1) the function g assigns a cost zero, and infinity otherwise. The value $z = 1$ indicates a configuration which is not interesting, in which case g is always zero, meaning that the function ψ gets directly the cost θ_1 from f. Extending this method to minimize an arbitrary high order function with several low-cost configurations is straightforward, simply by summing several potentials as the one described in Eq. (2.16). We refer the reader to the work of [83] for extensive formulation and details of the technique.

	situation				results/output				
Algorithm	states	topology	pairwise	high order	marginals	convergence	optimal	complexity	references
Plain BP	>100	tree	any	yes	yes	yes	yes	$O(nh^L)$	[13,32]
Loopy BP	>100	any	any	yes	yes	no	no	$O(eh^L i)$	[13,32]
TRW-S	any	any	any	yes	yes	yes	no	$O(eh^L i)$	
on trees	any	tree	any	yes	⇒ same as plain BP				
on 2-state grids	2	grid	sub-modular	no	⇒ same as Graph-cut				
Graph-cut	2	any	sub-modular	no	yes	yes	no		
α/β swap	2-32	any	semi-metric	no	no	yes	no	$O(h^2 B^{\alpha\beta} i)$	
α-expansion	2-32	any	metric	no	no	yes	no	$O(hBi)$	

Table 2.2: List of Inference algorithms. n: number of nodes, h: number of states/labels per node, e: number of graph edges, i: number of iterations, k: number of states covered by the truncated pairwise term. L: size (n. of variables) of the largest clique. $B^{\alpha\beta}$: cost of the graph-cut subroutine on a graph only containing nodes with states α, β. B : cost of the graph-cut subroutine. This table has been adapted from the ETH Zurich Computer Vision Laboratory (Re.Clu.Se) reading club, by Thomas Deselaers.

Chapter 3

Multiple Target Tracking for Intelligent Headlights Control

3.1 Introduction

This chapter focuses on the problem of constructing blob tracks (tracklets), which is actually one of multiple target tracking, but under two special conditions: 1) blobs are *featureless*, that is, they are undistinsguishable by their appearance (color, size, shape) and we can think of them as particles or points; 2) we have to deal with frequent occlusions as well as blob splits and merges. Those blobs come from the frame by frame segmentation of night-time driving video sequences. As such, the main difficulty is to correctly track far away targets that are represented as small (few pixels) feature-less blobs in the images. We approach it in a novel way, by formulating the problem as a maximum a posteriori inference on a conditional random field. We present qualitative (in video form) and quantitative results which show that our new tracking method achieves good tracking results as well as satisfying the real-time requirements of a driving assistance application.

Accident statistics demonstrate that driving at night is considerably more dangerous than its daytime counterpart [1]. This can be attributed, among other causes, to the lower performance of the human visual system under poor ambient lighting conditions: color and depth perception, and therefore object saliency, are reduced. Some studies like [2] show that drivers turn on high beams much less frequently than they could: only about one fourth of the time during which traffic conditions would justify their use. Among the reasons for this behavior, we highlight two: the need for a manual (and eventually, frequent) operation and the fear of dazzling drivers of leading, oncoming or overtaking vehicles. Recently, the combination of specialized on-board cameras, fast processors and machine learning techniques has enabled some automotive machine vision suppliers and companies to develop prototypes of 'intelligent headlights' controllers (IHC) for high-end car series, with acceptable results. However, as we will discuss, this problem is not completely solved.

The core of an IHC is pattern classification software able to discern bright spots (or image blobs, in computer vision terminology) originating from vehicles' head or rear-lights from those due to road lamps, traffic-lights and reflective infrastructure elements like poles, lane markings and traffic signs. Among the detected vehicles, the one with lowest vertical image coordinate, corresponds to the closest. By initially calibrating the camera and the light beams, the camera and the headlight principal axes can be aligned to adjust the light cone right under the aforementioned vehicle. This is achieved through a cut-off filter placed in front of each headlight, which avoids dazzling other drivers and maximizes the

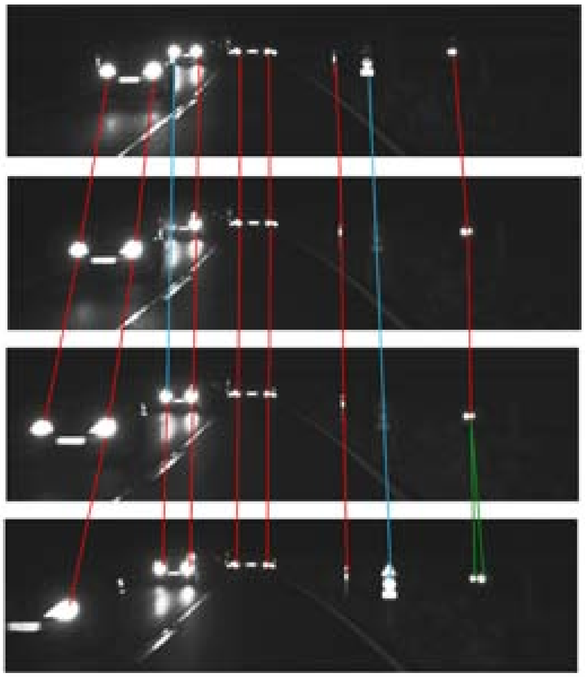


Figure 3.1: Example of blob tracking along four frames. In red, one–to–one associations, in blue two occlusions and in green a splitting.

illuminated area. Figure 3.2 shows a simple scheme of the headlight control system.

The main difficulties are due to the requirements of a low classification error rate, real-time processing and, perhaps more importantly, the need to detect all vehicles within the image field of view as soon as possible. This poses a problem in the case of very distant vehicles. Specifically, we consider a vehicle distant at 600 meters for oncoming vehicles and 400 meters for leading vehicles, because of the different glaring effect of the host vehicle high-beams on their drivers. At these distances, head/rear-lights are imaged as tiny blobs, fewer than ten pixels in size, so that appearance features such as intensity, color and shape do not provide sufficient information to perform a reliable classification of individual blobs *when frames are examined independently*.

The literature on nighttime, on-board, vision-based vehicle detection is rather scarce. All the works reviewed first segment the image by some variant of adaptive thresholding, and then perform the classification based on features related to color, shape, size and image location. The simplest classification methods use a set of heuristic rules with fixed thresholds [6, 17, 33, 75]. Other works employ more sophisticated machine learning techniques like decision trees [26], Bayes factors [34], Hidden Markov Models [44], Support Vector Machines [4,37,63,80] and Real-AdaBoost [65,66], which can be trained and thus possess much greater adaptability. Some of these works recognize that the classifier outcome is not sufficiently reliable and that decisions for one blob are not stable along time. To solve this, they either try to track blobs or pair them as belonging to one same vehicle.

Pairing, or more generally, clustering blobs, helps to better classify them since only those that form a consistent pair, according to constraints like similar vertical position, size, shape, color etc., can originate from a vehicle [23, 33]. However, this is not a convenient strategy if vehicle detection has to be used for IHC, because the two head or rear-lights appear as two detached blobs only when the target

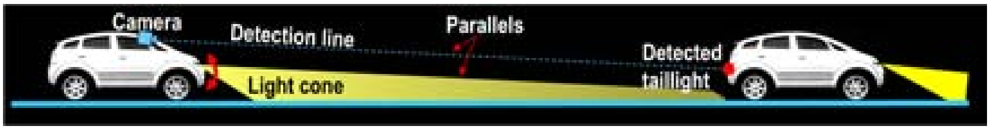


Figure 3.2: Headlight beam control system. The desired light cone is directed parallel to de detection line between the mounted camera and the closest detected vehicle light.

vehicle is close to the host vehicle. Hence, distant vehicles would never be recognized as such, since distant pairs of lights tend to join in a unique blob. Pairing suits vehicle detection for other driving assistance applications, like estimation of time to collision [6,17,34,37,75] or automatic cruise control. In these two cases, the separation of spotlights is necessary to estimate the distance to other vehicles. However, in any case, pairing usually does not consider motorbikes. Thus, pairing must be used just as an additional weak cue as in [65].

Therefore, tracking seems the only way left for an IHC to avoid the errors induced by the frame-by-frame independent classification. Specifically, the potential benefits of tracking blobs for IHC are its ability to

- increase the number of feature/attribute measurements of each tracked blob;
- provide the classifier with additional motion features [33,63,80], otherwise not available;
- allow the selection of 'interesting' blobs which are passed to the classifier as those that can be followed during a certain minimum number of frames [4,26,34,37];
- associate a confidence to the class label of a blob —high if it is consistent with labels of past frames, low if not— and make the final classification decision at the moment its confidence exceeds a certain threshold [65,66].

Few works on IHC perform tracking, probably due to its difficulty in this context. In [26,34,63] a simple nearest neighbor search is performed, based on image location and appearance features. This is also done in [4,33,37,80] with individual or clusters of blobs, though they first predict the position of blobs by means of a Kalman filter. In both cases, tracking refers to associating blobs from one frame to the next. In [66] proper tracking is replaced by a so-called 'temporal coherence analysis' whereby a confidence map is maintained, quantifying the belief in finding a vehicle blob at each pixel. This confidence is estimated on the basis of the blob labels at the frames immediately preceding the current frame. Despite fostering the temporal coherence of the classification, this method does not produce blob tracks.

In order to take full advantage of the potential benefits discussed previously, the tracking algorithm must deal successfully with blob occlusions, splittings and merges. Occlusion handling means that blobs which temporally disappear must not originate new tracks but be associated with their former track. Splittings occur when a blob corresponding to the two headlamps of an oncoming vehicle becomes two distinct blobs as the vehicle approaches. Splittings may also occur with static reflective surfaces like poles. Merging is the opposite case: as a leading vehicle (or a compound traffic sign) gets farther away, two blobs merge into a single one. These are frequent events in nighttime video sequences, and are caused by distant vehicles, light sources or reflections not directed towards the camera and distant, small or poorly reflective surfaces. In spite of their importance, none of the reviewed works which perform tracking deals with them.

Our main contributions are the following. First, we focus on the problem of building tracks of close, mid-distance and far away light sources/reflectors taking into account occlusions, merges and splits. In particular, we solve the problem of building continuous tracks in the presence of occlusions up to a certain duration. Second, we propose a new probabilistic tracking method whereby the problem is posed as a maximum a posteriori estimation in a Markov random field. The key idea is to represent

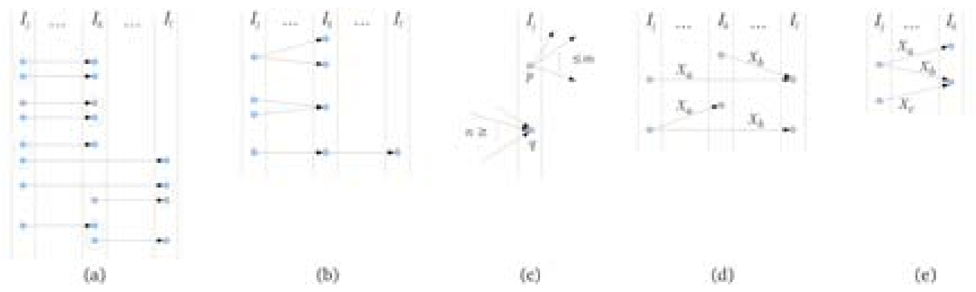


Figure 3.3: Types of associations involved in the (a,b) likelihood term and (c,d,e) the prior terms described in section 3.2. Vertical strips represent frames, circles are blobs and arrows associations between two blobs. Dashed ellipses mean neighbor blobs. Note that (a) and (b) are *all* the possible types of association pairs: in (a) from top to down, pairs leaving or arriving to neighboring blobs in the same frame, pairs leaving neighboring blobs but arriving to blobs in different frames, and pairs arriving to neighboring blobs in the same frame, but leaving from different frames. Also, the last row of (a) illustrates a pair where one association arrives to a blob, which is neighbor of the origin blob of the pairing association. (b) shows the pairs sharing a blob either at the origin, destination or intermediate frame. The constraint $X_a + X_b \leq 1$ on the association pairs in (d) precludes other pair possibilities.

the associations of two blobs from different frames within a certain time window, by a binary variable whose most probable state, either associated or not, must be estimated. Once a solution is found for every association for a time window, we propagate the result to the next frame by sliding the window. We provide extensive quantitative evaluations, based on annotations of tracks on five video sequences, and qualitative results. In addition, we will also see that the classification performance of problematic small blobs, corresponding to distant cars or small reflections, improves thanks to the proposed tracking algorithm.

3.2 Probabilistic Multiple Frame Assignment

Let w be the number of contiguous frames in a certain temporal window of the video sequence in which we want to track points (along this Chapter we indistinctly refer to these points as points, blobs or targets). We denote by $I_1, I_2, ..., I_w$ the different frames within it. Each frame contains a set of zero or more blobs, indexed by $p, q, ...$. An association a is an ordered pair of blobs from different frames, $a = (p, q)$, meaning that blobs p and q are observations (sensor measurements) from the same target, but at different frames. Let A be the set of all such associations,

$$A = \{a = (p, q) | p \in I_i, q \in I_j, 1 \leq i < j \leq w\}, \tag{3.1}$$

where $a, b, ...$ index the elements of A, so that we can denote all pairs of association without repeated combinations as $(a, b), a < b$. Let $\mathbf{X} = (...X_a...)$ be the vector of binary variables, one per association, where $X_a = 1$ if the corresponding association a exists, and zero otherwise. In the same way, the vector of all observations is denoted by $\mathbf{Y} = (...Y_a...)$, where each association $a = (p, q)$ is represented by the observation vector $Y_a = [p_x, p_y, q_x, q_y, p_{area}, q_{area}]$. Thus, each observation is a vector of measurements: the spatial coordinates and areas of its origin and destination points. Other properties could also be considered, like shape or intensity measures.

Our goal is to find the most likely configuration of the set $\mathbf{X}$ of association states, given the set of all observations $\mathbf{Y}$. This is, in probabilistic terms, to find the maximum a posteriori estimation,

$$\mathbf{X}^* = \arg\max_{\mathbf{X}} \mathbf{p}(\mathbf{X}|\mathbf{Y}). \tag{3.2}$$

In a Bayesian framework, the posterior probability of the hidden variables $\mathbf{X}$, given the observations, is proportional to the product of the likelihood and prior terms

$$p(\mathbf{X}|\mathbf{Y}) \propto p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}). \tag{3.3}$$

The likelihood term $p(\mathbf{Y}|\mathbf{X})$ represents the application specific observation model. The prior $p(\mathbf{X})$ is calculated from the application previous knowledge, which in our case consists of several constraints a feasible solution has to satisfy, as we will see later in section 3.2.2. The next two sections detail how do we define and compute these two terms.

3.2.1 Likelihood

To begin with, and for the sake of tractability, we assume that the observation model $p(\mathbf{Y}|\mathbf{X})$ factorizes as

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X}) = & \left[\prod_{a\in A} p_A(Y_a|X_a)\right] \cdot \left[\prod_{(a,b)\in N} p_N(Y_a, Y_b|X_a, X_b)\right] \\ & \cdot \left[\prod_{(a,b,c,...)\in O} p_O(Y_a, Y_b, Y_c, ...|X_a, X_b, X_c, ...)\right]. \end{aligned} \tag{3.4}$$

The first term models the likelihood of an association being active or inactive, depending on the similarity of location and area of the two blobs (p, q) involved in each association $a \in A$. The second term is the likelihood of two associations existing simultaneously. This exploits the spatial relationships of the observations of pairs of associations, imposing a local invariance to rotation and translation, and it is defined over the set N of all association pairs, as will be explained below. The third component represents the probability of a target being occluded during one or more frames, defined over the set O, which contains groups of associations involved in the occlusion of a blob. This term plays the key role of avoiding the trivial solution $X_a = 0 \ \forall a \in A$, as we will discuss later. Next, we describe in detail the specific probability models for these three terms p_A, p_N and p_O, which we call appearance, neighborhood and occlusion, respectively.

Appearance: The displacement of a point between two frames and its position are not independent. As the point approaches the camera, it moves towards the left and right image borders, and its apparent velocity increases. In contrast, it remains motionless when distant, positioned in the center of the image. Moreover, the blob position has a direct relationship with its area: the closer a blob is to the camera, the faster its area changes from one frame to another. Accordingly, an association $a = (p, q)$ is more probable if the areas of p and q are similar, and their positions change as described. We define the probability of associating blob p with blob q as

$$p_A(Y_a|X_a = 1) = \hat{f}_1(\mathbf{v}_{pq}, p_x)\hat{g}_1(|p_{area} - q_{area}|, p_x)). \tag{3.5}$$

$$p_A(Y_a|X_a = 0) = \hat{f}_0(\mathbf{v}_{pq}, p_x)\hat{g}_0(|p_{area} - q_{area}|, p_x)). \tag{3.6}$$

$\hat{f}_1$ is a density function modeling the dependency of the location and the displacement vector $\mathbf{v}_{pq}$ of the association a (See Fig. 3.5). The density $\hat{g}_1$ models the relationship between the original position

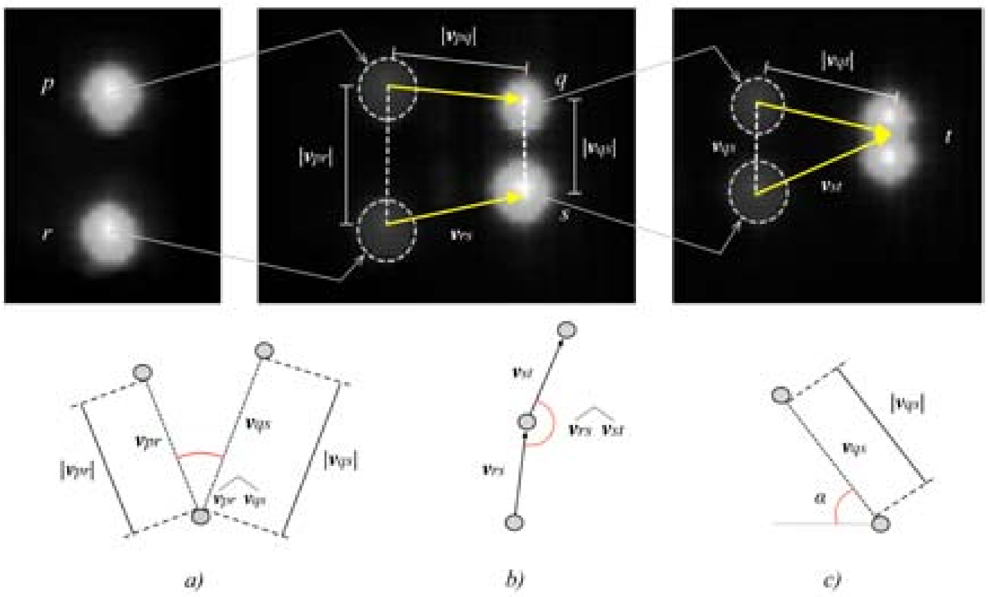


Figure 3.4: Example of the different measures used when calculating the components of p_N along three frames j, k, l. a) represents the angle and distance used in p_G. b) represents the angle when enforcing linear trajectory in the p_M. and c) represents the module vector when calculating the merging & splitting compatibility p_{SM}, and the angle against an horizontal reference vector.

of the blob and the frame-to-frame changes of the areas of the blobs. Analogously, $\hat{f}_0$ and $\hat{g}_o$ define the same correlations for the case of association a not existing.

All these probability densities are learned using a Kernel Density Estimator (KDE) [11]. The expression of $\hat{f}$, corresponds to the well known bivariate kernel density estimator

$$\hat{f}(x, y) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h_x h_y} K \left(\frac{x - x_i}{h_x}, \frac{y - y_i}{h_y} \right), \tag{3.7}$$

where K is a Gaussian kernel, h_x, h_y are the bandwidth components, which are data-driven and automatically selected, and n is the number of training data points.

Neighborhood: The term p_N is defined over the set N of pairs of associations,

$$N = \{(a, b) \in A \times A | a < b\}, \tag{3.8}$$

corresponding to Fig. 3.3a and the last row of 3.3b. The term p_N is responsible for modeling relationships between pairs of associations. We distinguish three components into which p_N factorizes, which we call *Geometry*, *Motion* and *Split & Merge*.

$$\begin{aligned} p_N(Y_a, Y_b | X_a, X_b) &= p_G(Y_a, Y_b | X_a, X_b) \\ &\cdot p_M(Y_a, Y_b | X_a, X_b) p_{SM}(Y_a, Y_b | X_a, X_b). \end{aligned} \tag{3.9}$$

Each of these terms defines the likelihood of different pairs of associations. For instance, two associations in consecutive frames, as illustrated in the third row of Fig.3.3b are involved in the *motion*

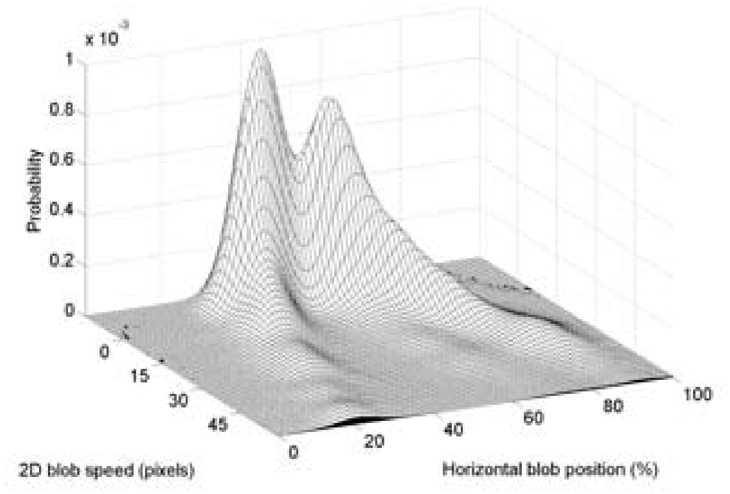


Figure 3.5: Representation of the density function $\widehat{f_1}$ of Eq. 3.5, which correlates the blob speed and image position. Most of the training samples are distant motionless light spots, which correspond to the two high density peaks. As the targets approach the camera their speed (in image coordinates) increases, and their position moves towards the image borders. Notice how the left peak is denser than the right, because incoming targets in the right lane is less frequent. In countries that drive on the left-hand side of the road, the density functions should be mirrored along the speed axis.

likelihood. On the other hand, two associations like those in the first row of Fig.3.3a are involved in the *geometric* likelihood.

Geometry: Targets do not move independently of each other. Two close targets are likely to move in a similar direction and with a similar speed. This can be seen as a local isometric mapping of the points from one frame to another. This means enforcing invariance on the distance and angles defined by pairs of points. Specifically, two associations are probable if the distance between their origin blobs is very similar to the distance between their destination blobs, and if the angle between the origin blobs is similar to the angle between the destination blobs. This is shown in Fig. 3.4a and 3.4b.

Formally, the likelihood p_G is defined over the set of pairs of blobs:

$$\begin{aligned} N_G = \{(a,b) \in N | a = (p,q), b = (r,s), \\ q \in N_p \vee p \in N_q \vee r \in N_s \vee s \in N_r\}, \end{aligned} \tag{3.10}$$

where N_p is the set of neighbors of p, according to some neighborhood relationship. In our case it is the set of k-nearest blobs of p, which are at a distance to p below a certain fixed threshold.

Let $a = (p,q)$ and $b = (r,s)$ be two associations belonging to N_G, X_a, X_b their states, and Y_a, Y_b their respective observations. Let $\mathbf{v}_{pr}$ be the vector between their origin blobs, and $\mathbf{v}_{qs}$ the vector between the destination blobs, as shown in Fig. 3.4 (top). Since we assume these measures to be independent, we can construct the probability as a convex combination of Gaussian densities, as follows:

$$\begin{aligned} &p_G(Y_a, Y_b | X_a = 1, X_b = 1) \\ &= \lambda_G \mathcal{N}(\widehat{\mathbf{v}_{pr}\mathbf{v}_{qs}}) + (1 - \lambda_G)\mathcal{N}(|\mathbf{v}_{pr}| - |\mathbf{v}_{qs}|), \end{aligned} \tag{3.11}$$

where $\mathcal{N}(x)$ denotes a Normal distribution. For the sake of readability we represent the gaussians as $\mathcal{N}(x)$, instead of $\mathcal{N}(x; \mu, \sigma^2)$, for some μ, σ. Thus, we learn their parameters, μ, σ^2 from training data, using the standard method of maximum likelihood. The parameter $\lambda \in [0, 1]$ weights the contribution of each term to the mixture of densities. The first Normal distribution measures the similarity between the orientation of $\mathbf{v}_{pr}$ and $\mathbf{v}_{qs}$, and the second enforces the similarity between $|\mathbf{v}_{pr}|$ and $|\mathbf{v}_{qs}|$.

Motion: Close targets tend to follow a linear trajectory when their positions are close enough to the camera, while far away targets, imaged around the image center, are static or oscillate up and down due to the movement of the vehicle on which the camera is installed. The set of pairs related to the motion component correspond to the third pattern of Fig. 3.3b, and is defined as

$$N_M = \{(a, b) \in N | a = (p, q), b = (r, s), q = r\}. \tag{3.12}$$

Two associations in the motion set N_M are probable if, while close to the camera, the displacement of their blobs follow a similar direction. Given two associations $a = (p, q)$ and $b = (q, r)$ from N_M, the likelihood of these associations existing simultaneously is defined as

$$p_M(Y_a, Y_b | X_a = 1, X_b = 1) = \hat{h}(\widehat{\mathbf{v}_{pq}\mathbf{v}_{rs}}, q_x), \tag{3.13}$$

where the density $\hat{h}$, depends on the angle $\widehat{\mathbf{v}_{pq}\mathbf{v}_{rs}}$ (3.4c), and the horizontal position of the central blob q. This encourages similarity of the vector directions of pairs of associations close to the image's left and right borders. Again, the correlation defined by $\hat{h}$ cannot be modeled by a simple Gaussian. A non-parametric Kernel Density Estimator is used to learn the density shape from training data.

Split and Merge: This term models the probability of two blobs merging, or one blob splitting in two. Given two associations $a = (p, q)$ and $b = (r, s)$, a splitting occurs when $p = r$, and a merging when $q = s$. The set of pairs belonging to the split and merge component follow to the first and second patterns of Fig. 3.3b, and is defined as

$$\begin{aligned} N_{SM} =&\{(a, b) \in N | a = (p, q), \\ &b = (r, s), (p = r) \vee (q = s)\}. \end{aligned} \tag{3.14}$$

The merging or splitting blobs are likely to have similar areas and to have very close positions. In addition, we restrict the mergings and splittings to happen only horizontal or vertically. This restriction reflects the nature of the merging and splittings that are originated by road pole reflections (vertical) or car headlights (horizontal). The probability of a merging or splitting is defined as

$$\begin{aligned} &p_{SM}(Y_a, Y_b | X_a = 1, X_b = 1) \\ &= [\lambda_{SM}\mathcal{N}(|\mathbf{v}_{pr}|) + (1 - \lambda_{SM})\mathcal{N}(|p_{area} - r_{area}|)] \\ &\cdot \mathcal{N}(|\pi/4 - \alpha|). \end{aligned} \tag{3.15}$$

The first two distributions form a Gaussian mixture, whose components are weighted by the parameter λ_{SM}. The first favors two targets merging or splitting when the distance between them $|\mathbf{v}_{pr}|$ is small, while the other favors area similarity. Figure 3.4c shows an example. The angle α is the angle between vector $\mathbf{v}_{qs}$ and a reference horizontal vector. Hence, the last distribution enforces horizontal and vertical alignment of the targets which are merging or splitting.

Occlusions: We say a blob p has been occluded along d frames when p, being visible in frame I_i, disappears during d consecutive frames to either appear again in frame I_{i+d}, or disappear definitively from the window. Let $A(p,d)$ be the set of all associations with origin in blob $p \in I_i$, and with destination blobs located at frames $I_{i+1}, I_{i+2}, ..., I_{i+d}$. This is defined as

$$A(p,d) = \{a = (p,q) \in A | p \in I_i, q \in I_{i+k}, k = 1...d\}. \tag{3.16}$$

The blob p is considered occluded with duration d when every $a \in A(p,d)$ is zero. Therefore, the conditional probability which models an occlusion depends on all the associations departing from p and having duration d. The set of associations involved in an occlusion, for all possible durations within a window of w frames, is

$$O = \{A(p,d) | d = 1...w-1, i = 1...w-d, p \in I_i\}. \tag{3.17}$$

The probability distribution p_O is built around two assumptions. First, the blobs close to the borders of the image are more likely to disappear. This is due to the movement of the blobs, which typically appear in the image center, and then move towards the image borders. However this does not always happen, because of leading and overtaking vehicles, which move faster than ours. Second, tiny blobs are also likely to disappear, as a consequence of the segmentation process. Finally, we encourage blobs to be associated with other blobs belonging to the closest frame possible. Put all together, for each $\{a, b, ...\} = A(p,d) \in O$,

$$\begin{aligned} & p_O(Y_a, Y_b, ... | X_a, X_b, ...) \\ &= \begin{cases} \hat{\imath}(p_x, p_{area})(1 - e^{-d}) & \text{if } X_c = 0 \; \forall c \in A(p,d) \\ 1 & \text{otherwise} \end{cases} \end{aligned} \tag{3.18}$$

The density distribution $\hat{\imath}$ models the probability of all associations coming from a blob p with duration d, being inactive. That is, blob p disappearing from d consecutive frames. The term $(1 - e^{-d})$ favors associations between blobs in nearby frames.

Note that if at least one association X_a does exist, $p_O = 1$, meaning that we do not penalize anything but blob occlusions. In the same way, every conditional distribution of the terms p_G, p_M and p_{SM} explained thus far, with the exception of the appearance term, depends on a specific realization of the random variables $(X_a = 1, X_b = 1)$. It is important to notice that the probability values for the rest of configurations, $(X_a = 0, X_b = 0)$, $(X_a = 0, X_b = 1)$, and $(X_a = 1, X_b = 0)$ are set to one, which means that the observation model does not 'penalize' these realizations. Hence, one may wonder why the trivial solution $\mathbf{X} = (0, ..., 0)$ is not the most probable. The appearance and the occlusion term, are in charge of avoiding a trivial solution in which every variable state is zero. For instance, if a blob p is not likely to be occluded in the next d frames, it will assign a low probability to the configuration $X_a = 0, X_b = 0, ...$, where $(a, b, ...) \in A(p,d)$.

3.2.2 Modeling the Prior

We include a constraint on the maximum number of blobs to which one blob can be associated. This may be used in tracking applications for which we know the bounds on the number of blobs involved in splits and merges. This gives rise to the two constraints shown in Fig. 3.3c. Given two frames I_i, I_j, from a window of length w, we define what we call the multi-assignment m-to-n constraint as

$$\sum_{a \in A(p)} X_a \leq m, \forall p \in I_i, i = 1 \dots w-1 \tag{3.19}$$

$$\sum_{b \in B(q)} X_b \leq n, \forall q \in I_j, j = 2 \dots w, \tag{3.20}$$

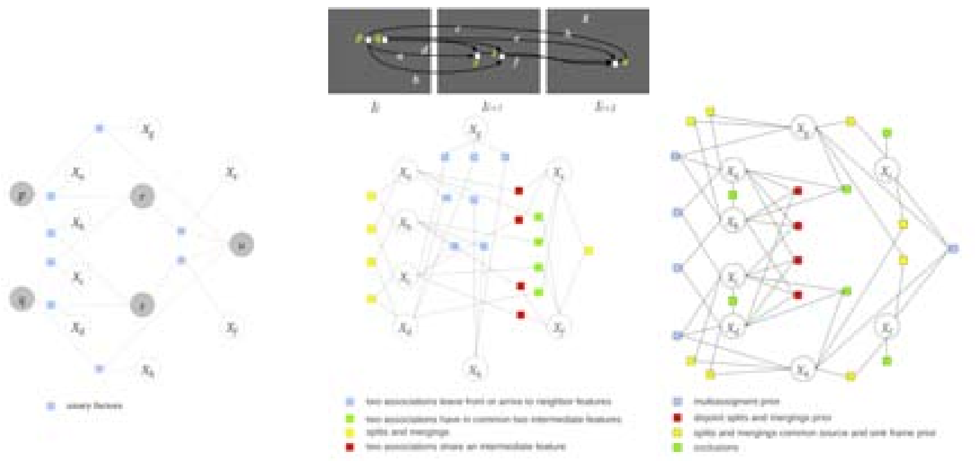


Figure 3.6: Example of factor graph for a window of three frames containing five blobs. Variables $X_a \dots X_h$ represent the eight possible associations $a = (p, r), b = (p, s), \dots h = (r, u)$. To ensure clarity, we represent the factor graph divided in three parts. In the left, only unary factors (appearance) are drawn. In the center, factors related to pairs of associations. In the right, occlusions and the three components of the prior: multi-assignment limits, splits and merges restricted to two frames, and split and merges on disjoint sets of blobs. Best viewed in color.

where $A(p)$ is the set of associations leaving blob $p \in I_i$ and $B(q)$ the set of those arriving at $q \in I_j$. In our case, $m = n = 2$, meaning that we restrict the number of targets merging or splitting to a maximum of two. For instance, when headlights or rear-lights merge or split.

Split and merge handling gives rise to two additional sets of constraints. The first, corresponding to Fig. 3.3d, comes from the condition that splits and merges occur in precisely two frames I_i, I_j. It takes the form

$$X_a + X_b \leq 1, \tag{3.21}$$

for all pairs $(a, b), a < b$ such that if $a = (p, q), b = (r, s)$ and $i < j < k$ then either $p = r \in I_i$, $q \in I_j$, $s \in I_k$ or $q = p \in I_i, r \in I_j, s \in I_k$.

The second set of constraints expresses the assumption that a merge cannot mix with a split and vice versa, as Figure 3.3e illustrates. For blobs within the same frame, the set of blobs involved in a split are disjoint from those involved in a merge. This takes the form

$$X_a + X_b + X_c \leq 2, \tag{3.22}$$

where $a = (p, q), b = (p, s), c = (t, s)$ and $p, t \in I_i$, $q, s \in I_j$, for all $1 \leq I_i < I_j \leq w$.

Note that all the constraints of Eqs. (3.19) - (3.22) have the form of an upper bound on a linear combination of a few association variables. Thus, if r is the number of constraints, all of them can be compactly expressed as $C\mathbf{X}^T \leq \mathbf{b}$, where $C = [\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_r]^T$ is a very sparse binary matrix whose rows select the variables of each constraint, and $\mathbf{b}$ is a column vector with bounds m, n, 1 and 2. Accordingly, the prior reduces to the compact form

$$P(\mathbf{X} = \mathbf{x}) = \begin{cases} 1 & \text{if } C\mathbf{x} \leq \mathbf{b} \\ 0 & \text{otherwise} \end{cases} \tag{3.23}$$

3.3 From Window Assignment to Sequence Tracks

So far, we have explained how to track targets within a window of w frames. Next, we will see how to extend the algorithm in order to track targets on longer sequences of frames, by sliding this time window.

Every association originating in the first frame of a window t, which is set as 'existing' by the inference algorithm, is added to the track results. Following, the window is moved to the frame $t+1$ and the inference is executed. This operation is repeated until the frame $t-w$ is reached. By setting the sliding step to one frame, the algorithm is able to recover every occlusion whose duration is within the window size. Moreover, in each step all the associations identified in the inference result are introduced as new observations in the next sliding window $t+1$, incorporating inference information from the past window. In order to do this, we define a new probability term p_{IW} which is included in the likelihood factorization, and works analogously to the motion component p_M, explained in Eq. (3.13). This term enforces linear trajectories in the movement of the blobs, but in this case between consecutive windows.

For instance, consider the first window of a sequence of frames, starting in frame 1. Blobs falling in the first frame of this window do not have a previous blob from which to extract trajectory information. However, in the features contained in other frames of the window, the algorithm uses the information of the previous frame to infer the likelihood of their positions, assuming the trajectory modeled by p_M. Once the first window is calculated and the sliding window moves one frame forward, we can use the trajectories information calculated in the previous window to provide the features in the first frame with this information. This is modeled by the term p_{IW}, and it is defined as:

$$p_{IW}(Y_a, Y_b|X_b) = p_M(Y_a, Y_b|X_a = 1, X_b). \tag{3.24}$$

Notice that the variable X_a is already set to 1, because the corresponding association a was *activated* in the previous window.

3.4 Approximate Inference with Belief Propagation

Searching for an optimal vector $\mathbf{X}$ which maximizes the expression in Eq. (4.2) is, in general, NP-hard. We use the max-product algorithm to calculate an approximation of the MAP configuration of the vector of random variables, on a Markov Random Field [76] formed by the variables X_a, the observations Y_a, for all $a \in A$, and the potentials defined by Equations (3.5) to (4.15).

Figure 3.6 shows an example of a factor graph resulting from a window of three frames. In the left graph, boxes represent appearance factors, which correspond to Eqs. (3.5), and (3.6). In the center only pairwise factors are represented, and correspond to Eqs. (3.11), and (3.13). In the right figure we display only Prior and Occlusion factors, corresponding to Eqs (3.18), and (3.19)-(3.22). An interesting question at this point is the size of the graph, in terms of number of variables and factors. Supposing we have n blobs at each frame, the graph will have $w(w-1)n^2/2$ variables which is a huge number. However multiple tracking problems are inherently sparse: the vast majority of potential associations are quite unlikely, so that reasonable application-dependent heuristics (gating) can reduce them to a manageable size. We have used a simple distance threshold to discard associations which join blobs that are very far away from each other.

The running time for Belief Propagation is $\mathcal{O}(MN^k)$, where M is the number of random variables, N is the possible labels for each variable, and k is the size of the maximum clique (number of variables involved in a factor). In order to keep inference problems tractable, most of the authors limit the size of the cliques pair-wise, at the expense of expressive power in the variable dependencies.

In our case, the cliques are the sets of variables involved in each of the terms explained in section 3.2.1. Therefore, the bigger cliques correspond to the multiplicity constrains of Eq. (3.19) and (3.20), and their maximum size can be very high, depending on the amount of blobs existing in each frame

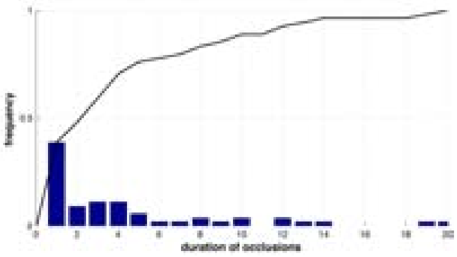


Figure 3.7: Histogram and accumulated histogram of occlusion durations. A window of size w is able to recover all occlusions with duration up to $w-2$.

of the window. To deal with the high dimensionality of these terms, based on the work of [83] , we take advantage of the high degree of sparseness of the constraint functions of Eq. (3.19) and (3.20), to transform the high order clique into several quadratic cliques, by adding extra variables. Analogously, we apply the same procedure to other high order functions, such as the occlusions factor of Equation (3.18). For the sake of completeness, we illustrate the reduction mechanism with a simple example in Chapter 2.3.

3.5 Experiments

In this work we have analyzed the quality of the tracking by evaluating the accuracy in terms of correct blob labeling, detection of occlusions, blobs merging and blobs splitting. We also study the system parameters configuration that permits reaching real-time running times, and we show how the drop on tracking accuracy under this *fast* set-up is not significant. All the experiments are conducted on real video sequences of vehicles at nighttime. These were recorded by a camera with a CMOS image sensor from Aptima Imaging$^{(TM)}$ of 752×480 pixels resolution (see [65] for more details). A lens, of 40° angular field of view makes the detection of distant taillights very challenging. For instance, a single taillight at 400 meters is imaged as a spot of 4 to 10 pixels. It is clear that a High Resolution camera, or a lens providing a narrower angular field would facilitate the task of detecting distant targets imaged now as tiny blobs. However, we foresee the need of using a low cost multiple purpose hardware, that can as well be applied to other driving assistance operations, like detection and recognition of road-signs, lane markings, and pedestrians.

Manually annotated ground truth data was used for training and evaluation of the tracking accuracy. The ground truth consists of blobs annotated with their corresponding track label, and contains 51 tracks, 8,919 blobs, 54 occlusions, 47 merges and 60 splits. Merging or splitting targets are annotated in the ground-truth as such when they belong to the same object. We have annotated 7 sequences of approximately 250 frames each. The training of the likelihood terms p_A, p_G, p_{SM} and p_O was conducted on 600 frames extracted from these 7 sequences. For the testing we have selected 5 of these sequences, and we have extracted 100 frames from each of them which are different from the frames selected for the training set. To avoid counting the same error multiple times, a miss-detection of a merging or splitting is not considered as an incorrect labeling.

For all the experiments we have set parameters λ_G and λ_{SM} to 0.5. Recall that these parameters weight the contribution of each term of the mixture of densities for the Geometry and Split & Merge distributions respectively. Other parameters like the length of the window or the number of iterations of the inference algorithm have a considerable impact in the results, and we need to choose them carefully. The length of the window establishes the duration of the occlusions that the algorithm will be able to recover. For instance, with a window of n frames, the algorithm is able to detect occlusions up to $n-2$

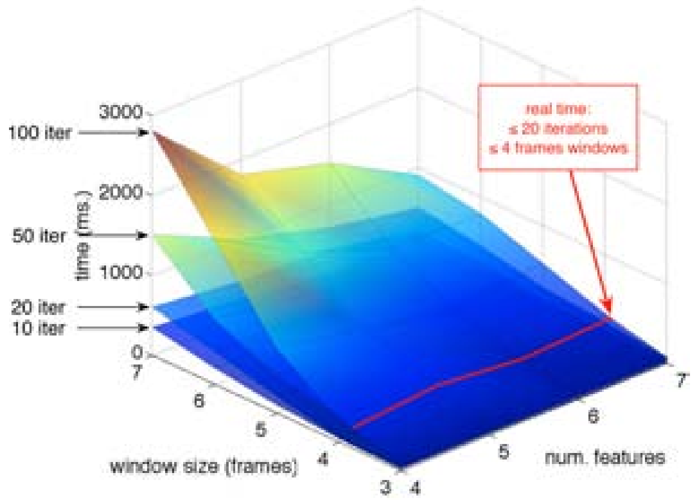


Figure 3.8: Shows the inference running time in milliseconds, as a function of the window size and the average number of blobs per frame. Each of the layers represents a configuration of the inference algorithm to run with 10, 20, 50 and 100 iterations. The red line shows the boundary where real-time requirements are satisfied. We limit the window size to 4 frames, with a maximum of 20 iterations in the inference algorithm.

frames of duration. We have counted the number of ground truth occlusions for different durations, as shown in Figure 3.7. With a window of 6 frames we should be able to recover occlusions up to 4 frames of duration, which is around 70% of the total occlusions in the ground-truth.

We first present quantitative results on the optimal parameters in terms of accuracy of the tracking in Table 3.1 . We choose a window of 6 frames and 500 iterations of the inference algorithm. The percentage of correct labeling is over 90%, which illustrates the suitability of our method for tracking feature-less tiny targets. The worst results were obtained for the detection of merging targets. This is due to the great difficulty when distinguishing a target which is merging or splitting from a target which is being occluded, or reappearing. For example, if two targets are very far from the camera and close to each other, and in the next frame there is only one target, it is very difficult, even for a human observer, to determine if the targets merged, or one of them has been occluded.

Table 3.1: Tracking evaluation for sequences A to E

	Sequences					
metrics in %	A	B	C	D	E	Mean
correct labeling	92	90	94	91	89	91.2
occlusions	63	60	68	72	66	65.8
mergings	38	62	53	48	39	48
splittings	71	59	65	58	63	63.2

The occlusion recovery performs well. The fact that 63.2% of occlusions are recovered should be analyzed taking into account the number of occlusions which the method is able to treat with a window length of 6 frames, which is around 70%. Thus, the percentage of occlusions which are well-treated among the ones which are tractable is 90%. This is clearly seen in Figure 3.9, where the red

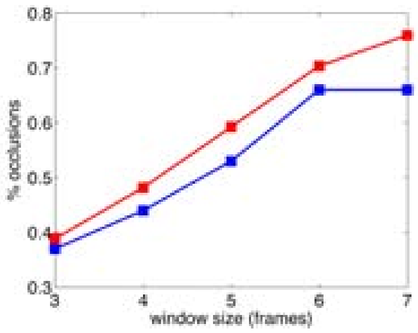


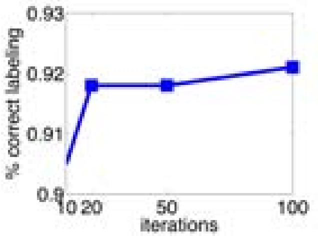
Figure 3.9: In red, percentage of ground-truth occlusions that could be recovered for each window length. In blue, the percentage of occlusions correctly detected.

line indicates the limit of *detectable* occlusions given by the window size, and the blue line shows the actual result. The less impressive results correspond to blobs merging and splitting detection with an average of 45% and 63.2% respectively. This events are very difficult to detect. Since we are not using any detector to gather our targets of interest, we use a threshold on the image brightness produced by the lamps and reflections, which is a very simple and fast technique to remove noisy observations. Specifically, its value is set to one third of the maximum possible gray level of the images. Consider a scenario where several tiny far away blobs are tracked simultaneously. These tend to disappear and reappear very frequently because their gray values lie near the segmentation threshold and these targets present intensities which flicker constantly due to the camera motion, and the unevenness of the road. When this happens, it is very hard to distinguish if the blob is actually disappearing or that two blobs have merged because they are very close. In the case of a blob reappearing, miss-detections of the splitting events also occur. Moreover, two blobs merging seldom occur in our video sequences, and the lack of examples of such events in the training data clearly hampers the proper modeling of their likelihood.

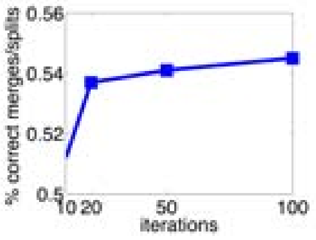
In order to meet the real-time requirements of the application, we need to reduce the inference running time that is the bottle-neck of the algorithm. In order to do this we choose the parameters which better balance the accuracy results and the running times. First, we perform an experiment setting the number of iterations to 500. Figure 3.9 shows the relationship between the window length and the occlusions that are correctly detected with this configuration. We need to make a compromise between the length of the window, which dramatically influences the computation time, and the number of occlusions which can be recovered. Choosing a window size of 5 frames, the method is able to recover every occlusion up to 3 frames of duration, or 60% of all occlusions. The occlusion recovery seems a reasonable criteria to establish the window length, since this parameter hardly influences any other aspect in the algorithm validation. However, increasing the window size results in the exponential growth of the factor graph, and thus, it is necessary to increase the number of iterations to achieve good results in the overall tracking accuracy.

Following, once the desired window size is established, we perform a set of experiments to show the evolution of the results depending on the number of iterations of the inference algorithm. Figure 3.10 shows results on blob labeling and blobs merging and splitting detection. It can be seen that the performance over 20 iterations does not increase significantly. With 20 iterations we are able to correctly label almost 92% and 54% of blobs merging and splitting.

One would expect the tracking accuracy to grow linearly with the number of iterations. This is not the case, as can be seen in Figure 3.10. We can find an explanation in the size of the factor graph, since the number of variables and cliques determines the number of iterations needed to achieve a sat-



(a)



(b)

Figure 3.10: a) Percentage of correct blob labeling. b) Percentage of blobs merging and splitting correctly detected. The inference in each experiment is set to 10, 20, 50 and 100 iterations

Table 3.2: Tracking results with real-time running times

Window size / Iterations	3 / 10	4 / 20	5 / 50	6 / 100
Correct Blob Labeling	91.8 %	92.2 %	93 %	91.2 %
Correct Merge&Spits	51.4 %	52.8 %	52.1 %	50.5 %

isfactory optimization. The bigger the factor graph, the more iterations the algorithm needs to reach a similar result. In order to exemplify this behavior we have selected pairs of window sizes and iterations and evaluated the algorithm accuracy for each of them. The pairs {window length / iterations} selected are: 3/10, 4/20, 5/50, 6/100. Table 3.2 shows the percentages of correct blob labeling and blobs merging/splitting for each of the pairs of settings. Note that the differences between the four tests are minimum (from 1% to 3%). This suggests that high accuracy results can be obtained with a few number of iterations (high running-times) as long as the size of the window is kept small, and thus, the size of the factor graph. The drawback of reducing the number of frames of the window is the amount of occlusions that the algorithm is able to detect. Despite this disadvantage, note that the method will scale well with the capabilities of the hardware where it is implemented, by adjusting the parameters (iterations and window length) to achieve fast running times at the expense of miss-detecting some occlusions.

A tracking system mounted in a vehicle is expected to run in real-time. Even though message-passing inference algorithms are not reputed for their speed, we find important to prove that our method is able to reach real-time requirements under a specific set-up. We have investigated the impact in the performance of two parameters: window size and number of iterations; following, we show that the method is able to reach sufficient running times, while preserving a good quality tracking.

For the MAP inference we have used the C++ implementation of the max-sum algorithm from the libDAI library [70]. Figure 3.8 shows the relationship between the window size, the inference iterations, and the average number of blobs per frame, respect to the total running time of the inference algorithm, in milliseconds. The red line delimits the parameter space where the real-time requirements are satisfied in the above mentioned machine, which is $\leq$ 60 milliseconds per frame, at 15 fps. The number of blobs is not critical, and usually more blobs does not necessarily mean higher running times. After heuristically reducing the number of potential associations based on the position of the blobs, the influence on the running time is rather dependent on the positions of the blobs in the image. For instance, when frames present blobs concentrated in reduced image areas, the *gating* mechanism based on a distance threshold cannot reduce the number of pairwise functions (See section 3.4).

Figure 3.13 shows four frames of the resulting tracking applied to real video sequences. The colored lines represent the different tracks, and the circles indicate the position of the blob centroid in a timespan of 10 frames. In the right column of the figure we show an alternative visualization that contains all the frames of the sequence. The horizontal yellow stripe indicates the window of 10 frames that is represented in the left column. Note that vertically aligned blob tracks cannot be distinguished clearly under this representation (e.g splitting of blue track in top row). Examples of a blob splitting can be seen in the first (top) and second images. The third image shows a car near to the camera that originates multiple blobs due to light reflections and the over-segmentation of the vehicle, producing several blobs' splitting and merging. A specially interesting case is the one shown in he last frame, where a distant vehicle identified by its two rear-lights is well tracked for a long time (red and sky-blue tracks). Recall that we are mainly interested in tracking small blobs originated by far away targets, since the tracking is aimed at assisting a blob classifier for headlight control that fails when the targets present almost inexistent appearance features. A classic Kalman based tracking algorithm would not be able to track two close small blobs like in the example, since the mounted camera movements would make extremely difficult to associate a new observation to a stored track.

We have constructed a web-page [24] where videos of 5 sequences with superimposed tracks can be viewed.

Finally, we provide two experiments to show how our tracking algorithm improves the performance of the classifier applied independently to each frame. We use exactly the same classifier than in [65]. In this work, Real-AdaBoost was used to obtain strong linear classifiers, which output a negative value for a blob classified as *non-vehicle*, and a positive value for *vehicle*. In order to integrate the tracking process in the classification we use the *hysteresis process* of [66], consisting of accumulating the classifier scores on the tracked target along time. Note that the blob classifiers were trained with video sequences different than the ones used for the test.

For the first experiment with the classifier we have augmented the number of frames and annotated blobs with respect to the tracking. The new testing set consists of 2,000 frames containing 2,151 vehicle blobs and 8,947 non-vehicle ones. We expect the tracker to help the most on small blobs (area below than 10 pixels). The performance of the classifiers is shown in Fig. 3.11, providing a comparison of the classifier without using tracking and using tracking to incorporate the above mentioned hysteresis process per blob. The classification performance for small blobs coming from vehicles increases around 15% in average, and around 5% for those blobs coming from non-vehicles. One may wonder whether this increase is significant. Yes, because 77% of the total amount of blobs in the sequences have an area below 10 pixels, which affirms the importance of the improvement.

In order to assess the relevance of the wrongly classified vehicle blobs, we have devised a second experiment where the system latency is computed when using tracking. For instance, the first time a small vehicle blob appears in a frame, due to the hysteresis process, the system may require a few frames more before providing a stable classification result for it (*vehicle* or *non-vehicle*). During such frames those small blobs account as wrongly classified vehicles in Fig. 3.11. However, if they are recognized in less than five frames, the overall system would behave well. The reason is that the image processing works at 15 frames per second and, thus, five frames correspond to 333 ms, which still leaves 166 ms to the headlights actuators for completing a total operating cycle of half a second. Figure 3.12 shows the results of this experiment performed on the annotated testing tracks. Note how the latency for vehicle blobs is low for small blobs and negligible for the rest.

3.6 Conclusions

We have shown that many-to-many feature matching can be applied to solve the problem of multiple target tracking, in the presence of target splits, merges and occlusions, obtaining high accuracy in real video sequences. We have developed a probabilistic model, in which the densities representing the application knowledge have been learned from training data. Tracking bright spots at night is known to be very challenging, especially for small targets whose images have an area of less than 10 pixels. Our method is able to correctly track an average of 90% of such small blobs. Finally, we have demonstrated that our tracking significantly improves the previous classification results without hampering its real-

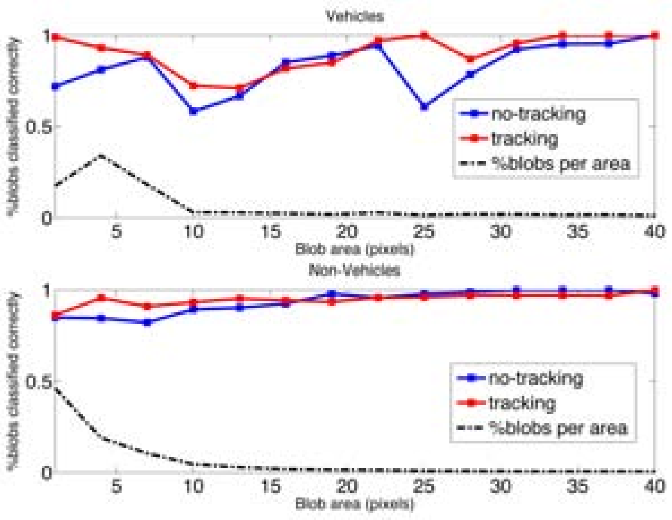


Figure 3.11: Comparison of the classification accuracy of the original classifier with no tracking, and the classifier with our tracking, for both vehicle blobs and non-vehicle blobs. The dotted line represents the percentage of blobs for each area. Small blobs (1-10 pixels) cover around 77% of the total number of blobs.

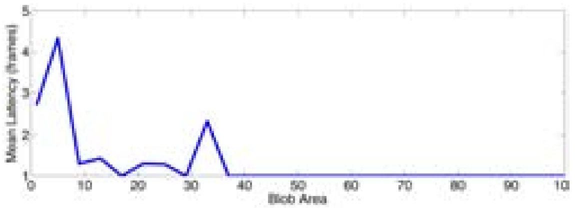


Figure 3.12: Latency of the classifier working with our tracking, related to the size (in pixels) of the vehicle blob being classified. For blobs bigger than 9 pixels, the response of the classifier is immediate (between 1 and 2 frames). For tiny blobs the latency is sufficiently small, being its maximum around 5 frames.

time requirements.

The main advantage of our method is its ability to encode complex relationships between the target characteristics, resulting in a flexible yet powerful model. We have introduced a novel explicit handling of occlusions, merges, and splits, creating continuous tracks of multiple targets. In IHC applications, this is necessary to extract multiple features from a blob along different frames, in order to improve the classification of difficult targets. However, the method can be easily extended to generic tracking applications. We have shown how the combination of our tracker with previously developed classifiers allows to improve the overall vehicle detection performance of the system, specially for distant vehicles.

Avenues for future research include: Merges and splits failures can be solved by increasing the amount of training data, and modeling a probability density which better suits the target behavior. In addition, blob tracking will allow to incorporate motion features as well as spatio-temporal appearance in the blob classification stage of the system.

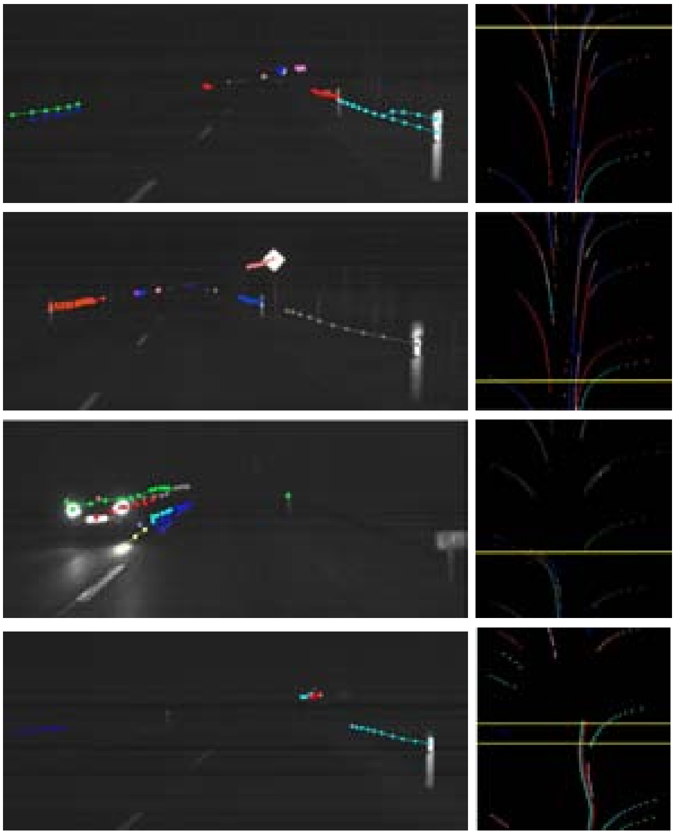


Figure 3.13: Examples of resulting tracks obtained by our method. Each color indicates a different track identity. First column contains example frames, where, for the sake of visibility track lengths have been limited to the last 10 frames. Second column contains the complete tracks, represented in a plane, x position against time, increasing upwards A yellow horizontal stripe represents the time span in which the left snapshot is taken. First and second frames contain distant small targets and splittings of close road-poles. The third shows a close vehicle generating multiple tracks due to over-segmentation and reflections. Fourth frame contains a long track of a pair of distant taillights. Better viewed in color.

Chapter 4

Data Association

4.1 Introduction

In the previous chapter we have proposed a method to build consistent tracks within a window of frames, but we did not mention how to tackle the problem of data association. That is, how to solve the ambiguities on track identities that can arise due to targets merging and splitting as well as due to inexact and noisy observations of targets. In this chapter we generalise the proposed method to a wider range of tracking applications, at the same time that we approach the problem of data association and *long* track generation. We propose a Markov random field specifically designed to solve the problem of data association, which we call Hypothesis Graph. We first present a general overview of the various problems one can find when mapping targets to observations in the context of multiple target tracking. Then we present our method and perform experiments in several of those variate scenarios.

In the context of multiple target detection and tracking the following definitions will help us to state the goal. A target or object is some real moving entity, imaged in a video sequence, that we want to follow in order to analyze its motion for some purpose (like people and vehicles for surveillance [10], particles in a turbulent flow for its characterization, live micro-organisms for lineage studies [64], [61], or insects for behaviour studies [55]. An observation or measurement is the detection of an object as it appears in an image. Note that a single observation can actually result from several objects whose observations overlap.

Data association is the process of relating objects to observations. In the absence of merges/splits, each target corresponds to a unique observation, and therefore targets are unambiguously identified as long as the track construction is correct. In presence of occlusions, mapping targets and observations is a difficult problem to solve. Moreover, tracking multiple objects implies multiple object interactions and mapping between observations, which is costly to solve optimally.

There are many works on visual multiple target tracking. Only some of them try to maintain identities in addition to build tracks and, being the most interesting type of result, we will focus on them in the following review. The usual classification of past works we have found is according to the strategy or the techniques employed for data association, that is, whether they are based on multiple hypothesis tracking (MHT) [81], joint probabilistic density association (JPDA) [94], particle filtering [47], integer linear programming, graph algorithms (like min-cut and set cover), inference on Bayesian networks [74], etc. MHT and JPDA are the most widely used approaches, but present some drawbacks. As MHT suffers from state space explosion when applied to real videos, JPDA assumes a fixed number of targets, and only considers measurements in the current frame step.

	Targets	
	t	$t+1$
(1). New targets may appear	0	1
(2). Targets can disappear	1	0
(3). Regular case	1	1
(4). A target can become n (e.g cell mitosis)	1	n
(5). m targets can become one (e.g cell fusion)	m	1

Table 4.1: The five possible scenarios regarding the evolution of a track along time.

Another relevant categorization criterion is whether the tracking is batch [107], [74] or online [10], that is, tracks (and identities) are resolved once the whole sequence is available or it is done as each frame is ready. Clearly, the batch strategy has the advantage of working with all the data along time and it makes sense to use it in problems which do not require an online answer like live cell tracking or turbulent flow analysis. However, in other applications a fast answer is needed to make a decision, like in surveillance or headlights control [89].

We believe that a better understanding of the state of the art can be grasped on the basis of the actual multiple target tracking problem being solved in each case. We mean that by just slightly changing the way the targets or the observations are assumed to evolve along time, or the (often implicit) relationships between a target and its observation (how may it appear in the image), one gets a very different problem to solve. This in turn determines the kind of methods to use. Just as an example, if targets are perfectly segmented (no false positives or negatives, each target gives rise to exactly one observation and to each observation corresponds one target) we have a problem of one-to-one data association which can be solved by the Hungarian method [54]. However, if one target may be over-segmented into several regions and we want to be aware of it, the problem is quite different.

The different tracking scenarios can vary from the simplest case (one target is one measure, and one measure is one target), to more complicated situations. In the most general case, a target can produce 0,1, or more measurements, and one measurement can be produced by 0, 1 or many targets. Table 4.1 presents different scenarios regarding the evolution of targets in time. In order to unequivocally define our tracking application, we should take into account the behavior of our targets in time and their relationships with the image measurements. Table 4.2 presents these relationships for the different sequences we provide in the experiments: Synthetic flow in FIg. 4.1, Vehicle headlights in Fig 4.2 and Bacteria growth in Fig. 4.3.

Instead of designing a tracking method for a specific instance of a problem, our goal is to provide a generic multiple target tracking algorithm that can handle as many of those situations within a unique framework.

4.1.1 Overview of the Approach

We propose a two-component algorithm that outputs the complete trajectories of each of the targets in a video sequence. The first component handles the creation of tracklets within a local window of frames, similarly to the process of track creation explained in chapter 3. The second component performs tracklet linking and data association, which is the main contribution of this chapter. The tracklet creation is based on examining a window of a few frames, and establishing correspondences between the observations in each of these images. We define a tracklet as an ordered list of observations of the same target, between frames j and l, generated by a series of one-to-one associations between

	Target Scenario	Targets	Obs.
Synthetic Flow	(1),(2),(3)	1	1
		n	1
Headlights	(1),(2),(3),(4)	1	0
		1	1
		1	n
		m	1
Bacteria	(3),(4)	1	1

Table 4.2: Definition of our application tracking problem. Relationship target-observation, combined with the evolution in time of the targets.

observations in consecutive frames.

This Chapter as well as the previous, is focused on tracking multiple targets which seldom produce any appearance information, or such attributes are useless because every target looks the same (See Figures 4.1-4.3, for samples of the applications' frames). This is an important handicap when addressing a problem of data association. We overcome this disadvantage by exploiting instead the target's motion information, as well as assuming certain *rigidity* on the movement of the targets between contiguous frames. Graph Matching provides a perfect tool to exploit this knowledge.

The Track Identity Linking step has two main goals. First, estimating the identity of the target of those observations presenting certain uncertainty or ambiguity (data association). Second, linking tracklets of different windows which belong to the same target. We simultaneously solve these two problems by modeling the tracklet identity ambiguities in what we call an Hypothesis Graph, and then inferring the most likely hypothesis of track-target correspondences.

4.2 Construction of Local Tracklets

Analogously to the previous Chapter, we present a probabilistic-based graph matching approach to construct target tracklets in a window of frames. However, in this Section we do not want to restrict our problem to the one of Headlight Control. To do so, we remove some of the constrains introduced in Chapter 3 and we do not explicitly model any assumption that does not generalise to several tracking scenarios (e.g headlights are aligned horizontally). Lets recall the formulation introduced in Chapter 3, and let w be the number of frames in a certain temporal window of the video sequence. We denote by $I_1, I_2, ...I_w$ the different frames within this window. Each frame contains a set of zero or more observations, indexed by $p, q, ...$. An association a is an ordered pair of observations from the same target, but at different frames. Let A be the set of all such associations,

$$A = \{a = (p, q) | p \in I_i, q \in I_j, 1 \leq i < j \leq w\}, \tag{4.1}$$

where $a, b, ...$ index the elements of A, so that we can denote all pairs of association without repeated combinations as $(a, b), a < b$. Let $\mathbf{X} = (...X_a...)$ be the vector of binary variables, one per association, where $X_a = 1$ if the corresponding association a exists, and zero otherwise. In the same way, the vector of all measurements is denoted by $\mathbf{Y} = (...Y_a...)$, where each association $a = (p, q)$ is

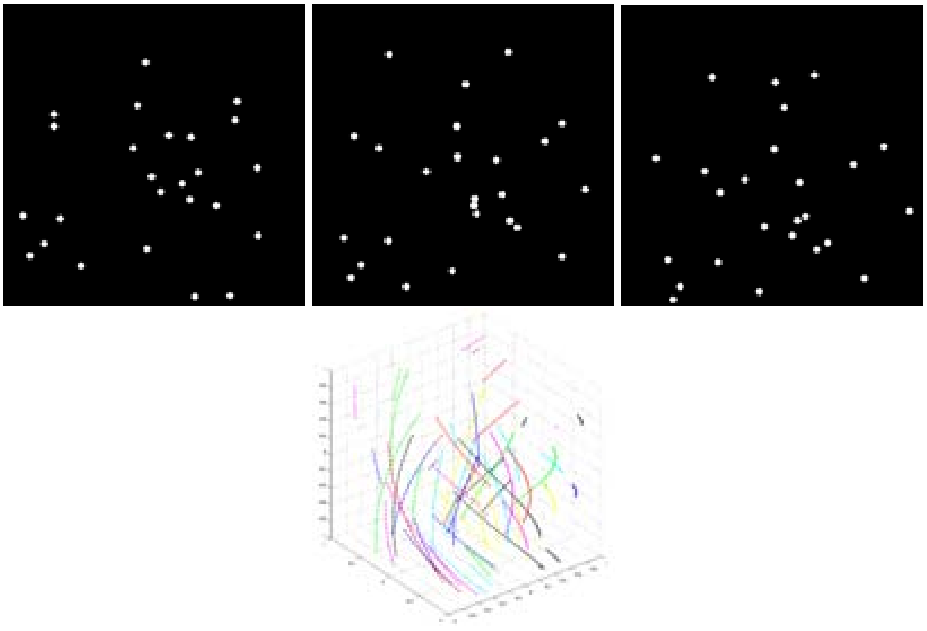


Figure 4.1: Sample frames of particles in a synthetic helical flow and flow lines. Each particle is always seen as a blob and one blob corresponds to one or several particles, if they overlap. Blobs merge and split but don't get occluded by other things.

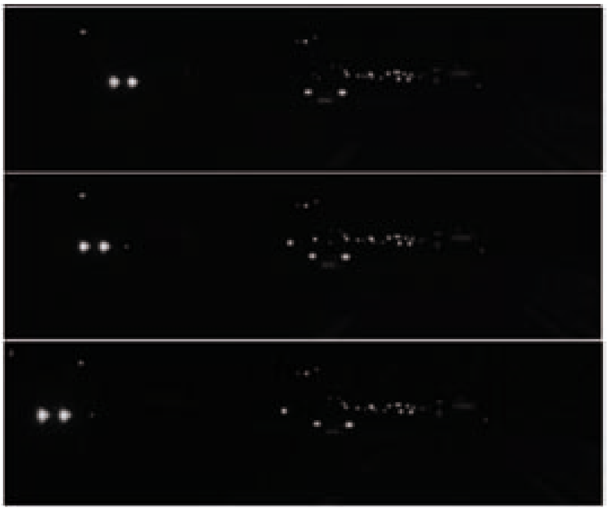


Figure 4.2: Successive frames from a night driving video sequence recorded by an on-board camera. One blob may correspond to two far away light sources or reflections. Blobs merge, split and get occluded by other vehicles, trees and fences.

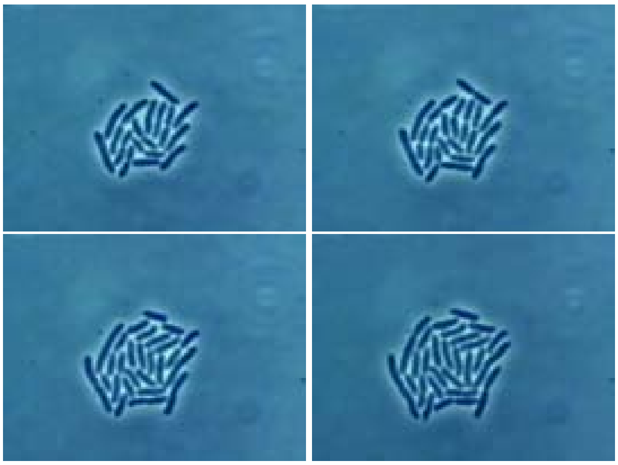


Figure 4.3: Sample frames of the bacteria growth video sequence. Every bacterium will be correctly segmented and to each region will correspond a single bacterium (perfect detection and no overlapping). Some bacteria divide (splits) while others just grow.

represented by $Y_a = [p_x, p_y, q_x, q_y]$. Thus, each association is attributed with the spatial coordinates of its origin and destination points. Although, other properties may be also considered, like size, shape, or intensity measures. We only consider spatial coordinates to keep our formulation as generic as possible.

Our goal is to find the most likely configuration of the set $\mathbf{X}$ of association states, given the set of all measurements $\mathbf{Y}$. This is, to find the maximum a posteriori estimation,

$$\mathbf{X}^* = \arg\max_{\mathbf{X}} \mathbf{p}(\mathbf{X}|\mathbf{Y}). \tag{4.2}$$

4.2.1 Observation Model

To build a generic observation model, we start by enumerating the set of premises that every Multiple Target Tracking scenario should satisfy:

- Measurements belonging to the same track can not move too far between consecutive frames.
- Targets follow fairly smooth trajectories with constant speed between consecutive frames.
- Close targets in colliding directions are likely to merge.
- A target entrance and departure strongly depends on its location in the image.

We encode the first three assumptions in the following likelihood factorization. The fourth constraint will be modeled in the data association step, as we will explain later.

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{a\in A} p_A(Y_a|X_a) \cdot \prod_{(a,b)\in N} p_N(Y_a, Y_b|X_a, X_b)$$

The first term models the likelihood of an association being active or inactive, depending on the location of the two features (p, q) involved in each association $a \in A$. The second term, defined over the set N of pairs of associations, is the likelihood of two associations existing simultaneously. This pairwise terms smooths the object motion (speed and direction) along several frames, and also models the likelihood of merging and splitting events. See Figure 4.4, were the sets of possible association sets are shown.

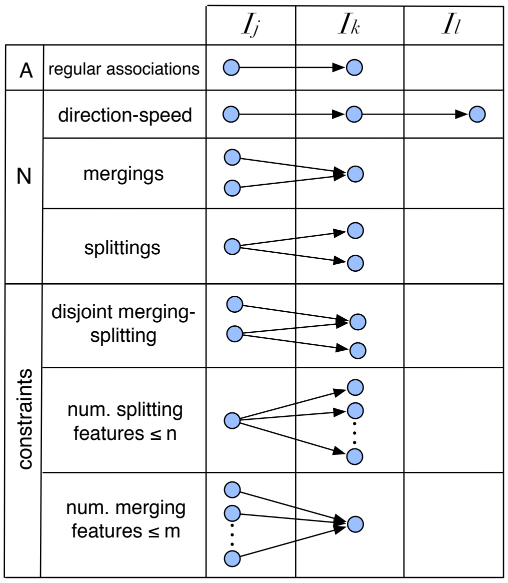


Figure 4.4: Sets of associations involved in the likelihood (A, N) and prior (constraints).

Following we present the probabilistic modeling of each of the likelihood terms based on the previously stated assumptions.

Displacement: The likelihood of a single association is defined as:

$$p_A(Y_a|X_a) = \mathcal{N}(|\mathbf{v}_{pq}|, \mu_A, \sigma_A), \tag{4.3}$$

where $\mathcal{N}$ is as a Normal distribution, defined on the norm of the vector $\mathbf{v}_{pq}$, or the target velocity.In order to define our observation model as generic as possible we do not establish any correlation between the movement of a target and its appearance or image position. Although in the context of a specific application it would be convenient to apply constraints more complex and discriminative.
The pairwise term of the likelihood is, in turn, factorized in three different terms: p_L, p_M and p_S. The first penalizes sudden changes on speed and direction, and the other two model the likelihood of two targets merging and splitting.

Linear Trajectories and Speed: The set of pairs of associations related to the trajectory of the tracks is defined as

$$N_L = \{(a,b) \in N | a = (p,q), b = (q,r)\}. \tag{4.4}$$

and its pairwise likelihood is defined as a mixture of densities,

$$\begin{aligned} p_L(Y_a, Y_b|X_a, X_b) &= \alpha\mathcal{N}(\widehat{\mathbf{v}_{pq}\mathbf{v}_{qr}}, \mu_{dir}, \sigma_{dir}) \\ &+ (1-\alpha)\mathcal{N}(|\mathbf{v}_{pq}| - |\mathbf{v}_{qr}|, \mu_{vel}, \sigma_{vel}). \end{aligned} \tag{4.5}$$

where the parameter $\alpha \in [0,1]$ weights the contribution of each component. The first Normal distribution models inter-frame target direction changes in terms of angles between consecutive motion vectors. The second density encodes the changes in target velocity, which are expected to be near zero, always

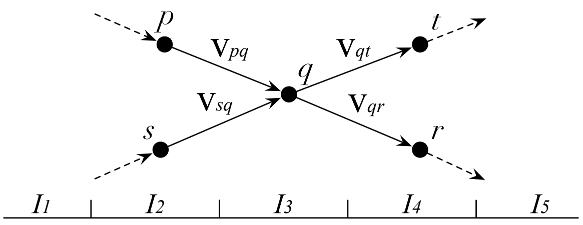


Figure 4.5: Target motion vectors involved in a two targets merging and splitting

between consecutive frames. Figure 4.5 shows a simple example of target motion vectors.

Merging & Splitting: The following densities model the probability of two features merging, or one feature splitting in two. Their respective sets of pairs of associations are:

$$\begin{aligned} N_M =& \{(a,b) \in N | a = (p,q), b = (s,q)\}. \\ N_S =& \{(a,b) \in N | a = (q,t), b = (q,r)\}. \end{aligned} \tag{4.6}$$

Their pairwise densities define a correlation on the direction and distance between merging or splitting features. In this case, no assumptions can be made about the data following a Gaussian distribution. Instead, we use a Kernel Density Estimator to model the functions $\hat{f_M}$, and $\hat{f_S}$ from training data.

$$p_M(Y_a, Y_b | X_a, X_b) = \hat{f_M}(\widehat{\mathbf{v}_{pq}\mathbf{v}_{sq}}, |\overrightarrow{ps}|) \tag{4.7}$$

$$p_S(Y_a, Y_b | X_a, X_b) = \hat{f_S}(\widehat{\mathbf{v}_{qt}\mathbf{v}_{qr}}, |\overrightarrow{tr}|) \tag{4.8}$$

Notice that in contrast to Chapter 3, in Eq. 3.15, we do not include knowledge about the specific application in the observation model (e.g horizontal alignment of merging/splitting points).

4.2.2 Hard Constraints

Analogously to the prior term defined in the previous Chapter, we include a set of constraints on the labels taken by sets of correspondences in which we want to impose a certain condition. Again, we limit the number of maximum correspondences involved in a split or merging event, as in Eq. 3.19 and 3.20. However, this time we do not want to set this parameter equal to two (case of headlights) but let the user establish a value suitable for the tracking application. For instance, if the user chooses $w = 1$, the matching becomes one-to-one, and no mergings or splittings are allowed during the tracklet generation.

Finally, we also add a constraint to avoid bizarre tracklet configurations, like a merging/splitting spanning in more than two frames, or a merge mixing with a split and vice versa. See Figure 4.4 (disjoint merging-splitting). This was also introduced in Eq. 3.21 and 3.22.

4.3 Online Data Association

In the following section we introduce the major contribution of this Chapter, consisting of a probabilistic method to address the data association problem. Given a set of tracklets generated in the previous step, the goal is to find the most probable one-to-one correspondences between tracklets and track identities. Some of the tracklet identities can be unambiguously determined if they do not interact with any other tracklet along their lifetime. Unfortunately, in a context with a great amount of targets it is less likely to find tracklets which do not interfere with each other.

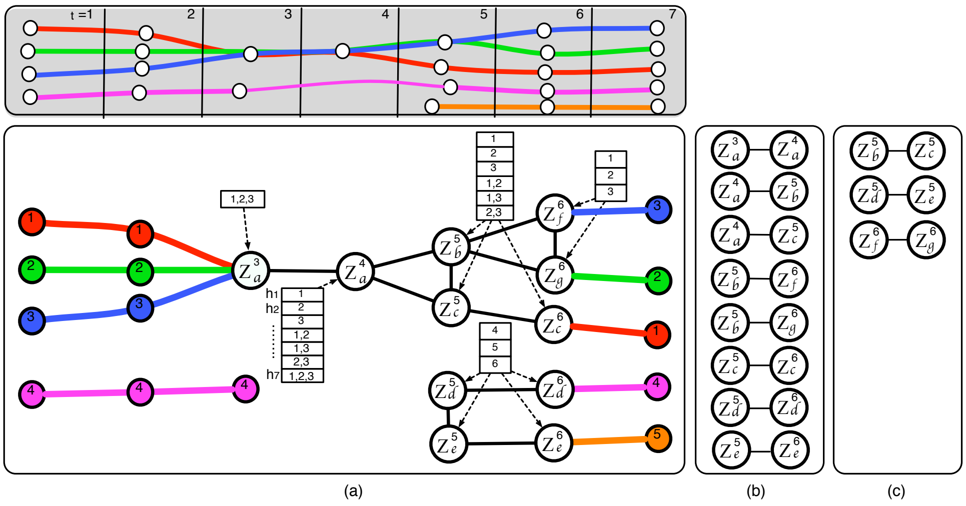


Figure 4.6: Example of Hypothesis Graph, in the presence of several ambiguous events. The (a) top shows 7 frames with white circles representing the observations and colored lines indicating the track each target follows. The dotted segment represents an occlusion. In (a) bottom, the Hypothesis Graph is represented. A white circle denotes a vertex of the Hypothesis Graph (an observation whose track label is unknown). The tables associated with such vertices show the list of hypothesis at each time step. In (b) the label Smoothing Dependencies are shown, and in (c) the set of Identity Coherence dependencies. Best viewed in color.

Lets assume that a set of tracklets $\{t_1, t_2, ..., t_n\}$ is constructed up to frame s. Each tracklet is, in turn, a list of contiguous observations between two frames. In the present context, an observation or measurement is defined simply by the feature centroid in image coordinates, as $o_a \in O$. Thus, a tracklet a between two arbitrary frames I_i,I_k, is formally denoted as $t_a^{i:k} = \{o_a^i, o_a^{i+1}, ..., o_a^k\}$, being $i < k \leq s$. However, the measurements used to find the target identities are mainly related with the movement of the targets. We say that $M_a^k = o_a^k - o_a^{k-1}$ is the motion vector of the observation $o_a \in I_k$.

Following, we formally define the Hypothesis Graph, and introduce a probabilistic method to obtain the most likely hypothesis of track labels.

4.3.1 Hypothesis Graph

We define an Hypothesis Graph as an undirected graph $G = (V, E)$ over sets of vertices $V \subset O$ that represent *ambiguous* observations. The set E of graph edges contains pairs (a, b) of node indexes, and denotes dependency relationships between the graph nodes. We identify two types of dependencies: Label Smoothing, and Identity Coherence, respectively grouped in sets $E_{ls}, E_{ic} \in E$, as we will explain in Section 4.3.2. Figure 4.6 shows an example of Hypothesis Graph.

We say an observation $o_a^k \in V$, if any of the following statements is true:

- The measurement is the result of a splitting.
- The measurement comes from multiple tracks.
- The observation o_a^{k-1} was also ambiguous.
- It is the first measurement of the tracklet, and exist occluded tracklets in past frames which are candidates to be recovered.

Let $\mathbf{Z} = \{Z_1, Z_2, ..., Z_n\}$ be a vector of multidimensional random variables, each corresponding to a vertex from the Hypothesis Graph, and $\mathbf{M}$ be the set of all motion vectors. Each variable realization Z indexes one of the possible hypothesis present in its associated ambiguous observation. An Hypothesis h is defined as a set of an arbitrary number of track labels $\{l_1, l_2, ...\}$. The goal is to label each ambiguous variable with the most probable hypothesis.

We propose a similar probabilistic approach to the one presented in Section 4.2. The set of most likely hypothesis for each of the ambiguous measurements maximizes the posterior probability,

$$P(\mathbf{Z}|\mathbf{M}) = P(\mathbf{M}|\mathbf{Z})P(\mathbf{Z}) \tag{4.9}$$

where the likelihood function takes the form:

$$P(\mathbf{M}|\mathbf{Z}) = \prod_{o_a^k \in V} P(M_a^k|Z_a^k) \prod_{o_a^k, o_a^{k+1} \in E_{ls}} P(M_a^k|Z_a^k, Z_a^{k+1}). \tag{4.10}$$

The first likelihood term models the probability of the measurement M_a^k belonging to a track listed in any available hypothesis of Z_a^k. The second, imposes a smoothing constraint on the track label values between two contiguous observations, as well as modeling the probability of a track departure from an hypothesis. The smoothing constraint will be introduced in the next section. The first component is defined as:

$$P(M_a^k|Z_a^k) = \prod_{h \in Z_a^k} \prod_{l \in h} P(M_a^k|l) \tag{4.11}$$

where,

$$\begin{aligned} p(M_a^k|l) = {} & \beta\mathcal{N}(|M_a^k - M^j(l)|; \mu_1(m), \sigma_1(m)) + \\ & (1-\beta)\mathcal{N}(\widehat{M_a^k, M^j}(l); \mu_2(m), \sigma_2(m)). \end{aligned} \tag{4.12}$$

The measurement $M^j(l)$ denotes the motion vector of the last detected observation which could be labeled with complete certainty as belonging to track l. The term encourages the selection of hypothetic tracks, whose motion vectors are similar to the original non-ambiguous track, in terms of direction and speed. The index j denotes the frame where the observation was detected, and $m = k - j$ refers to the age of the original measurement. The Normal distribution that governs the hypothesis likelihood will widen and become less restrictive, depending on the frames elapsed since the measurement was last observed. This permits certain variation on the observations (motion) of occluded targets that are recovered later as soon as the target produces observations again. The parameter β weights the contribution of the speed or direction on the hypothesis likelihood.

4.3.2 Pairwise Potentials

We define two types of dependencies:

Label Smoothing: The label smoothing dependency favors a consistent labeling of ambiguous observations along time, and encourages the generation of long tracks by smoothing the label value between contiguous observations within a tracklet (See Figure 4.6.(b)) Bring to mind the formulation of the likelihood of a measurement belonging to an hypothesis in Eq. (4.10). The Smoothing term is then defined as

$$P(M_a^k|Z_a^k = h_1, Z_a^{k+1} = h_2) = \tag{4.13}$$
$$= \begin{cases} 1 & \text{if } h_1 = h_2 \\ P(o_a^k|h_1,h_2)^{|h_1|-|h_2|} & \text{if } |h_1| > |h_2| \\ 0 & \text{otherwise.} \end{cases}$$

This equation enforces continuity on the track labels between contiguous observations from different frames. Note that given two hypothesis h_1, h_2 from two connected nodes, we enforce the same track labels to appear in both hypothesis (smoothing). If the newest hypothesis has fewer number of track labels, we model the probability of a track disappearing with the term $P(o_a^k|h_1, h_2)$, which is a normal distribution constructed around the assumption that tracks close to the image borders are likely to disappear. Any other configuration is considered incoherent and forbidden.

Identity Coherence: The Identity Coherence is responsible of ensuring that two or more observations in the same frame, whose hypothesis realizations can be contradictory (e.g contain the same identity), will be coherent after the inference. Since it is independent of the observations, it models the prior of the probability of Eq.(4.9):

$$P(\mathbf{Z}) = \prod_{o_a^k, o_b^k \in E_{ic}} P(Z_a^k, Z_b^k), \tag{4.14}$$

where

$$P(Z_a^k = h_1, Z_b^k = h_2) = \begin{cases} 1 & \text{if } h_1 \cap h_2 = \emptyset \\ 0 & \text{otherwise.} \end{cases}$$

These pairwise terms are represented in Figure 4.6.(c). Notice that in some tracking applications this constraint does not exist; therefore, we allow this restriction to be dropped depending on the tracking application. Furthermore, the Identity Coherence restriction can be selectively placed to distinguish both cases: grouping measurements of the same target, and mutual-occlusions of targets.

4.3.3 Handling of Long Occlusions

The last important detail to complete the method formulation is the handling of long occlusions. We address this issue with a very intuitive assumption: Every observation which starts a new tracklet is a candidate to contain the identity of track which ceased being observed during the last L frames. Lets denote as T_{occ} the list of these track identities, and let Z_a^k be the observation of a new tracklet a detected in frame k. Being N the number of tracks identified up to the present frame, the set of realizations (hypothesis) of Z_a^k is defined as

$$Z_a^k = \{T_{occ} \cup \{N+1\}\}. \tag{4.15}$$

The Eq. (4.11) is then slightly modified to include the likelihood of a new track appearing in the scene:

$$P(M_a^k|Z_a^k) = \begin{cases} \prod_{h \in Z_a^k} \prod_{l \in h} P(M_a^k|l) & \text{if } l \in T_{occ} \\ P_{new}(M_a^k) & \text{if } l = N+1 \end{cases} \tag{4.16}$$

Note that the distribution stays unchanged if the realization of Z_a^k suggests the recovery of an occluded track in T_{occ}. Otherwise, the density P_{new} indicates the probability of detecting a new track. The distribution P_{new} is assumed normal, defined on the minimum distance between the feature centroid and the nearest image border. Analogously to the departures, the entrance of targets is more likely in the image borders.

4.4 Learning and Implementation

All probability densities assumed Gaussian are learned from training data using Maximum Likelihood Estimation. The densities which can take an arbitrary shape are as well learned using a non-parametric method like Kernel Density Estimator. The training data is annotated manually using a software specifically developed for that purpose.

In order to infer the most likely configuration of random variable values, we construct two Markov Random Fields, each of them representing the posterior probability for one of the layers: tracklet generation and data association. The maximization of both posteriors of Eq. (4.2) is usually NP-Hard. To overcome this problem, we approximate the solution using the Tree Reweighed Belief Propagation, which is a message passing algorithm which infers the Maximum a Posteriori configuration of the set of variable realizations. We use a C++ implementation of the algorithm (libDAI), developed in [70].

4.5 Experiments and Results

We evaluate our Multiple Target Tracking algorithm in experiments on synthetic and real image sequences, and provide quantitative results of the experiments. Usually works on Multiple Target Tracking use standard metrics to evaluate the error on the prediction of the localization of the targets in each frame. A popular approach in recent works suggests the use of MOT Metrics to evaluate MTT precision and accuracy [12]. This measure takes into account four different aspects of the quality of the results:

- Precision of the hypothesis localization.
- False positive errors.
- Missed detections.
- Number of track label miss-matches.

An important difference between our experimental demonstration against other examples shown in the literature is that we do not include a detection phase in the tracking process. This means that we do not filter the objects that appear in the image, and thus we consider every observation as a potential target to track. This is justified due to the nature of the applications we are dealing with. In the synthetic scenario it is obvious that all the objects present in the images are valid targets, since we do not introduce any artificial clutter or noise. In the headlight tracking application, we threshold the intensity values of the images to discern the interesting blobs, and we track indistinctively every light, and every reflection, which are both present in our ground-truth as valid targets. In the last example, the bacteria growth sequence, we manually construct a perfect segmentation, which does not produce any undesired artifacts.

Therefore, we cannot evaluate the precision of our hypothesis, since the hypothesis location is always the same as the target location. A target cannot be miss-detected, since non-occluded targets have at least one observation, and every detection has at least one target associated, meaning that false-positives cannot occur. The only MOT component that we can use as a quality measure is the number of track label miss-matches. Moreover, we also measure the accuracy of tracklet generation using a typical feature-matching evaluation metric, by simply counting the ratio of correct matchings against the total. Table 4.3 shows the quantitative results for the three experiments.

4.5.1 Synthetic Sequences

We have generated two synthetic sequences of 100 frames, each containing a number of targets imaged as a circle with a fixed radius. The sequence A contains an average of 10 particles per frame, the sequence B, 15 particles per frame. The radius of the particles is 10 and 5 for each sequence respectively. Figure 4.7 shows a time-stamp of 20 contiguous frames of both sequences. In the first sequence the

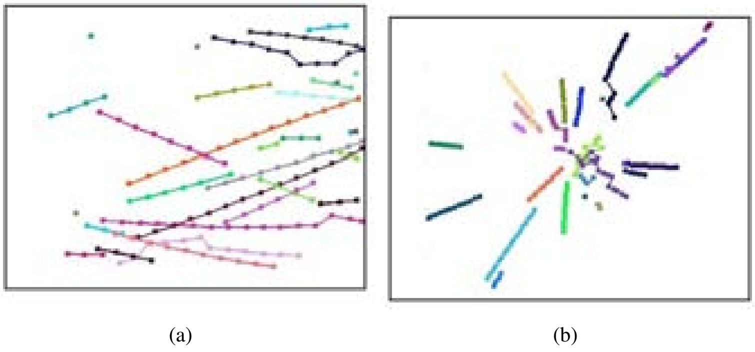


Figure 4.7: Time-stamp of 20 frames in both synthetic tracking sequences. In (a), targets move from left to right disappearing in the right image border. In (b),targets move from the image borders towards the image center, where they disappear.

motion is achieved with a XZ projection of a 3D helical motion of targets. In the second the particles move towards a sink in the center of the image . It can be seen how the particles follow more or less linear trajectories in both cases. Each color represents a track label. Sudden changes of colors, or sharp corners along tracks, indicate a miss-match of track labels.

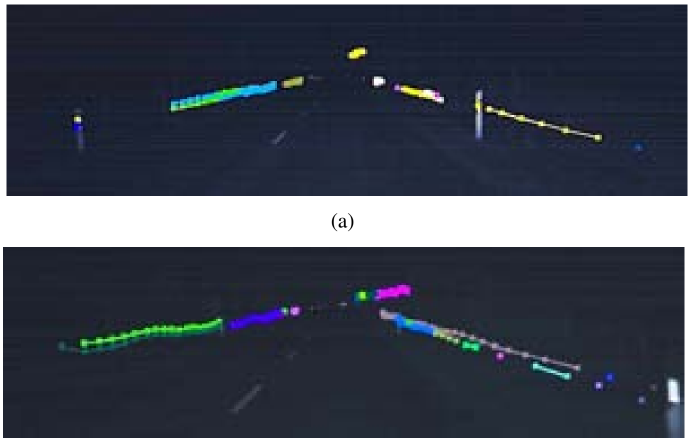
In the first synthetic video targets appear from the left image border, and move with constant speed and random direction towards the right image border. The results show few target miss-labeling and long and consistent tracks. Note that there exist several target-target interactions, originating in some cases a fast merging followed by a splitting. The second synthetic experiment shows targets moving towards the image center and disappear. The performance in this case is slightly hampered by the cluster of close targets that results in the image center, which can confuse both tracklet generation and data association processes. Besides, since the image centre behaves as a target *sink* the multiple occlusions in that area produces an undesirable effect: targets are assigned with the label of a nearby occluded target, and the data association algorithms believes that it has successfully recovered an occlusion.

4.5.2 Tracking of Car Headlights

In the context of an Intelligent Headlight Control Application, the main problem is to classify a blob in the image as a car or a reflection, in order to automate the activation of the light beams. Usually, a complex classifier gathers features from every blob in the image, and labels them as vehicle or non-vehicle. A tracker can also be included working in parallel with the classifier [89], in order to provide additional information (e.g combining the classifier beliefs of a given target between different frames). This is the reason why, in this specific application, we are interested of tracking every blob in the image, and there is no need to perform a detection process.

We perform multiple target tracking in two sequences of 100 frames each. Note that this scenario is specially difficult because the camera recording the images is constantly moving since it is mounted in a car. Far away lights are represented as very tiny blobs of few pixels that are very hard to track. Classic trackers like Kalman Filter would certainly perform poorly in this situation, since measurements of distant targets are separated by few image pixels, and the movement of the camera makes very hard to solve the data association problem. Moreover, there are hardly any appearance features to rely on.

Although both sequences perform well in terms of tracklet generation and identity linking, the first sequence shows a slightly better accuracy. This is due to several labeling mistakes regarding elements close to the camera that move relatively fast. This is shown in figure 4.5.2 (b), at the right side of the



(a)

(b)

Figure 4.8: Representation of the target tracks in both headlights sequences. Each color represents a different label.

Application	N. Obj	% Trackets	% Labels
Synthetic1	36	0.12	0.16
Synthetic2	52	0.19	0.27
eadlights1	29	0.31	0.24
Headlights2	35	0.27	0.34
Bacteria	43	0	0.11

Table 4.3: Results for every video sequence. First column shows the total number of objects that appear in the sequence. Second column the ratio of incorrect tracklets. Third column shows number of track label miss-matches against total number of objects

image. The other most common type of mistakes are related to far away tiny blobs which, as stated before, are the main challenge of the tracking application.

4.5.3 Bacteria Growth

This experiment consists of tracking a growing number of bacteria, which are continuously dividing in two. This is an example of a tracking scenario where a target can become two, and we are interested in tracking these targets at the same time that we construct what is known as the cell mitosis lineage. The sequence provided has 54 frames, reaching in the last frame a maximum of 43 targets simultaneously in the image. In Figure 4.5.3 we show qualitative results of the bacteria tracking experiment.

In this application we slightly modify the appearance likelihood of Eq. (4.3), to improve the results, by profiting from the little appearance information that the targets display. We use the overlap ratio between the pixel areas the bacteria cover in consecutive frames, to determine the most likely correspondence between bacteria. The results show a perfect tracklet creation, and almost no target miss-labeling regarding the data association process. Notice that in the case of this application we need to remove the identity coherence constraint of Eq. 4.15, since we want several targets to take the same label simultaneously (originated from the same parent bacteria).

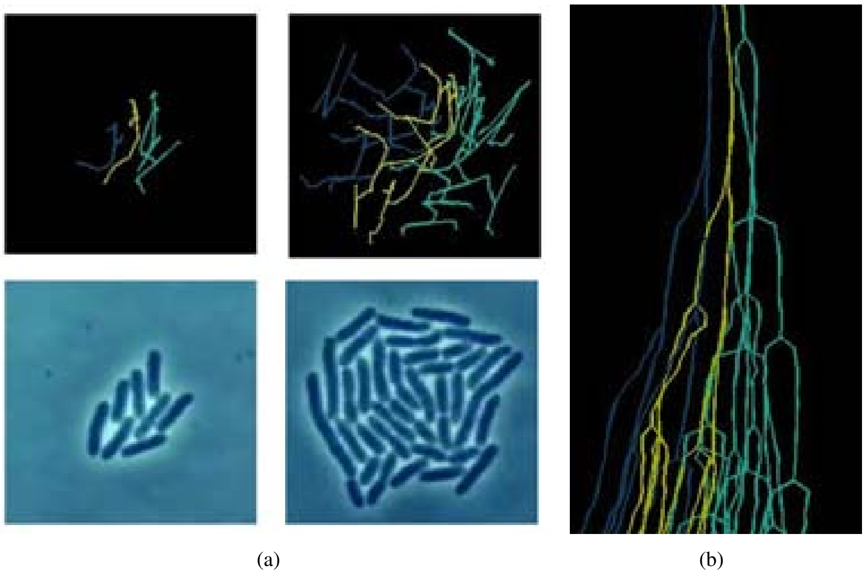


Figure 4.9: In (a), bottom, two frames of the bacteria growth sequence. In (a) top, the corresponding paths. The three different colors indicate the original bacteria parent that originated the track. In (b), it is represented the lineage tree, by plotting each track's horizontal component against time.

4.6 Conclusions

We have modeled the problem of Multiple Target Tracking with presence of occlusions, merges and splits, as a two stage probabilistic method. The probability densities that model the target behavior and data association are all learnt form training data. We have provided insights into the different scenarios one can find when dealing with the problem of Tracking, and we also present our model as a general solution to deal with different tracking scenarios simultaneously. We have proved the suitability of our approach in three different experiments, one synthetic and two with real images, in which we track particles presenting non or very poor appearance features. This makes it a challenging problem, mainly when addressing the data association of objects and observations.

Chapter 5

Region Matching

5.1 Introduction

So far in this dissertation we have performed matching without using appearance features. This was justified since in our motivational application (headlights) there were hardly no discriminant appearance features to be extracted from the headlight blobs. In the following chapters we are interested in relying on such high-level appearance features that are extracted from local image patches. Specifically, our goal now is to perform *dense* matching, understood as finding correspondences between regions provided by a segmentation algorithm. In this chapter we propose a new hierarchical image representation, which improves the performance and accuracy of region correspondences between pairs of images. We use inference on high-order graphical models as our preferred optimization framework in a consistent fashion with respect to the previous chapters.

Image region matching has been widely used for applications such as 3D surface registration [109], object retrieval [48, 96] and place recognition [98]. The main challenge when matching image regions lies in encoding the large variability in both appearance and arrangement of regions obtained from natural images under changing conditions (viewpoint, illumination). The existing segmentation algorithms are often sensitive to small changes, generating image regions with poor repeatability in terms of shape and size, and fusing and dividing regions inappropriately.

A common approach to tackling these variability problems is the use of graph matching. As we show in the previous chapters, most graph matching methods establish geometric constraints on the image structure by preserving a distance measure between nodes embedded in a Euclidean space [69, 97]. One major drawback of this approach is its restriction to a near-rigid or near-isometric assignment, which results in poor performance when there are large variations in the node arrangements of the two graphs to be matched.

One way to overcome this limitation is to rely on the statistical properties of the graph. Commute times between graph nodes have been recently used in computer vision applications to characterize the layout of a graph, proving to be stable against structural variations of the scene. In [8], commute-times are applied to describe the structure of the image content. In the context of graph matching, H. Qiu et al. [78] computes the minimum commute-time spanning tree to achieve a robust layered graph representation.

In the case of region matching, considering many-to-many correspondences is a critical issue, since the inconstant pattern of regions generated by a segmentation algorithm produces frequent region fusions and divisions. Some authors extend one-to-one matching by combining two *one-to-many* and

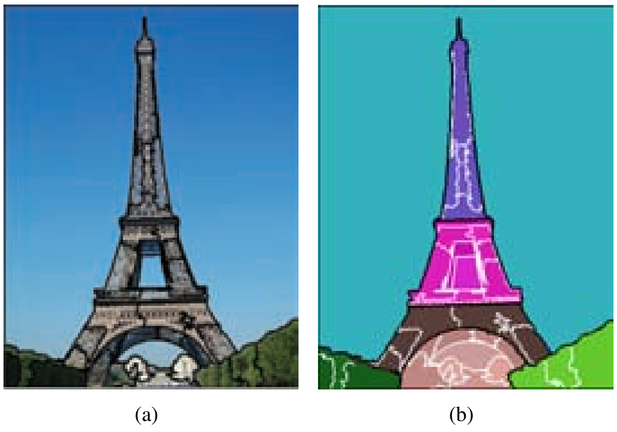


Figure 5.1: (a) Example of segmented image using *mean-shift* (region-layer). (b) Image segments grouped in seven clusters, represented by different colors (cluster-layer).

many-to-one mappings [108]. In [40], whose region matching problem is closely related to ours, *many-to-many* matching is carried out by simply labeling a node with that of its neighbor. Many-to-many region matching is achieved in [96] by using graph-editing operations.

Works like [29] have noted the convenience of combining *many-to-many* matching with an abstract model of the scene structure. In the context of object categorization, the abstraction of the scene, together with a *many-to-many* matching framework, is necessary to achieve greater robustness against inter-category variations [28]. Following this line, we propose a coarse-to-fine hierarchical representation that encodes the image global structure in a loose manner using commute times, while relying on local invariant features to handle photometric changes of the image. The bottom level of the hierarchy also handles *many-to-many* correspondences by comparing subsets of region boundary shapes. This hierarchical abstraction allows us to break down the matching problem into smaller sub-graphs, facilitating the computation of a large volume of configurations of region fusions and divisions.

The graph matching is modeled as a high-order discrete energy minimization. Because solving such high-order functions is a difficult problem with existing optimization techniques, we propose very sparse high order potentials which, as shown in [83], can be efficiently optimized with standard message passing inference algorithms.

This Chapter is organized as follows: In section 5.2 we introduce the hierarchical model and its two layers of abstraction. In section 5.3, the method is formulated as an energy optimization, composed of several potential functions. Section 5.3.4 details the inference algorithm, as well as the optimization of the high order terms. The experiments and results are presented in section 5.4, and finally our work is concluded in section 5.5.

5.2 Hierarchical Region Matching

We address the problem of matching two instances of the same object, which can undergo significant changes in illumination, viewpoint, and scale. As a consequence of these changes, regions in two different images that correspond to the same object, when segmented using an algorithm, display poor repeatability. In addition, occlusions are frequent and fusions and divisions of regions from one image to another may occur as well.

Our image representation contains information at two levels of abstraction. The lower level or *region-graph* contains image regions generated by a standard segmentation algorithm such as *mean-shift* [20]. The upper level, or *cluster-graph*, groups regions into potential semantic objects, with similar appearance and location. While the cluster level encodes the layout of the semantic components of the image, the region level improves the correspondences by examining local features within these components. Figure 5.1 shows an example of the region and cluster layers of an image.

5.2.1 Region Layer

From each image we obtain a set of regions using a standard segmentation algorithm. We define them as $r_i = (xy_i, T_i, H_i)$, where xy_i denotes the position of the region centroid, and T_i and H_i its texture and hue descriptors respectively. In order to express the region layout, we define a distance between regions as

$$D_r(r_i, r_j) = \alpha f(xy_i, xy_j) + (1-\alpha) g(T_i, T_j), \tag{5.1}$$

where f is the Euclidean distance between region centroids, g is the distance between their texture descriptors(e.g. local binary pattern), and $\alpha \in \{0, 1\}$ is a weighting factor which adjusts the importance of location or appearance in the graph structure. We express the local region structure by connecting nodes which are attached to each other. The graph adjacency matrix is defined as

$$\Omega_r(r_i, r_j) = \begin{cases} D_r(r_i, r_j) & \text{if } r_i \text{ shares a boundary with } r_j \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

5.2.2 Cluster Layer: graph of commute times

We group the image regions with the LP-based stabilities clustering method introduced in [50], using the distance of equation (5.1) as the affinity matrix. To achieve homogeneous groups of regions, we also penalize the color entropy of the clusters by assigning a penalty on every region r, as follows:

$$p_i = \frac{1}{K} \sum_{r_k \in \text{Knn}(r_i)} h(H_i, H_k) \tag{5.3}$$

where $h(r, k)$ is the euclidean distance between hue descriptors (e.g opponent color [101]) of two regions, and $Knn(r_i)$ is the set of K-nearest regions of r_i. Each of the clusters acts as a node on the cluster-graph, composing a collapsed version of the region-graph. We define a distance between clusters by averaging all region edge weights between two clusters as

$$D_c(c_i, c_j) = \frac{1}{|c_i||c_j|} \sum_{r_k \in c_i} \sum_{r_l \in c_j} \Omega_r(r_k, r_l). \tag{5.4}$$

The cluster-graph is a fully connected graph, and its adjacency Ω_c is given by the commute time matrix between clusters. As shown in [19], this can be computed from the spectrum of the normalized Laplacian of the adjacency graph:

$$CT(c_i, c_j) = vol \sum_{k=2}^{N} \frac{1}{\lambda_k} \left(\frac{\phi_k(c_i)}{\sqrt{d_i}} - \frac{\phi_k(c_j)}{\sqrt{d_j}} \right)^2 \tag{5.5}$$

where $vol = \sum_{k=1}^{N} d_k$, and d_k is the degree of node k. The terms ϕ_k and λ_k denote the k^{th} eigenvalue and eigenvector of the graph Laplacian.

5.3 Formulation

We approach the graph-matching problem as the optimization of a energy function. We propose a high order matching model that combines information of the image semantic structure and region appearance. Many-to-many correspondences are scored jointly by applying a similarity measure between invariant shape descriptors, calculated on the boundaries of sub-sets of regions.

5.3.1 Energy function

Let R_1 and R_2 be the sets of regions generated from two images, I_1 and I_2 respectively. The set of all possible correspondences between the regions is denoted by $R = R_1 \times R_2$. Analogously, let C_1 and C_2 represent the sets of clusters of regions, and the set of all matchings between clusters as $C = C_1 \times C_2$. A matching configuration of both graph layers between two images is represented as a binary vector of indicator variables $x \in \{0,1\}^{C \cup R}$, where each variable,

$$x_a = \begin{cases} 1 & \text{if } a = (i,j) \in C \cup R \text{ is active} \\ 0 & \text{otherwise} \end{cases} \tag{5.6}$$

Note that a matching configuration refers to the particular set of correspondences made between nodes in I_1 and I_2. Our energy function is defined as the weighted sum of 4 energy terms,

$$\begin{aligned} E(x) &= \lambda_{capp} E^{capp}(x) + \lambda_{rapp} E^{rapp}(x) + \\ &\quad \lambda_{ct} E^{ct}(x) + \lambda_{shape} E^{shape}(x), \end{aligned} \tag{5.7}$$

where the λ coefficients weight the influence of each of the terms. The *capp* and *sapp* terms refer to the appearance cost on the cluster and segment layer respectively. The *ct* term favors a coherent cluster structure on pairs of cluster associations, and the *shape* is a high order term with two different roles: First, it encourages subsets of segments to match other subsets with similar shape, and second, it communicates between both layers by constraining the clusters and segments that can be matched simultaneously. Next, we develop the potential function formulation for each of the terms.

5.3.2 Potential Functions

The term E^{capp} is defined as

$$E^{capp}(x) = \sum_{a=(c_i,c_j)\in C} \theta_a^{capp} x_a \tag{5.8}$$

and encourages correspondences between clusters with similar appearance. The cost θ_a^{capp}, with $a = (c_i, c_j)$ is the average distance between the descriptors of the regions contained in the clusters:

$$\theta_a^{capp} = \frac{1}{|c_i||c_j|} \sum_{r_k \in c_i} \sum_{r_l \in c_j} g(T_k, T_l) + h(C_k, C_l), \tag{5.9}$$

where c_i and c_j are clusters in the first and second image respectively. Note that functions g and h were previously mentioned in equations 5.1 and 5.3.

In the E^{ct} energy term, we use commute times to capture the global information of the cluster arrangements. Our model does not impose any restriction on the number of correspondences to which one cluster or region can be matched. We denote the set N of pairs of associations involved in a fusion or division as every pair $x_a = (c_i, c_j), x_b = (c_k, c_l)$ such that $c_i = c_k \vee c_j = c_l$. The set M contains the remaining pairs of cluster associations $x_c = (c_i, c_j), x_d = (c_k, c_l)$ such that $c_i \neq c_k \wedge c_j \neq c_l$. We define the potential as

$$E^{ct}(x) = \sum_{x_a, x_b \in N} \theta_{ab}^{fd} x_a x_b + \sum_{x_c, x_d \in M} \theta_{cd}^{ct} x_c x_d \tag{5.10}$$

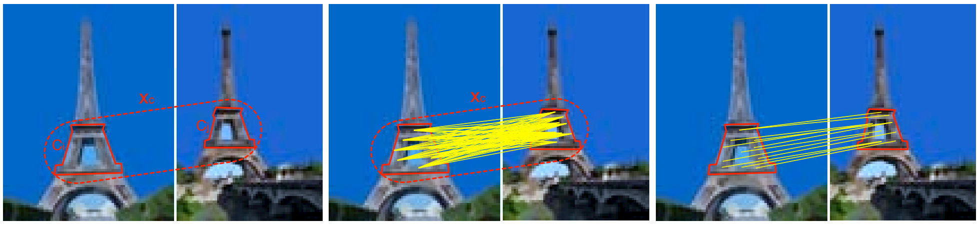


Figure 5.2: Left pair shows an example of cluster correspondence (in red), defined by the indicator variable $x_c = (c_i, c_j)$. The center pair shows all possible region correspondences (in yellow) between clusters c_i, c_j. The right pair shows the final many-to-many region assignment corresponding to the lowest cost clique configuration.

and the costs involved as

$$\theta^{fd}_{ab} = \begin{cases} e^{D_c(c_i,c_k)/2\sigma_1^2} & \text{if } c_i = c_k \\ e^{D_c(c_j,c_l)/2\sigma_2^2} & \text{if } c_j = c_l \end{cases} \tag{5.11}$$

$$\theta^{ct}_{cd} = \frac{|CT(c_i,c_j) - CT(c_k,c_l)|}{CT(c_i,c_j) + CT(c_k,c_l)}, \tag{5.12}$$

where the σ_1^2 and σ_2^2 act as normalization parameters. The first cost θ^{fd} encourages fusions or divisions among clusters that are very close to each other, in terms of centroid distances and appearance. The θ^{ct} penalizes pairs of associations with dissimilar commute time distances between the source and destination cluster graph edges. The energy term E^{rapp} is a unary measure of appearance compatibility between image regions. It is defined as the sum

$$E^{rapp}(x) = \sum_{a=(r_i,r_j)\in R} \theta^{rapp}_a x_a \tag{5.13}$$

The cost θ^{rapp} encodes the similarity between two image regions. We have adapted the *self-similarity* descriptor introduced in [90] to describe the compatibility between two regions. Finally, the high order potential E^{shape} binds both graph layers by constraining the cluster and region correspondences that should not be active simultaneously. It also models the compatibility of many-to-many region matching by defining a shape similarity between sub-sets of regions. We describe it in detail in the next section.

5.3.3 High Order Potential

For every cluster association $x_c = (c_i, c_j) \in C$, we generate a high order clique $\mathbf{x}_{ij}$, composed of all region correspondences within the two clusters c_i and c_j, and the cluster correspondence itself x_c (see Figure 5.2). Formally,

$$\mathbf{x}_{ij} = \{x_a \cup x_c | x_c = (c_i, c_j) \wedge x_a = (r_k, r_l) \forall r_k \in c_i, \forall r_l \in c_j\} \tag{5.14}$$

Let Z be the set of all high-order cliques. The high order energy term is defined as

$$E^{shape}(x) = \sum_{\mathbf{x}_{ij}\in Z} \Psi(\mathbf{x}_{ij}). \tag{5.15}$$

The optimization of the high order terms is a difficult problem to solve [56]. In order to make the optimization feasible it is very convenient to model potential functions which are sufficiently sparse [**?**], so that only a few configurations contribute with a low cost, while the remaining configurations have a high constant penalty. We can take advantage of our hierarchical model to make our terms very sparse, and at the same time unify the matching of both layers, by assigning large costs to incompatible layer configurations. Given a clique $\mathbf{x}_{ij} = \{x_c, x_k, x_l, ..., x_n\}$, where x_c is the cluster association indicator variable, and $x_k, x_l, ..., x_n$ represent the correspondences between the regions, we define the high order potential as

$$\Psi(\mathbf{x}_{ij}) = \begin{cases} \theta_\infty & \text{if } x_c = 0 \wedge \sum_k^n x_k > 0 \\ \theta_0 & \text{if } x_c = 0 \wedge \sum_k^n x_k = 0 \\ s(\mathbf{x}_c) & \text{otherwise} \end{cases} \tag{5.16}$$

The term θ_∞ is a high penalty that forbids *incoherent* inter-layer configurations. Note that if the cluster correspondence x_c is not active, any active region correspondence within that pair of clusters is penalized. The term θ_0 is a constant cost when no region is matched in the clique. The function $s(\mathbf{x}_c)$ defines a cost for each *compatible* configuration of region correspondences $\mathbf{x}_c = \mathbf{x}_{ij} - \{x_c\}$, within the two clusters associated by the variable $x_c = 1$.

The function s pulls down the energy lower bound around the variable space configurations which are allowed. Moreover, it acts as a multi-wise region compatibility cost, evaluating the similarity between the shapes of the regions in correspondence. Let G_i be the set of all possible subgraphs of Ω_r, obtained by combining the regions $r_k, r_l, ..., r_n \in c_i$, in p-tuples, from $p = 1$ to $p = n$. Each of these combinations is then characterized by a shape descriptor $SC(g_i)$, of the points in the boundary of the image area defined by the subgraph combination $g_i \in G_i$.

Given a variable set $\mathbf{x}_c = \{x_a, x_b, ..., x_n\}$ related to the cluster correspondence $c = (c_i, c_j)$, and their respective sets of combinations G_i, G_j, we define the *shape* cost as

$$\theta^{shape}(\mathbf{x}_c) = \delta(SC(g_i), SC(g_j)) + \rho, \tag{5.17}$$

with

$$\rho = 1 - \frac{1}{min(|c_i|, |c_j|)} \sum_{a \in \mathbf{x}_c} x_a \tag{5.18}$$

The cost θ^{shape} is defined for every p-tuple $g_i = \{r_1, ..., r_p\} \in G_i$, and every q-tuple $g_j = \{r'_1, ..., r'_q\} \in G_j$, with $a = (r_1, r'_1), b = (r_1, r'_2), ..., n = (r_p, r'_q)$. The function δ is a similarity measure between shapes. Notice how the different realizations of the variable set $\{x_a, x_b, ..., x_n\}$ establish which regions in each side of the clique contribute to the joint shape represented by the descriptor. See Figure 5.3. In our experiments, we have used *shape-context* [9] in order to enable partial shape similarity measures and to achieve rotation and scale invariance in the outlines of the tuples. The term ρ is an extra penalty for unmatched regions.

We can further increase the sparsity of our potential function by forbidding configurations which produce *unnatural* matches. For instance, the number of active correspondences between two clusters c_i, c_j should not exceed the maximum number of regions contained within them. Thus, the final expression of the s function is formulated as

$$s(\mathbf{x}_c) = \begin{cases} \theta_\infty & \text{if } \sum_{a \in \mathbf{x}_c} x_a \leq max(|c_i|, |c_j|) \\ \theta^{shape}(\mathbf{x}_c) & \text{otherwise} \end{cases} \tag{5.19}$$

Note that adding this constraint forbids *many-to-many* region correspondences. This is in principle undesirable, since the scope of the matching is restricted to *one-to-one*, *one-to-many*, and *many-to-one* mappings, losing expressive power in the solution. On the other hand we considerably reduce the computation time, and prevent *trivial* solutions like matching all regions of the source cluster to all regions of the destination cluster.

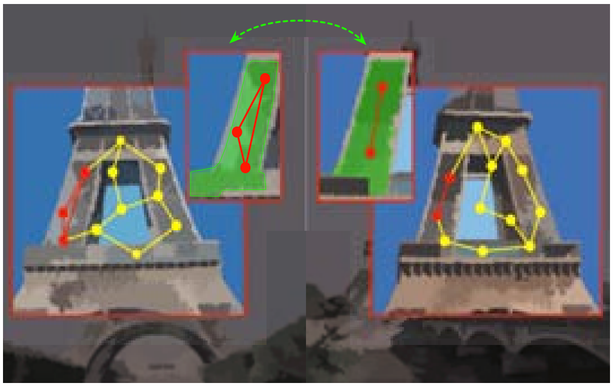


Figure 5.3: Example of sub-graph matching of joint region boundaries. The nodes in red denote a example of region tuples from each cluster. The green area represents the joint region pixels that define the boundary that is used to measure the shape similarity between the tuples.

5.3.4 Optimization Algorithm

We can gather all terms introduced previously, and formulate the optimization problem compactly as follows:

$$\min_{x \in C \cup R} E(\mathbf{x}|\theta) = \sum_{a \in C \cup R} \theta_a x_a + \sum_{(a,b) \in N \cup M} \theta_{ab} x_a x_b + \sum_{\mathbf{x}_{ij} \in Z} \Psi(\mathbf{x}_{ij}) \tag{5.20}$$

The goal is to find a configuration of labels of the vector **x** with the maximum a posteriori probability (MAP), or with the minimum energy E. This is, in general, an NP-hard problem. We approximate the solution using a message-passing inference algorithm called Tree-Reweighed Belief Propagation [105]. Unfortunately, the computational complexity of algorithms based on message-passing increases exponentially with the clique size [56]. As pointed out in section 5.3.3, we convert the high-order terms to pairwise terms, taking advantage of the sparseness of the potential function Ψ.

Analogously to the technique introduced in Section 3 we apply the technique of [83] to transform sparse high order potentials to equivalent quadratic ones, by adding an additional *switching* variables for each of the *favored* clique configurations. The only limitation of this approach is the high dimensionality of these variables, which can reach 2^L states, with L being the size of the largest clique. Following this idea we are able to reduce our energy objective function to pairwise, at the expense of slightly increasing the number of variables. In case the above mentioned limitation due to the high dimensionality of the *switching* variables becomes a problem, it can be avoided by adjusting the clustering parameters to keep the size of the clusters sufficiently small.

5.4 Experiments and Results

We evaluate our method on pairs of images of world-famous monuments downloaded from Flickr. The monument classes selected are: *Eiffel Tower, Liberty Statue, Notre Dame cathedral* and *Taj Mahal*. Figure 5.4 shows examples for each of the classes. The image dataset is composed by 20 images per class. The images were selected manually in order to obtain object instances with different scales, illumination conditions, and camera viewpoints.

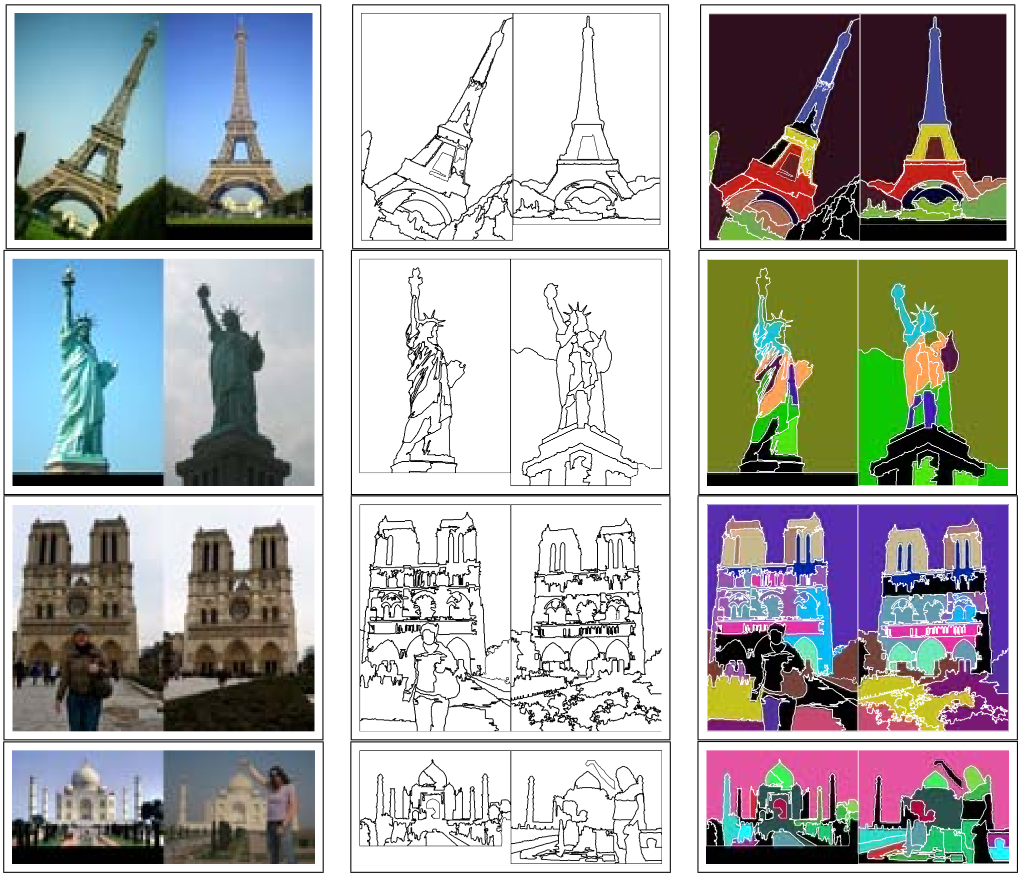


Figure 5.4: Sample results for each of the classes of the dataset, in each row. The left column shows pairs of original images, the center column the segmented images, and the right column the matching result. Regions in the two images which share the same color are in correspondence. Areas in black are not matched. Best viewed in color.

5.4.1 Region Matching on Image Pairs

The goal of this experiment is to show the accuracy of the algorithm in terms of the number of regions that are correctly matched. We evaluate our method on pairs of images of world-famous monuments downloaded from Flickr. The monument classes selected are: *Eiffel Tower, Liberty Statue, Notre Dame cathedral* and *Taj Mahal*. Figure 5.4 shows examples for each of the classes. The image dataset is composed by 20 images per class. We consider a pair of regions to be a correct correspondence if the regions share a minimum of 50% of the common pixel area of two objects. We compute the *region mismatch error* as the ratio of failed correspondences to the total number of correct region pairs. When an image region is not included in both images, we consider it as an occlusion, and matching such a region is considered to be an error. We also count as an error *many-to-many* matchings between groups of regions that could be matched *one-to-one*.

To perform these experiments we set the energy component weights to $\lambda_{capp} = 3, \lambda_{rapp} =$

Table 5.1: Region matching error ratio

mismatch error ratio	Eiffel	Liberty	T.Mahal	N.Damme
spectral pairwise [58]	0.34	0.45	0.51	0.35
ours pairwise	0.46	0.58	0.67	0.53
ours H.O	0.25	0.17	0.30	0.21

$5, \lambda_{ct} = 4.5$ and $\lambda_{shape} = 2.5$. The normalization parameters of the commute time costs σ_1 and σ_2 are both set to 0.25. Finally the cost of leaving all correspondences inactive within a clique, θ_0 is set to 1.25. In this experiment we remove the constraint of Eq. 5.19 to allow *many-to-many* correspondences.

Table 5.1 shows a comparison of the average error ratio per class. We compare our algorithm with and without high order terms, to the pairwise spectral matching presented in [58], and show that the higher order potential clearly reduces the ration of mismatched regions. Removing the high order terms in our framework results in a less expressive configuration of region correspondences, which tends to produce matchings between large *patches* of regions, usually belonging to the same cluster. Moreover, the main handicap of [58] is the one-to-one correspondence constraint, which is very restrictive in the context of region fusions and divisions.

The homogeneous and repetitive texture and color in some monuments (Eiffel tower, liberty statue) misleads the appearance costs, resulting in an aliasing effect where regions in one image are matched to similar incorrect regions in the other image. Adding high order terms solves this problem and, surprisingly, the average computation time is reduced.

Figure 5.4 shows some examples where the *many-to-many* matching plays an important role. Note the Taj Mahal dome which is split in two regions in the left image and correctly matched to the whole in the right image. Analogously the top blue part of the Eiffel tower also matches correctly after joining two regions. More complex many-to-many region mappings can also be seen in the red regions of the Eiffel tower, and the arm and body of the statue of liberty.

We have observed that some mismatched regions are due to a symmetric image structure. This is true for instance, for the bottom left and bottom right vegetation areas surrounding the Eiffel tower. The effect is mainly caused by the commute time graph layer, which imposes equal penalties to clusters that are symmetrically arranged. Moreover, if the cluster appearance is not sufficiently discriminative, as is the case in the greenish areas on the Eiffel tower images, it is likely that one of the symmetric regions in the first image will be matched to the second symmetric region in the second image and vice versa.

5.4.2 Convergence of the Inference Algorithm

Following we describe an interesting effect on the convergence of the inference algorithm. One would expect the addition of high order cliques in the energy function to significantly increase the inference running time, after adding extra variables of high dimensionality. The time complexity of message-passing based algorithms is $O(M^L)$, where M is the number of possible variable labels, and L the size of the biggest clique. As can be seen in Figure 5.5, the convergence rate of the inference algorithm increases considerably with the high order terms and thus the average computation time is reduced. We have run a set of experiments on image pairs of each class, and analyzed the inference computation times. The running times differences between classes are mainly related to the number of regions produced by the segmentation. The Taj Mahal and Notre Dame classes usually compose a scene more complex than the Eiffel tower and the liberty statue, and are segmented in more image regions.

This is due to the influence of the high order potentials on the shape of our original energy function. Recall that the high order terms add a constant cost to all the energy variable space, with the exception of certain variable configurations that are not penalized. The energy tends to fall into the local minima

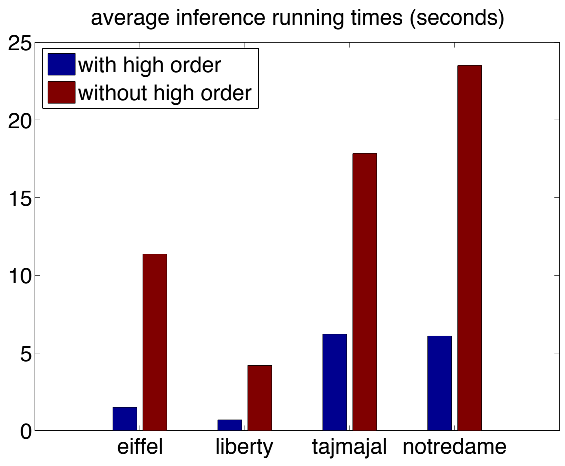


Figure 5.5: Average optimization running times of each class. In blue the times obtained from running the algorithm with high order terms. In red, the time for the same experiments without high order energy terms.

defined by the *allowed* variable configurations with low cost, and the optimization quickly converges within this restricted variable space.

5.5 Conclusions and Future Work

In this work we have presented a novel approach for image region matching. Our method first encodes the image information with a coarse-to-fine hierarchy graph, whose levels embed different abstractions of the image contents. The upper level contains information regarding the appearance of the scene semantic objects, as well as information on their spatial arrangements using statistical properties of the graphs. The bottom layer handles the matching at region-level by encoding information on appearance and region shapes. Our method explicitly handles *many-to-many* correspondences by scoring the similarity of subsets of region boundaries, which are encoded in high order energy terms.

We demonstrate the effectiveness of our method performing matching on a few images of monuments. The results prove the matching to be robust against large variations on illumination, and viewpoint. The commute times metric applied to encode the scene layout overcomes the structural noise problem usually encountered in graph-based representations.

We believe the layered graph-based scene representation proposed in this work is a natural and effective framework to perform region matching in arbitrary images. Of course this requires further validation of our approach in larger datasets.

Chapter 6

Co-segmentation

6.1 Introduction

After developing the region matching model presented in the previous chapter, we supposed that finding region correspondences between images could have potential benefits when applied to image co-segmentation. In this chapter we present an algorithm that makes use of region correspondences between images in order to perform non-supervised co-segmentation. We propose an iterative algorithm that is initialized with an estimate of the image foreground. To avoid the iteration to expand to a trivial result (all pixels foreground) the region matching imposes inter-image labeling constraints that effectively bound the foreground/background area. We present results that show that our method overcomes the non-supervised algorithms, and it is comparable to other state-of-the-art supervised methods.

Bottom-up segmentation of generic images is a long standing goal in computer vision for its many potential applications. It is a highly unconstrained and ill-posed problem which has given rise to a multitude of approaches. Co-segmentation is a recent approach to cope with this lack of constraints. Given two or more images showing the same object, or similar instances of the same object class, the goal is to partition them into foreground (object) and background regions under the assumption that the background changes significantly while the foreground does not. Co-segmentation methods leverage this fact in order to determine what is the foreground region. Note that this is a chicken and egg problem: the aim is to compare something —the two foregrounds— that is still unknown. As difficult as it may appear, it is a especially appealing approach because, in its purest version, it only requires providing an additional image containing the same or a similar instance of the target object. Several applications of co-segmentation have been explored in the literature. One consists of increasing the performance of image retrieval by removing the background from the image similarity metric, thus focusing on the object which is sought [84]. Also, automatically segmenting the instances of an object in a collection of pictures would allow to create visual summaries [7]. A few more specific applications are the segmentation of neuron cells from electron microscopy images sequences [104], the reconstruction of 3D models of individuals (with user interaction) [51] and recognition of people wearing the same clothes [35].

In this Chapter we describe a new approach to the co-segmentation problem which has several unique characteristics (see section 6.1.2). Based on matching super-pixels resulting from an over-segmentation algorithm, it not only produces foreground/background partitions but also relates their regions. We believe that this will allow to extend the applications of co-segmentation to those needing the correspondence between foreground pixels, contours or regions, like parts-based recognition or 3D

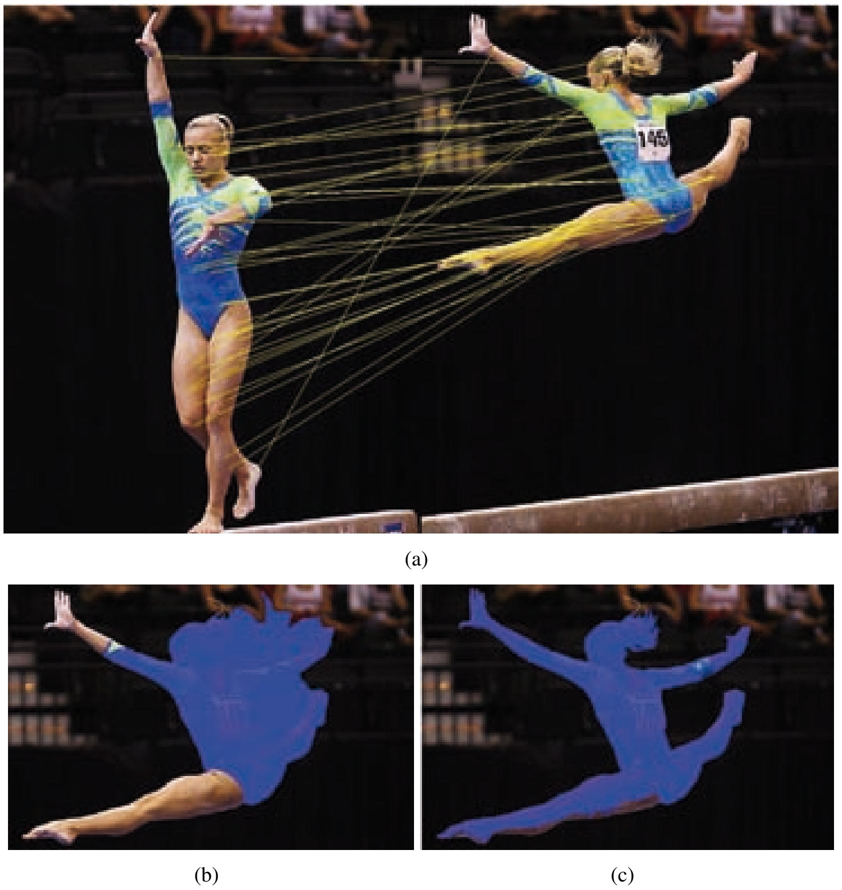


Figure 6.1: In (a) yellow lines show region matching results for the foreground area of two images. In (b) the blue-colored pixels show the results of the foreground objectness-based initialization. In (c), our results.

reconstruction. Another distinctive feature is that we model both the hypothesized foreground and background appearance distributions separately. Thus, the assumption of similar foreground and changing background is replaced by image-specific distributions of foreground and background. Provided that they can be well estimated, the co-segmentation may succeed even though the background does not change much.

6.1.1 Previous work

In the seminal work by Rother *et al.* [84], the problem was posed as labeling pixels as foreground or background through energy minimization. The energy included a key term to measure the L1-norm dissimilarity between the unnormalized foreground histograms and another pairwise term regularizing the solution in both images.

Subsequent methods [71, 84, 102] compared foreground color histograms in different ways, succeeding on relatively simple image pairs. However, color histograms are clearly dependent on lighting conditions and also on the foreground scale since they are not normalized. Non surprisingly, the most recent works employ additional features, such as SIFT and texture descriptors [72], saliency [16], and Gabor filters [41].

Maybe the distinctive trait of each method is how does it build the partition hypothesis and perform the foreground comparison, that is, how to cast the co-segmentation problem into some solvable formalism. The dominant approach is minimization of an energy function equivalent to MAP estimation on a MRF [7, 16, 84, 102] but other original approaches have been tried like the minimization of a quadratic pseudoboolean function [72] or max-flow min-cut optimization [41]. More interesting, Vicente *et al.* [103] generate a large number of candidate segmentations for the two images by means of a variation of Min-Cuts. Then a Random forest regressor, trained with many pairs of ground-truth segmentations, scores each pair of segmentations, one for each image. An exact A^* search finds the pair of segmentation proposals with the highest score. This is one of the few automatic methods reporting results on iCoseg, a challenging benchmarking dataset.

Closely related to co-segmentation is the problem dubbed as co-clustering. The goal is similar though the approach is not. Given two or more images and their overs-egmentations, the aim is to group the regions in each image into two or more clusters, each corresponding to an object of interest. One difference with respect to co-segmentation is that co-clustering concerns regions, not pixels. Glasner *et al.* [36] perform this clustering by comparing the color and shape of groups of regions. They are thus able to co-segment two or more images with similar backgrounds provided that the foreground shape is roughly the same, like in nearby frames of a video sequence. They pose co-clustering as a quadratic semi-assignment problem which is solved by linear programming relaxation, like in [104]. Joulin *et al.* address co-segmentation of two or more images by means of unsupervised discriminative clustering. Their elegant formulation ends up in a relaxed convex optimization.

All these works provide original formulations and successful results for automatic co-segmentation. But only a few of them [46,103] go beyond the relatively simple image pairs of the first papers ('banana', 'bear', 'dog', . . . on varying backgrounds), and focus on the challenging datasets iCoseg and MSRC. iCoseg contains a varying number of images of the same object instance under very different viewpoints and illumination, articulated or deformable objects like people, complex backgrounds and occlusions. MSRC contains images of different objects belonging to the same class, with varying aspect.

6.1.2 Goal

The novelty of our work is a new automatic co-segmentation method which exhibits the following characteristics:

- Fully unsupervised, meaning that there is no need of training with ground-truth segmentations of images from the same or from other classes (like in [102]).
- Able to work with more than two images, a novelty just explored in recent papers [36, 46, 72]. This means that not only the formulation is more general but that the method has to scale well with the number of images.
- The method not only produces a foreground / background partition of the images but also computes many-to-one and one-to-many associations (correspondences) among regions from different images. These regions may constitute object parts and thus co-segmentation would allow further applications.
- It is comparable with state of the art non-supervised *and* supervised methods on the benchmark datasets iCoseg and MSRC. On this regard, we take as reference the very recent works by Vicente *et al.* [103] and Joulin *et al.* [46] for the reasons mentioned previously.
- Performing well in the difficult case of similar backgrounds, overcoming the constraint associated with the first co-segmentation methods, as explained above.

The reminder of this Chapter is organized as follows: In section 6.2, we formulate the co-segmentation problem as an unsupervised binary-class labeling of a set of images depicting the same object instance. We start proposing a multi-image representation, and define the problem in terms of an energy minimization. In section 6.3, we extend the scene representation to include several segmentation proposals to capture the image elements at different scales, and we present a generative appearance model of the scene, trained at testing time. In section 6.4, we show that spectral matching can be used to establish correspondences between image regions, by exploiting the underlying information of region arrangements. Section 6.5 presents the experimental set-up and results on standard databases, while the last section concludes the Chapter.

6.2 Co-segmentation Formulation

Let us consider a set of images $I = \{I_1, I_2, ..., I_n\}$ containing an instance of the object of interest. Let us also consider that the images have been partitioned based on visual appearance, by a segmentation algorithm such as mean-shift [20]. We propose a two-layered MRF composed of region nodes and pixel nodes. The indexing of nodes is denoted by $\mathcal{V} \in \mathcal{V}_r \cup \mathcal{V}_p$, where the sets $\mathcal{V}_r$ and $\mathcal{V}_p$ correspond to regions and pixel nodes respectively. Slightly abusing notation, we write $\mathcal{V}_r(k)$ to denote the regions of image k, and $\mathcal{V}_p(k)$ to refer to the pixels. The MRF comprises a vector of boolean random variables $\mathbf{X} = (X_i)_{i \in \mathcal{V}_r} \cup (X_j)_{j \in \mathcal{V}_p}$. The variables can take two values: $X = 0$ indicates background and $X = 1$ indicates foreground. Our goal is to infer a consistent labeling of regions and pixels that better separates the foreground and background areas of the image set. We state the problem as the minimization of the following energy function:

$$E(\mathbf{X}) = \lambda_1 E^{pixel} + \lambda_2 E^{region} + E^{scale} + E^{matching} \tag{6.1}$$

The first two components E^{pixel} and E^{region} are unary potentials encoding the likelihood of pixels and regions belonging to foreground or background, weighted by parameters λ_1 and λ_2. The E^{scale} term enforces a consistent labeling of pixels and the region they belong to. The last term $E^{matching}$ encourages coherent inter-image labeling of regions. The next sections detail the formulation of these terms. Figure 7.3. shows an example of MRF.

6.3 Generative Foreground/Background Model

We first compute several segmentation proposals at different scales in order to have a rich description of the scene. Then, an iterative algorithm infers the appearance likelihood distribution of foreground and background. The peculiarity of our framework is that the distributions are trained at testing time using a rough estimate of the image foreground/background labeling, instead of ground-truth annotations.

6.3.1 Multi-scale Segmentation Pool

An over-segmented dictionary $\mathcal{R}$ of regions is generated using mean-shift with different sets of parameters, over every image in the set I. The original region set $\mathcal{V}_r$ is re-defined to include every region of the dictionary, for all images: $\mathcal{V}_r = \mathcal{R}_k, \forall I_k \in I$. Our model comprehends pixels as well as regions. Each pixel has as many parent regions as levels of segmentations computed to build the dictionary (See Figure 7.3). To encourage a coherent labeling between regions and pixels we introduce the first energy component E^{scale}, as

$$E^{scale}(\mathbf{X}) = \sum_{(i,j) \in \Delta} \eta(1 - \delta(X_i, X_j)), \tag{6.2}$$

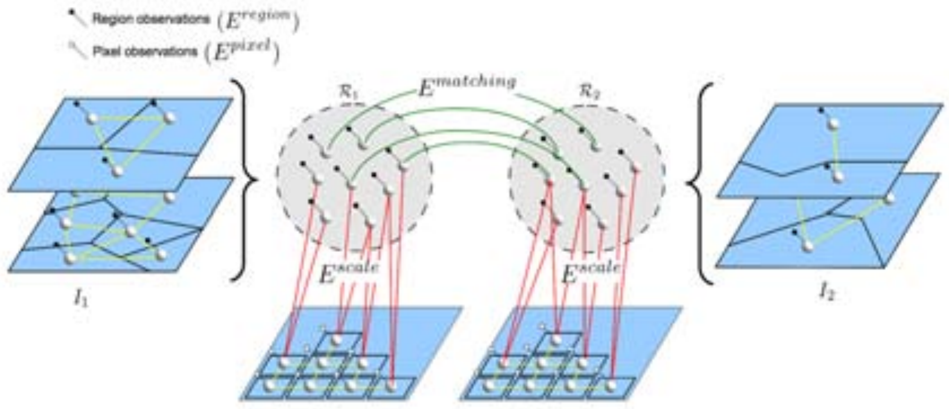


Figure 6.2: Markov Random Field of the multi-scale multi-image model (two images), illustrating the energy components. In the left and right sides, it is shown the pool of proposal segmentations (two scales) for images I_1 and I_2. The two big gray circles represent the dictionaries of regions $\mathcal{R}_1$ and $\mathcal{R}_2$. The small white and black circles denote the singleton potentials for regions and pixel nodes. The red vertical edges (Δ) connect pixel nodes with regions from the image dictionaries. The green horizontal lines ($\mathcal{E}$) show examples of region correspondences. The yellow intra-image edges between pixels and regions denote optional smoothing potentials.

where the cost η penalizes pairs of pixel and region nodes with different labels. The function δ is the Dirac delta function, and the set Δ contains the indexes of every pair of overlapping regions and pixels.

While pixel labeling helps to overcome errors propagated from the mean-shift segmentations (providing finer labeling atoms), the region level enforces spatial grouping. Moreover, multiple segmentations capture the image semantics at different scales, making the inter-image region matching robust against scale variations.

6.3.2 Pixel and Region Potentials

Defining the likelihood of a pixel/region belonging to the foreground or the background in a unsupervised framework is a challenging task. A priori, there is no information available about such distributions. However, analogously to [103], we assume without loss of generality that the image foreground is an object. Therefore, we can use the algorithm of [5] as a measure of objectness. Note that the objectness measure is applied out-of-the-box, without re-training with the databases used in the experiments. This is important because we want our method to be free of ground-truth segmentations, or ground-truth class labeling. The method of [5] outputs a prior of the object location as the probability of covering an object with a sampled window. We sample 10^4 bounding boxes, and calculate the probability of a pixel belonging to an object by simply averaging the score of every bounding box containing that pixel. Then we extend these scores to the regions, by averaging the probabilities of every pixel contained in each of the regions.

One of the typical requirements for performing co-segmentation is that the appearance of the foreground differs from that of the background to a certain extent. We propose a generative image model, on which the objectness measure plays a guiding role to iteratively infer both distributions. The inference is based on the premise that the objectness-based initial distribution resembles the foreground and is distinct to the background. Note that even if this may not be always true for every image, it is very likely that it remains true if we jointly model the distribution from several images. Figure 6.1.(b) shows an example of an objectness initialization.

The distributions are constructed with simple features. We use the RGB color of the image pixels and a texture descriptor extracted from every region of the set $\mathcal{R}$. In our implementation we use Gaussian Mixture Models (GMM) to estimate pixel color distributions. The foreground and background GMMs are denoted as $\mathcal{H}^f$ and $\mathcal{H}^b$ respectively. We also train a texture-based appearance model of the image regions using a texture descriptor (Local Binary Pattern). However, in this case, a parameter estimation for GMMs with high dimensionality (50 bins) requires a large amount of training data and computation time. Instead, a linear SVM classifier $\mathcal{F}$ is trained over the texture descriptors of the foreground and background estimation.

The algorithm starts by initializing the labels of pixels and regions using the output of the objectness measure, and estimating the color and texture distribution of the background and foreground from this first rough labeling. We feed our unary potentials by querying the learnt distributions, and optimize the energy function of Eq. 6.1 to obtain a new labeling. Then, we iteratively update the distributions from the last output labeling until reaching a maximum number of iterations, or a convergence criteria. The procedure is detailed in Algorithm. 2. Constructing independent distributions for texture and color makes the model robust against difficult cases where the foreground and background have a similar appearance. When one of the features (color, texture) is not discriminative enough, we rely on the other to forbid one distribution to *leak* into the other.

A weakness of such an iterative approach is the initial seeding. A poor initialization results in ill-formed distributions with spurious samples that may bias the foreground model towards the background, and vice versa. In practice, what we observe is that one of the distributions slowly expands with every iteration, and quickly covers every pixel of the image. In order to make the model robust to poor initializations, we build as many appearance models as images in I in such a way that the distribution corresponding to image I_k is constructed with the information of every image except I_k. Following this approach, the wrong training samples will not contribute with a high probability when the same samples are used at testing time to query the distribution. We extend the previous formulation to denote the two GMMs of a specific image k as $\mathcal{H}_k^f$ and $\mathcal{H}_k^b$. This also applies to the texture classifier of image k, now denoted as $\mathcal{F}_k$. Given the set of n images, the singleton potential for the regions is formulated below, as the logarithm of the probability estimate returned by the texture classifier.

$$E^{region}(\mathbf{X}) = \sum_{k}^{n} \sum_{i \in \mathcal{V}_r(k)} -log(\hat{P}_k^f(T_i)X_i + \hat{P}_k^b(T_i)\overline{X}_i), \tag{6.3}$$

The probability $\hat{P}_k^f(T_i)$ is the estimate for the *foreground label* predicted by the classifier $\mathcal{F}_k$ on the texture descriptor T_i of region i. The term $\hat{P}_k^b(T_i)$ is the estimate for the *background label* on the same texture descriptor.

The singleton potential of a pixel node is the resulting cost of applying the logarithm to the color likelihood distribution $\mathcal{H}$. Formally,

$$E^{pixel}(\mathbf{X}) = \sum_{k}^{n} \sum_{j \in \mathcal{V}_p(k)} -log(P(C_j|\mathcal{H}_k^f)X_j + P(C_j|\mathcal{H}_k^b)\overline{X}_j), \tag{6.4}$$

where C_j is the color (e.g. RGB value) of pixel j. The term $\mathcal{H}_k^f$ refers to a gaussian mixture model trained on the foreground pixels of every image except I_k, and $\mathcal{H}_k^b$ is the analogous model for the background.

Algorithm 1 Iterative Foreground/Background Modeling

```
1: Initialize $\mathbf{X}_i, \mathbf{X}_j \leftarrow$ objectness, $\forall i \in \mathcal{V}_r, \forall j \in \mathcal{V}_p$.
2: repeat
3:     Estimate $\mathcal{H}_k^f \leftarrow GMM(\mathbf{X}_j = 1), \forall I_k \in I$
4:     Estimate $\mathcal{H}_k^b \leftarrow GMM(\mathbf{X}_j = 0), \forall I_k \in I$
5:     Train SVM $\mathcal{F}_k \leftarrow \mathbf{X}_i, \forall I_k \in I$
6:     $\mathbf{X}^* \leftarrow argmin_{\mathbf{X}} E(\mathbf{X})$
7:     Update labels $\mathbf{X}_i, \mathbf{X}_j \leftarrow \mathbf{X}^*$
8: until convergence
```

6.4 Region Matching

A key aspect when tackling the co-segmentation problem is the exploitation of the inter-image information. For challenging cases in which objects are deformable and change considerably in terms of viewpoint and pose, it is difficult to leverage the spatial distribution of the regions in order to find correspondences. Usually, matching methods establish geometric constraints on the image structure by preserving a distance measure between nodes embedded in a Euclidean space. One major drawback of this approach is the restriction to a near-rigid or near-isometric assignment, which results in poor performance when there are large variations in the node arrangements. We overcome this limitation by relying in the statistical properties of the graph of regions, by applying commute times as a distance between pairs of matching regions.

Let (r_i, r_j) be the indexes of two arbitrary regions from the dictionary $\mathcal{V}_r$. The distance between regions is defined as

$$D(r_i, r_j) = \alpha d(C_i, C_j) + (1 - \alpha) d(S_i, S_j), \tag{6.5}$$

where C denotes the RGB color of the image region as the mean color of the pixels contained in it. In the second term, S refers to the SIFT descriptor extracted from the image regions, obtained by computing a dense SIFT on every pixel of a region using a 16 by 16 patch, and clustering them in 8 bins. The function d is a χ^2- distance measure, and α is a weight expressing the influence of feature similarity.

The structure of regions within an image is represented as a graph of regions, with its adjacency matrix defined as:

$$\Omega(r_i, r_j) = \begin{cases} D(r_i, r_j) & \text{if } r_i \text{ shares a boundary with } r_j \\ 0 & \text{otherwise} \end{cases} \tag{6.6}$$

Commute times have been recently used in [8] to characterize the layout of a graph, proving to be stable against structural variations of the scene. The commute time matrix between regions can be efficiently computed from the spectrum of the normalized Laplacian of the adjacency graph:

$$CT(r_i, r_j) = vol \sum_{k=2}^{N} \frac{1}{\lambda_k} \left(\frac{\pi_k(r_i)}{\sqrt{d_i}} - \frac{\pi_k(r_j)}{\sqrt{d_j}} \right)^2 \tag{6.7}$$

where $vol = \sum_{k=1}^{N} d_k$, and d_k is the degree of node k. The terms π_k and λ_k denote the k^{th} eigenvalue and eigenvector of the graph Laplacian.

We denote every possible correspondence a between one region in image I_1 and another region in image I_2 as $a = (r_i, r_j) \in I_1 \times I_2$. The matching score of those correspondences is defined by the matrix M, where,

- $M(a, a)$ denotes the affinity of individual assignments given by the distance between regions defined in Eq. 6.5. Given a correspondence $a = (r_i, r_j)$,

$$M(a, a) = D(r_i, r_j) \tag{6.8}$$

iCoseg	Ours	[46]	[103]	uniform	obj.
Alaskan bear	**86.4**	74.8	90.0	79.0	79.5
Red Sox Players	**90.5**	73.0	90.9	86.8	77.2
Stonehenge1	**87.3**	56.6	63.3	88.8	85.5
Stonehenge2	**88.4**	86.0	88.8	68.4	70.0
Liverpool FC	**82.6**	76.4	87.5	82.9	85.0
Ferrari	84.3	**85.0**	89.9	73.9	78.0
Taj Mahal	**88.7**	73.7	91.1	83.4	74.9
Elephants	**75.0**	70.1	43.1	83.5	80.6
Pandas	60.0	**84.0**	92.7	68.7	81.3
Kite	**89.8**	87.0	90.3	76.0	77.3
Kite panda	**78.3**	73.2	90.2	62.0	78.4
Gymnastics	87.1	**90.9**	91.7	62.7	75.8
Skating	76.8	**82.1**	77.5	73.7	72.9
Hot Balloons	**89.0**	85.2	90.1	78.2	84.1
Liberty Statue	**91.6**	90.6	93.8	64.4	79.4
Brown Bear	**80.4**	74.0	95.3	82.2	78.1
mean accuracy	83.9	78.9	85.3	75.9	78.6

Table 6.1: Bold numbers represent classes in which our method's performance overcomes the non-supervised competitor [46]. In bold red, the scores for classes in which our method outperforms the state-of-the-art supervised method. In the second column the results as reported in [103].

- $M(a, b)$ defines how well a pair of region correspondences match. In our case, we use this term to preserve a commute time distance between pairs of regions in correspondence. Given a pair of correspondences $a = (r_i, r_j), b = (r_k, r_l)$,

$$M(a, b) = \frac{|CT(r_i, r_j) - CT(r_k, r_l)|}{CT(r_i, r_j) + CT(r_k, r_l)} \tag{6.9}$$

As stated in [?], the matching problem reduces to find the set of region correspondences $(r_i, r_j) \in \mathcal{E}$ that maximizes the score M. If we represent the set of possible correspondences as a vector of indicator variables, such that $y(a) = 1$ if $a \in \mathcal{E}$, and zero otherwise, the matching problem can be formulated as the following optimization:

$$y^* = argmax(y^T My) \tag{6.10}$$

We use the algorithm of [?] to optimize the above objective, and define a new set of corresponding regions $\mathcal{E} = \{a|y^*(a) = 1\}$, matching every possible pair of images of the input set. Finally, we introduce the last energy term, $E^{matching}$, which imposes a penalty on corresponding regions with different labels. We can write it analogously to Eq. 6.2, as

$$E^{matching}(\mathbf{X}) = \sum_{(i,j)\in\mathcal{E}} \theta(1 - \delta(X_i, X_j)), \tag{6.11}$$

by noting that the penalty is equal to 0 when both corresponding regions belong to the foreground or both to the background, and θ otherwise.

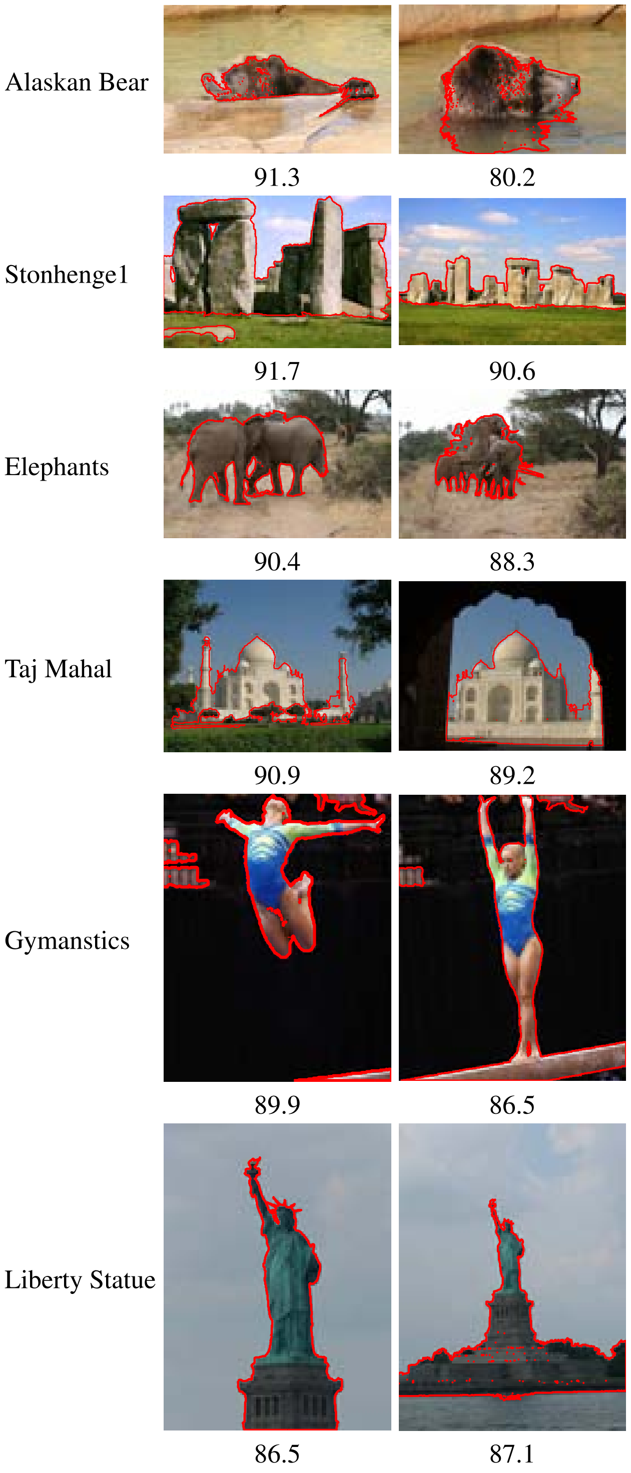


Figure 6.3: Example results on the iCoseg dataset. A red line separates foreground and background areas.

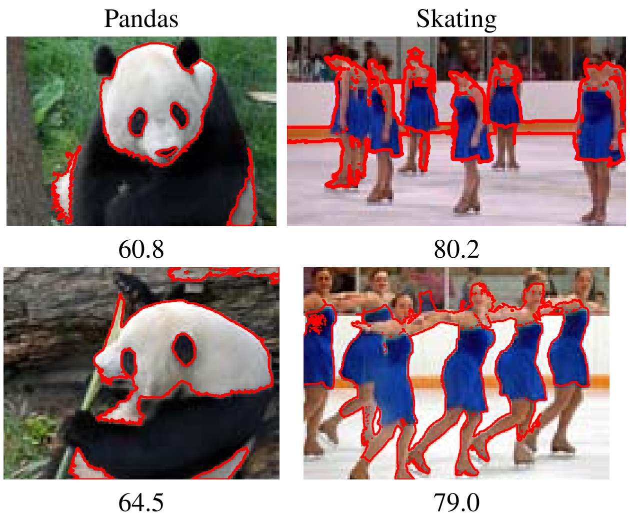


Figure 6.4: Examples of cases where the method fails in classes *Panda* and *Skating*. In the case of the panda, the objectness initialization leads the foreground to a local minima on the white fur. The images of the *Skating* class present complex compositions of objects.

6.5 Qualitative and Quantitative results

We evaluate our method with three different experiments. We report results on the iCoseg and MSRC datasets, and we illustrate the application of co-segmentation by matching, with an example of part-based correspondence.

We present qualitative and quantitative results of our algorithm. The segmentation accuracy of a given image is measured by computing the ratio of correctly labeled pixels of foreground and background with respect to the total number of pixels, like in [103].

6.5.1 Experimental set-up

The sub-modularity of the pairwise potentials is assured, since we only apply a cost on the variable configurations $[X_i \neq X_j]$. This lets us optimize the objective using graph-cuts but, in principle, any other optimization algorithm could be used as well. We choose graph-cuts because it provides a good trade-off between optimization speed and low energy bound. The number of components in the GMMs is automatically determined using the center-based clustering algorithm proposed in [50]. The last detail remaining is the value of the penalty scalars (θ, η) and unary weights (λ_1, λ_2). Since we want to avoid learning these parameters from training data, we adopt a conservative approach to set them: we set both penalties to the minimum integer cost 1, and we uniformly assign the same weight $\lambda = \lambda_1, \lambda_2$ to both singleton potentials. At the end, our method only depends on three parameters: λ, α, and the maximum number of iterations. It is worth to mention that only the λ value has an important influence on the algorithm performance. The stopping condition of the iterative process depends on the ratio of pixels that switched label since the last iteration. If this percentage is less than 2.5%, the algorithm stops. We use three different levels of scale segmentations, with the same mean-shift parameters for every image. We set the parameter α which weights the contribution of color and SIFT in the distance measure to 0.5. The parameter λ to scale the appearance potential is set to 2.75.

It is very common in the literature to apply a smoothing on the label values using an extra pairwise

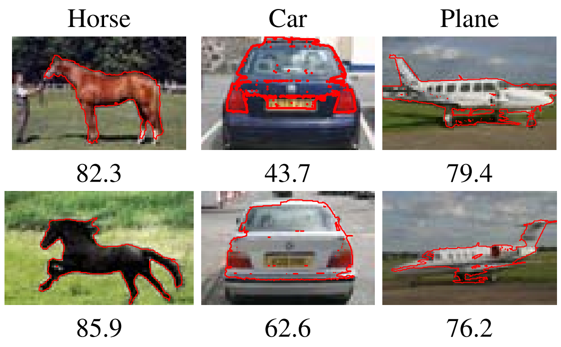


Figure 6.5: Example results on the MSRC dataset. The images show a high color variability between foregrounds of the same class, in *Horse* and *Cars (back)*. The background of the *plane* class does not significantly change.

term between neighbor variables. We leave it as an optional energy component, because it proved to be not much influential in the performance of the algorithm, and we avoid setting an extra penalty parameter.

6.5.2 iCoseg database

The iCoseg database was introduced in [7]. It contains 643 images divided into 38 classes with hand-labelled pixel-level segmentation ground-truth. Each class is composed of approximately 17 images. For this experiment, and for the sake of comparison, we use the same sub-set of 16 classes reported in [103]. We simultaneously co-segment groups of (at most) 10 images, and average the results for each of the groups. The images of each group are randomly selected, to avoid unfair grouping of affine-looking images.

In Table 6.1, we compare the performance of our method with a recent non-supervised method [46]. That is, a method that does not require ground-truth segmentations of object instances. Our results are on line with state-of-the-art algorithms such [103] (third column), which trains a pairwise energy from ground-truth segmentations of pairs of objects, tested against new groups of images.

The fourth column shows results with a uniform segmentation: best error rate of full (all ones) and empty (all zeros) segmentations. The last column contains the objectness-based initialization results. The bold red figures show classes in which our method outperforms [103]. This is mainly due to the high similarity on the image background (Elephant and Stonehenge), for which our method performs better because it estimates both foreground and background distributions. On average, our result for all 16 classes is slightly below [103] (just -1.4%).

The *Pandas* class performs the worst because an incorrect objectness initialization in the majority of its images keeps the foreground distribution *trapped* inside the white fur patches of the panda (See Figure 6.4). If the object presents a high color variability within the same instance, a correct initialization of the first foreground estimate is key to achieve satisfactory results. The *Skating* class is difficult to segment due to the complexity of the object. Again, the appearance variability between the color of the skaters' costume and the body parts, makes the foreground distribution fall in a local minima covering only the costume. The skaters' legs and heads are labeled as background.

MSRC	images	Ours	Joulin et al. [46]	uniform
Cars (front)	6	65.9	**87.65**	64.0
Cars (back)	6	52.4	**85.1**	71.3
Face	30	76.3	**84.3**	60.4
Cow	30	80.1	**81.6**	66.3
Horse	30	74.9	**80.1**	68.6
Cat	30	**77.1**	74.4	59.2
Plane	30	**77.0**	73.8	75.9
Bike	30	62.4	**63.3**	59.0

Table 6.2: Segmentation accuracy for the MSRC dataset and Weizzman horses.

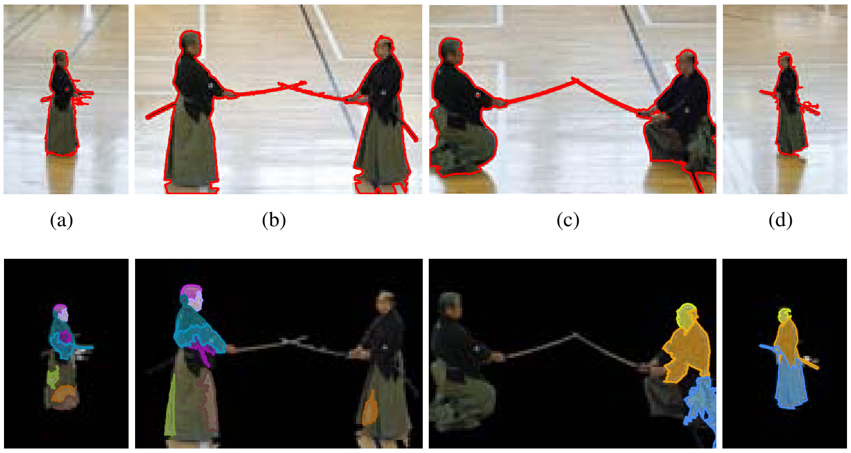


Figure 6.6: In (a,b,c,d), the output of the co-segmentation. The pair (e,f) shows an example of matching, and (g,h) another. Each color represents a correspondence between a pair of foreground regions.

6.5.3 Objects with variate appearances

The MSRC database depicts images of different instances of the same class. This contradicts one of the main hypotheses of our method, which strongly relies on a unique aspect of the foreground object. For instance, some classes show objects with variate colors and textures (e.g. Cars). We show that our method performs reasonably well even though this assumption does not hold.

Table 6.2. shows comparative results on the MSRC dataset and Weizman Horses dataset. In comparison to [46], our method outperforms the reported results when the background hardly changes, such as the *plane* class. As pointed out in [46], plane images have a similar background (the airport) that makes the task of separating foreground and background harder. Our algorithm does not suffer from this limitation, as long as foreground and background do not look alike.

Cars perform poorly due to the small number of images available in the database (6), and the large intra-class variability, especially regarding color. Figure 6.5. shows an example of this. Our method tends to identify windows and light-beams as the common foreground region.

6.5.4 Part-based recognition

Obtaining region correspondences within a co-segmentation framework is very useful for understanding the semantics of a scene. In our third experiment we use the region matching output to identify parts of the common objects co-segmented. We simply gather the regions selected as foreground by our co-segmentation algorithm, and select the correspondences with the highest spectral matching scores. Figure 6.6. shows the corresponding regions of four images of the *Kendo* class from the iCoseg database. The images are paired in two sets to show the part-recognition of each of the fighters. The heads and regions close to the head present the higher matching scores, being the only regions with discriminative features. In Figure 6.6. (e,f), only the upper part of the body finds matching candidates scoring over the threshold. In (h), the whole body of the fighter is correctly matched.

6.6 Conclusions

We have proposed a multi-scale multi-image representation that is able to model complex scenes containing several objects. A non-supervised iterative algorithm is presented, that is able to separate foreground and background by modeling both appearance distributions on pixels and regions. We show that is possible to take advantage of an explicit matching of regions, that increases the consistency of the foreground and background models across images. Our approach has shown to be robust against deformable objects as well as changes on object poses and camera viewpoint. We also overcome the limitation of other recent methods, which require background dissimilarity among the input images.

Our algorithm shows competitive results with state-of-the-art methods, and outperforms recent non-supervised co-segmentation approaches. It has shown good performance on two types of databases, one (iCoseg) contains the same object instance per class, while the other (MSRC) contains objects with varied appearances within the same class. One of the main advantages of our method is that it does not require different backgrounds in the input images. Other advantage is that the performance improves with the number of images available. We also present a prove of concept to illustrate the use of co-segmentation as a starting point to perform part-based recognition.

Chapter 7

Video Cosegmentation

7.1 Introduction

In the previous chapter we introduce a non-supervised image co-segmentation algorithm. In this chapter we extend our formulation to *video* co-segmentation, which consists on separating foreground from background areas in a set of videos, which depict the same (or similar) object performing the same (or similar) action. To the best of our knowledge this problem has never been approached before in the computer vision literature. It is especially interesting the fact that additional videos can be provided from heterogeneous sources (such as Youtube) in order to improve single video segmentation. We provide qualitative results on benchmarking videos showing that our approach is comparable with state-of-the art video segmentation methods.

Video segmentation has been defined as the problem of partitioning a video sequence into coherent regions with regard to motion and appearance properties [60]. We adopt here a more specific definition: we are interested only in those regions belonging to the objects of interest, which are those appearing in the foreground, over a possibly changing background. The outcome, thus, is a set of regions spanning space and time, sometimes dubbed `tubes'. Foreground segmentation is useful for several computer vision tasks including video analysis, object tracking, object recognition, 3D reconstruction, video retrieval, and activity recognition.

In spite of its potential applications, relatively few works address the problem of video segmentation, perhaps because the addition of one dimension increases the already high difficulty of unconstrained 2D segmentation. The reviewed works show that the most favored approach is to extend single image segmentation techniques to multiple frames, exploiting the fact that there is redundancy along the time axis and that the motion field is smooth. Thus, for instance, Levinshtein et al. [60] extend super-pixel grouping [59] (also known as turbopixels) to 3D. Sundaram and Keutzer [92] apply spectral clustering to all the video sequence pixels with an affinity matrix given by the gPb 2D contour detection algorithm [67] which combines intensity, color and texture. Several works pose the problem as one of labeling using minimum energy optimization of a Markov Random Field where nodes are now voxels [22, 57, 62, 99] or 2D regions [42], again a successful segmentation strategy in single images. Grundmann et al. [38] build their hierarchical algorithm for long sequences upon Felzenszwalb and Huttenlocher's [31] graph algorithm for 2D image segmentation. Likewise, Huang et al. adapt the graph-cut algorithm to run on 3D hypergraphs whose nodes are regions resulting from an over-segmentation of each frame.

Our approach is different in that we do not pursue video segmentation through the extension of

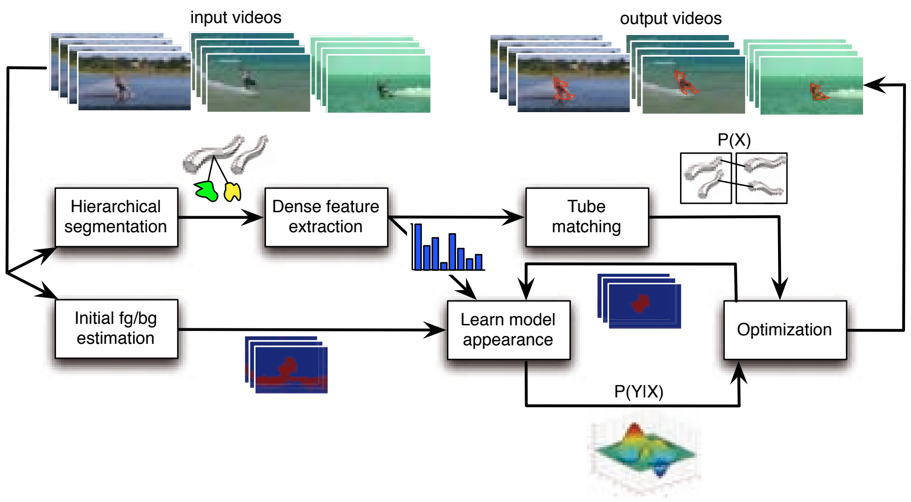


Figure 7.1: Diagram describing the steps of the algorithm.

some image segmentation *algorithm* but instead the extension of the *concept* of co-segmentation to include videos. Segmentation being a highly under-constrained problem, many practical methods resort to providing prior knowledge or constraints on how the objects of interest look (in terms of shape, size, color, location or structure). Image co-segmentation trades the need for such knowledge for something much easier to obtain, namely, additional images showing the same object, or objects of the same class, from different viewpoints. Now the segmentation problem is posed as one of differentiating the similar object regions in all the images from the more varying background. In this paper, for the first time, we extend this approach to video segmentation: given two or more video sequences showing the same object (or an object belonging to the same class) moving in a similar manner, we aim at outlining its spatio-temporal regions in all the videos.

In this Chapter we build on the co-segmentation method presented in Chapter 6, as we want to preserve several of its desirable properties:

Foremost, our model does not need training segmentation data in order to learn the foreground and background distributions. On the contrary, it learns such distributions at testing (segmentation) time.

As opposed to other co-segmentation methods, ours models the background in addition to the foreground distribution, thus being able to cope with videos with similar backgrounds.

Following the rationale of co-segmentation, our model is able to work with more than a pair of videos. In fact, with more videos both the commonality of the foreground and the diversity of the background increase.

We want to make clear that our method is not a simple extension to one more dimension. In fact, this is not possible because our proposed method is not a segmentation algorithm, but a reformulation that requires a different graphical model on which inference takes place in order to label regions as foreground or background. It also introduces the concept of a hierarchy of tubes and image regions. Moreover, our method does not work by co-segmentation of frames but of whole video sequences, as we will explain.

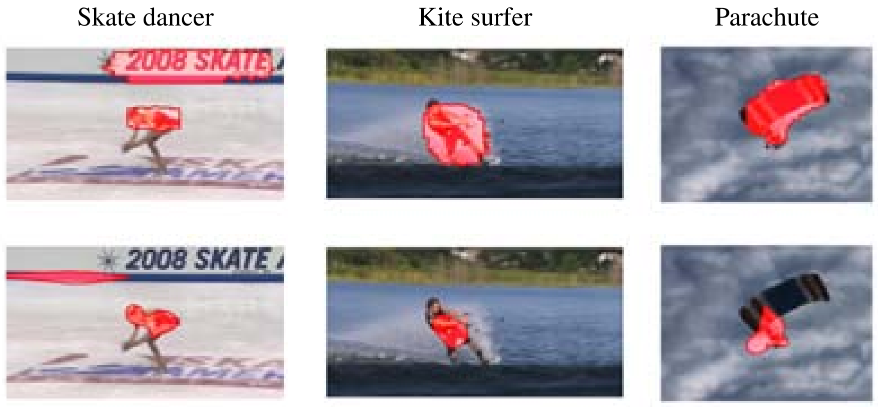


Figure 7.2: The first row shows examples of initializations with the objectness measure of [5]. The second row shows examples of the saliency-based measure. In some cases the objectness measure achieves a more accurate result (Parachute), while in others the saliency obtains a better labeling (Dancer).

7.2 Method Overview

Our goal is to perform figure/ground separation on multiple videos, posed as an optimization problem. The main idea is that the common foreground elements between different videos share certain appearance and motion characteristics. By establishing correspondences between those elements across different videos we should be able to enforce certain restrictions on the label (foreground or background) that such elements are likely to take.

Given a set of input videos, we start by grouping the pixels at two levels. At the higher level, video pixels are grouped in space-time, defining a set of video tube volumes. At the lower level, pixels are grouped into regions within each frame, as is typically done in a regular image segmentation algorithm. Each of these elements (tubes and regions) are described using densely extracted features.

An initial estimation of the foreground and background labeling is needed in order to construct a probabilistic distribution of the feature vectors of tubes and regions. We obtain such a labeling using a measure of objectness together with a saliency algorithm. As this initialization is a rough approximation, the models describing foreground and background are likely to contain incorrectly labeled samples.

We present a probabilistic framework where the likelihood of each element belonging to the foreground or background is calculated from the above mentioned representations. In addition, the model constraints are introduced through a prior term that provides region-tube consistency and enforces labeling coherence between corresponding objects across different input videos. The foreground and background representations are iteratively improved from new labeling that results from the optimization of a posterior distribution. Figure 7.1 shows a diagram detailing the steps of the process.

7.3 Iterative Foreground/Background Modeling

The task of co-segmentation requires relying on features that discriminate foreground from background regions. It seems a good assumption that foreground objects will have an appearance that is distinct from that of the background in addition to dissimilar motion behavior. For this reason, our model is composed of two types of elements: tubes and regions. The former encodes motion while the latter encodes appearance.

In order to generate these elements we apply the algorithm of [38], which produces hierarchical video segmentations whose regions are coherent along time. Two levels of segmentation granularity are generated for each frame. With the higher (less granulated) level we generate the collection of tubes by overlapping the regions having the same label across frames. With the lower (more granulated) level we generate the collection of regions. Since the segmentation is hierarchical, we can trivially establish a non-ambiguous correspondence between region elements and their corresponding tubes at the higher level. A graphical representation of the model is shown in Figure 7.3. Our goal is to label both regions and tubes as foreground or background.

7.3.1 Initial foreground estimate

The unsupervised nature of our approach requires an initial estimation of the appearance of the foreground and background. We have experimented with two different approaches. The first is an objectness measure based on visual cues that we apply to each frame independently [5]. The second is a video saliency measure, which also takes motion into account [43]. In Figure 7.2 we show examples of initializations. The objectness measure tends to fail in cases where the foreground contains motion blur and also when other elements of the scene appear to be objects due to their sharp edges and closed boundaries. The saliency measure works well in general, but performs poorly when the foreground is a fairly complex object, or composed by several parts. To obtain our initialization we combine both estimations by averaging the pixel score returned by objectness and saliency, and thresholding the resulting map.

7.3.2 Formulation

Let $\mathcal{V} = \{V_1, V_2, \ldots, V_N\}$ be a set of on N input videos. For all those input videos we extract two sets of labeling elements, tubes and regions, which we respectively denote as $\mathcal{T} = \{t_1, t_2, \ldots, t_M\}$ and $\mathcal{R} = \{r_1, r_2, \ldots, r_L\}$. A region refers to a set of pixels extracted from each video frame, while a tube is defined as a set of overlapping pixels across different frames of the same video. Note that while a region can only belong to one video frame, a tube belongs to various frames.

We propose a Markov Random Field that comprises tube nodes and region nodes of all videos in two separate layers, as illustrated in Figure 7.3. We represent these nodes as a vector of boolean random variables divided into two disjoint sets, denoted as $\mathbf{X} = (X_i)_{i\in\mathcal{T}} \cup (X_j)_{j\in\mathcal{R}}$. A variable $X = 0$ indicates that the tube or region belongs to the background, and $X = 1$ otherwise. An observation $Y \in \mathbf{Y}$ is a histogram of visual cues extracted from a tube or region element. Our goal is to obtain the MAP labeling that maximizes the following posterior, given the set of observations $\mathbf{Y}$:

$$P(\mathbf{X}|\mathbf{Y}) = P^{\mathcal{T}}(Y^{\mathcal{T}}|X^{\mathcal{T}})P^{\mathcal{R}}(Y^{\mathcal{R}}|X^{\mathcal{R}})P(\mathbf{X}). \tag{7.1}$$

The factor $P^{\mathcal{T}}(Y^{\mathcal{T}}|X^{\mathcal{T}})$ defines the likelihood of each tube variable belonging to the foreground or background. Analogously, $P^{\mathcal{R}}(Y^{\mathcal{R}}|X^{\mathcal{R}})$ defines the likelihood of each region belonging to the foreground or background. The factor $P(\mathbf{X})$ is a prior that imposes intra-video and inter-video labeling constraints. The set of random variables $X^{\mathcal{T}}$ refers to the set of tube variables $(X_i)_{i\in\mathcal{T}}$, while $X^{\mathcal{R}}$ refers to the set of region variables $(X_j)_{j\in\mathcal{R}}$. We describe in detail each of the factors in the following sections.

7.3.3 Figure-ground likelihood model

Once an initial labeling estimation is available we can model the appearance of foreground and background. Since the entities to be labeled are tubes and regions, it is desirable that each type of element

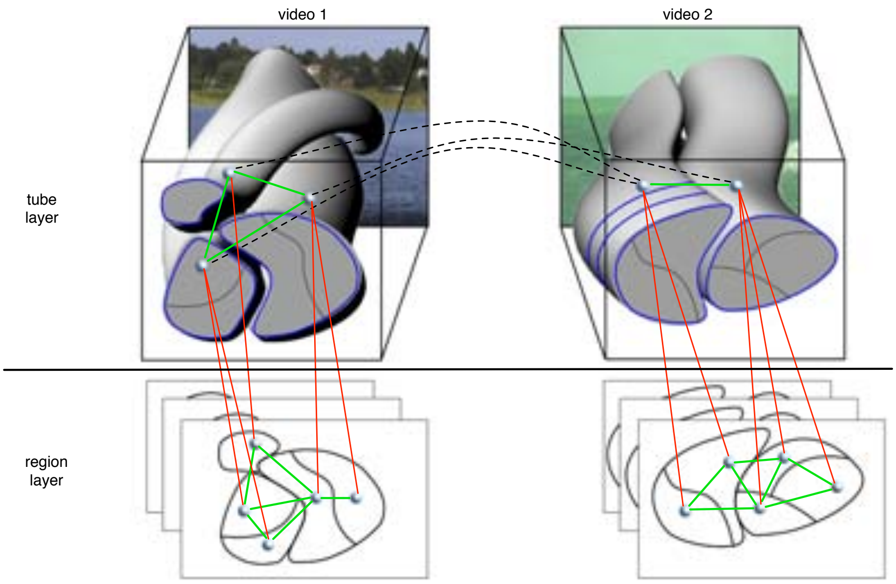


Figure 7.3: Two layer model of regions and tubes. The upper layer shows the tube volumes and the lower layer shows the regions. The graph nodes represent the tube and region random variables, and the connections represent the pairwise relationships encoded in the prior. The green edges illustrate the smoothing constraints and the red lines connecting the two layers represent the tube-region coherence, P^{coh}. The dotted lines that connect tubes from different videos refer to the matching constraint P^{match}.

encodes complementary information. As the tubes are spatio-temporal entities, their related likelihood $P^{\mathcal{T}}(Y^{\mathcal{T}}|X^{\mathcal{T}})$ encodes 3D features, or motion features. In the case of the regions, being 2D image areas, their distribution $P^{\mathcal{R}}(Y^{\mathcal{R}}|X^{\mathcal{R}})$ encodes image appearance features such as color or texture.

Describing the motion characteristics of a tube is achieved by uniformly sampling 3D HoG descriptors [49] along the tube volume. The descriptors span space and time, and are clustered in a unique descriptor vector using a histogram of accumulated descriptors [45]. Similarly, we sample 8×8 patches from the region elements and calculate texture (LBP) descriptors that are also aggregated in a single vector.

The likelihood distributions of the feature vectors of regions and tubes are calculated as:

$$P^{\mathcal{T}}(Y^{\mathcal{T}}|X^{\mathcal{T}}) = \prod_{k \in V} \prod_{t \in \mathcal{T}(k)} \hat{P}^f_k(H_t)X_t + \hat{P}^b_k(H_t)\overline{X}_t. \tag{7.2}$$

The term $\hat{P}_k(H_t)$ is the probability of a tube descriptor H_t, given by a logistic regressor k trained with 3D HoG descriptors. Note that there are as many classifiers as input videos. Testing samples from a given video are highly likely to be labeled in a similar manner to training samples from the same video, given their appearance and motion similarities. Consequently, a poor initialization of the training labels would result in poor labeling in the test set. For this reason, each regressor k is trained with data from videos $\{v_1, \ldots, v_{k-1}, v_{k+1}, \ldots, v_n\}$ and tested with the remaining video v_k. This removes the inherent likelihood bias towards training features from the same video.

Similarly, the likelihood of the region variables can be formulated as:

$$P^{\mathcal{R}}(Y^{\mathcal{R}}|X^{\mathcal{R}}) = \prod_{l \in V} \prod_{r \in \mathcal{R}(l)} \hat{P}_l^f(LBP_r)X_r + \hat{P}_l^b(LBP_r)\overline{X}_r, \tag{7.3}$$

where the term $\hat{P}_l(LBP_r)$ refers to the probability of the regressor l trained with LBP texture descriptors, given a region descriptor LBP_r. The super-indices f and b refer to the labeling of foreground and background respectively.

7.3.4 Iterative likelihood estimation and labeling

Recall that we start with a rough initialization of motion and appearance distributions, and iteratively update them. At the current iteration the motion and texture distributions are modeled using the labeling from the previous iteration, and then the likelihood densities are fed with the probabilities of the trained regressors. Finally, the posterior expression of Eq. 7.1 is optimized and a new labeling is generated. The process repeats and the likelihood models are progressively refined until satisfying some convergence criteria. Ideally, at each step the labeling solution obtained gets closer to the optimal foreground/background separation. Although fg/bg models are based on Support Vector Machines (SVM) in our framework, one could use statistical models or even modern manifold learning methods. The procedure is graphically detailed in Figure 7.1 and in Algorithm 2.

Algorithm 2 Iterative Foreground/Background Modeling

```
1: Initialize $\mathbf{X}_t, \mathbf{X}_r \leftarrow, \forall t \in \mathcal{T}, \forall r \in \mathcal{R}$.
2: **repeat**
3:     Train $SVM_k \leftarrow \mathbf{X}_t, \forall V_k \in \mathcal{V}$
4:     Train $SVM_l \leftarrow \mathbf{X}_r, \forall V_l \in \mathcal{V}$
5:     Estimate $\hat{P}(H_t) \leftarrow SVM_k(H_t) \forall t \in \mathcal{T}$
6:     Estimate $\hat{P}(LBP_r) \leftarrow SVM_l(H_r) \forall r \in \mathcal{R}$
7:     $\mathbf{X}^* \leftarrow \arg\max_{\mathbf{X}} P(\mathbf{X}|\mathbf{Y}) \propto P(\mathbf{Y}|\mathbf{X})P(\mathbf{X})$
8:     Update labels $\mathbf{X}_t, \mathbf{X}_r \leftarrow \mathbf{X}^*$
9: **until** convergence
```

7.4 Modeling the Prior

An iterative procedure such as the one described in the previous section cannot be used in practice if we assume a uniform prior distribution. An inaccurate initial estimation would produce poor appearance models of foreground and background that within such an iterative process are likely to expand until reaching a trivial solution (e.g every label is background).

We propose an informative prior that imposes certain restrictions that bound the foreground/background partition and limit its expansion. Moreover, it is desirable to constrain the labeling of the tubes and the regions included in them, to obtain a coherent labeling of regions and tubes. For the first constraint, we establish correspondences between tubes from different videos, similarly to the image co-segmentation. Our goal is to combine the information from different videos and constrain the label of corresponding elements to take the same value. Therefore, we seek to exploit inter-video information in such manner that a defective initialization in one of the videos will be balanced by an expected correct labeling in other input videos. Figure 7.3 shows a graphical interpretation of the model constraints represented by the connections between tubes from different videos (black, dotted) as well as by the links connecting tubes and regions (red).

The prior term factorizes into two terms corresponding to the two types of constraints we want to enforce:

$$P(\mathbf{X}) = \prod_{(t_1,t_2)\in\mathcal{E}} P^{match}(X_{t_1}, X_{t_2}) \prod_{(r,t)\in\mathcal{F}} P^{coh}(X_r, X_t) \tag{7.4}$$

The set $\mathcal{E}$ contains all the pairs of tubes that are in correspondence, while the set $\mathcal{F}$ defines pairs of regions and tubes, such that for a given pair (r, t) the region r is contained in the tube t. The first factor P^{match} is responsible for ensuring that corresponding tubes from different videos take the same label, and we formulate it as:

$$P^{match}(X_{t_1}, X_{t_2}) = \begin{cases} \alpha & \text{if } X_{t_1} \neq X_{t_2} \\ \beta & \text{otherwise} \end{cases} \tag{7.5}$$

The second factor P^{coh} enforces a coherent labeling between the hierarchical levels of the model, such that a region takes the same label as the tube to which it belongs. Formally,

$$P^{coh}(X_r, X_t) = \begin{cases} \gamma & \text{if } X_r \neq X_t \\ \delta & \text{otherwise} \end{cases} \tag{7.6}$$

The parameters $\alpha, \beta, \gamma, \delta$ are constant probability values. These are set in order to assign a high probability (~ 1) to a coherent labeling and a low probability otherwise (~ 0).

The definition of the set $\mathcal{F}$ is straightforward given a hierarchical segmentation. The set $\mathcal{E}$ requires a matching strategy to put tubes from different videos in correspondence. In this work we use a simple approach based on the tube appearance and motion features given by the 3D HoG descriptor. Let W be all possible combinations of pairs of videos. We define the set $\mathcal{E}$ as:

$$\mathcal{E} = \bigcup_{(V_1,V_2)\in W} (t_a, t_b) \ \mid \ d(H_a, H_b) < \theta, t_a \in V_1, t_b \in V_2 \tag{7.7}$$

where H_a, H_b refer to the aggregation of the descriptor vectors of tubes t_a and t_b calculated from sampled 3D HoG descriptors, as explained in Section 7.3.3. The function d is any distance metric between descriptor vectors, and θ is a threshold which defines *good* correspondences. In this work we use the Euclidean distance. The prior can also include terms that encourage a smooth labeling in local video areas. This is a typical approach in the segmentation literature which usually helps to reduce noise and improve labeling consistency. We leave this term as an optional feature as we did not observe a significant improvement in our experiments.

7.5 Experiments

In this section we show quantitative results on a subset of the Chroma database [95], in 4 different sequences with the same foreground objects but different backgrounds. In a second experiment we show qualitative results on other typical benchmark videos from the video segmentation literature.

7.5.1 Optimization and Experimental Set-up

We choose to optimize the posterior distribution using graph cuts (step 7 in Algorithm 1), but any other inference or optimization algorithm could be used as well. We apply the logarithm to the probability values and produce scalar unary and pairwise penalties for all possible variable realizations. Every pairwise term is sub-modular, as we only apply a high cost (low probability) on the configurations $[X_i \neq X_j]$. The high probability parameters $\alpha = \gamma$ are set to 0.9 and the low probability parameters $\beta = \delta$ are set to 0.1. The iterative process stops when the percentage of pixels that switch labels between

video	initial.	result
cha-cha1	0.57	0.61
cha-cha2	0.73	0.81
cha-cha3	0.45	0.56
cha-cha4	0.65	0.74

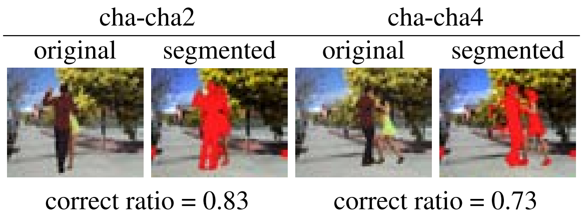


Figure 7.4: Results and example frames on the chroma dataset. In the left, table with results of the four videos of the *cha-cha-cha* class. The central column shows the accuracy of the initialization, and the right column the final labeling. The score is calculated by counting the ratio of pixels correctly labeled.

two iterations is less than 2.5%, or the number of iterations exceeds the maximum of 6. Finally, the threshold on the distance between tube descriptors that defines the set $\mathcal{E}$ is set to 0.4.

7.5.2 *Cha-cha-cha* videos

The Chroma database is possibly the only benchmark video segmentation dataset available that provides videos containing the same instance of an object (only the *cha-cha-cha* class), which is particularly suitable for the co-segmentation problem. To the best of our knowledge, there are no results reported in any major computer vision conference or journal using the Chroma database.

We apply our algorithm to the first 100 frames of the 4 videos of the *cha-cha-cha* class. In Figure 7.4 we show quantitative and qualitative results with respect to the initial labeling. The videos show two people dancing with two different synthetic animated backgrounds. Even though the foreground corresponds to the same object instance in all sequences, the videos are very challenging due to the high clutter and movement of the background. The background appearance and movement changes between different videos (one emulates zooming while others emulate camera translation). This difficulties the tube matching, that performs poorly between pairs of videos with different background. Videos 1 and 3 show a significant lower segmentation accuracy than videos 2 and 4 due to a wrong initialization. This is mainly caused by the background appearance features that confuse the objectness and saliency measures.

7.5.3 Videos from heterogeneous sources

Our method requires several videos depicting an object with similar appearance and motion. Most benchmarking databases do not contain this type of data. Therefore, we selected videos from existing databases and collected other videos from Youtube with similar characteristics in order to perform co-segmentation. We use one video from the *segTrack* [99] database and two from the set provided in [38]. These videos were chosen because similar videos, in terms of appearance and motion of the foreground elements, were available on Youtube.

In this experiment we present qualitative results on three different videos: *kite sur ng*, *ice dancer* and *parachute*. We complete the input sets of three videos for each experiment using the additional Youtube videos. Figure 7.5 shows frame examples of the video results, compared to the works of [38] and [57]. Unfortunately no ground-truth is available in most of the videos.

The performance of our method is comparable to the state-of-the-art results presented in [57]. However, the segmentation accuracy greatly depends on the motion and appearance consistency of the input set, as well as on the quality of the initial labeling. Our method requires good initializations in at least some of the video inputs, so that the learnt foreground/background model is sufficiently representative.

If every input video is badly initialized, the method performs poorly. In cases where the videos downloaded from Youtube do not contain objects with similar motion and appearance, the performance is also hampered, and generally decreasing with every iteration, because the prior constraints are not able to prevent the foreground from expanding or contracting.

The sets of input videos for each benchmarking sequence are shown in the left column of Figure 7.5. We observe noise and segmentation artifacts in some of our result videos (around the kite surfer). This could be avoided applying a local smoothing on the label values. However the performance does not change dramatically due to this constraint, as the foreground appearance model already encourages the labeling to *cluster* around local areas with high likelihood appearance similarity.

7.6 Conclusions

In this work we have presented a novel non-supervised video co-segmentation algorithm. To the best of our knowledge, it is the first method that applies the concept of co-segmentation to video, understood as gathering information from several sources in order to jointly separate foreground and background. We introduce a two layered multi-image model that labels video volumes and image regions simultaneously, by iteratively learning and updating the foreground and background distributions built over motion and appearance features. We provide preliminary experimental validation on a subset of benchmarking video segmentation videos. We also introduce the problem of co-segmentation using heterogeneous video sources. Our method proves to be qualitatively comparable to state-of-the-art results, although we acknowledge that our validation is limited to cases where additional input videos with similar motion and appearance are available.

In the future we would like to generate and release ground-truth on video co-segmentation in order to improve experimental validation for this problem. We would also like to explore the application of video co-segmentation to other computer vision tasks like action recognition or video retrieval.

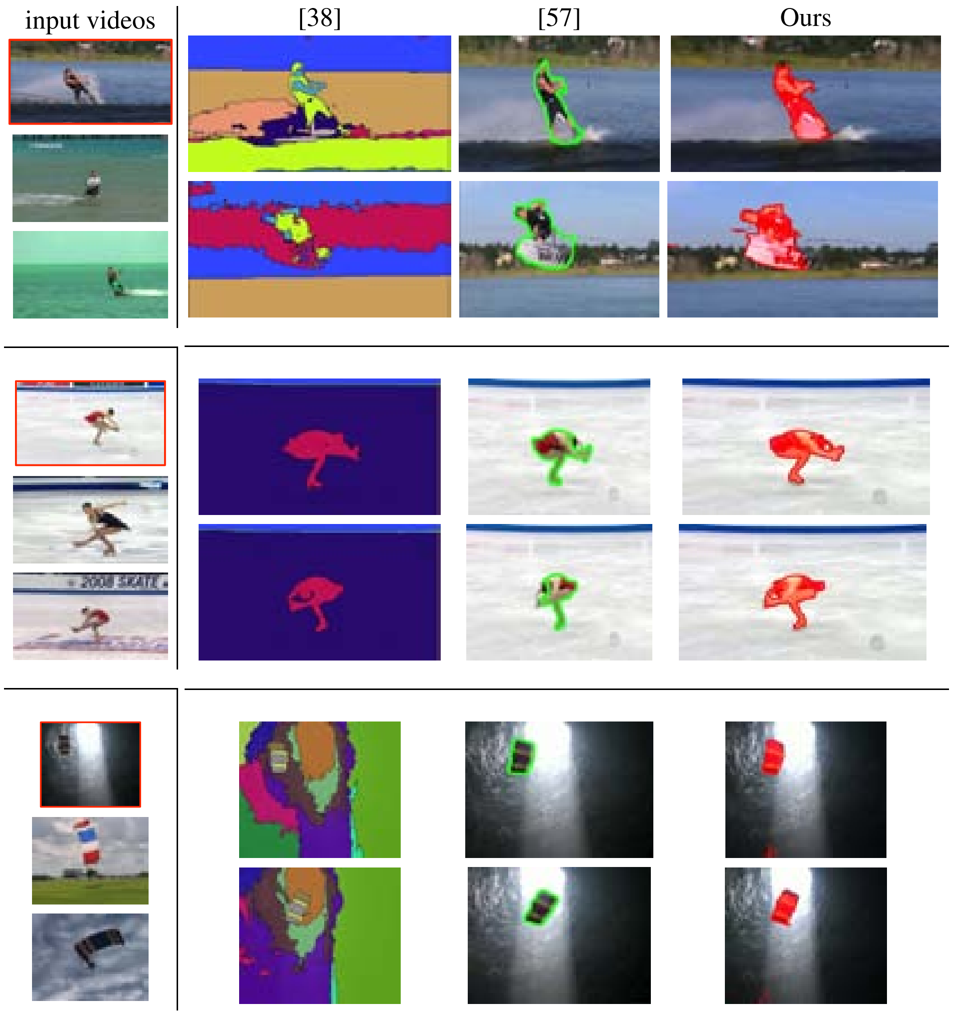


Figure 7.5: Results on three videos from SegTrack [99] and [38]. For each of these videos (framed by a red border), two videos were downloaded from Youtube for the purposes of co-segmentation. One sample frame from each input video is shown in the left column. In the right column we show examples of two frames for each of the result videos. We compare our method with the results obtained by [38] and [57].

Chapter 8

Conclusions

Feature matching is an important problem in several domains of computer vision. In this dissertation we have introduced different techniques to address the problem of matching under the umbrella of probabilistic inference on high-order graphical models. We have demonstrated our approach successfully on three vision tasks related to tracking, image understanding, and segmentation of images and videos.

Among the contributions of this work we would like to stress the exploitation of high-order representations to encode complex structural relationships about the problem at hand. Going beyond the *local* constraints of the typical unary and pairwise models requires taking especial considerations in the optimization process, which has also been addressed in this dissertation. We also would like to emphasize the need of many-to-many correspondences in problems where the one-to-one assumption does not apply, if one takes into account realistic conditions, or the complex nature of the problem itself.

We have shown how point matching between video frames can be effectively used to perform multiple target tracking in the context of a challenging application like headlights tracking. The lack of appearance features, as well as the unpredictable motion of the headlight blobs makes extremely difficult to tackle the tracking problem using classic methods like Kalman Filtering, in the presence of occlusions and targets merging and splitting. However, taking profit of the local structure of the blob arrangements in a many-to-many matching framework leads to satisfactory results. Moreover, learn the likelihood distributions of the observations from training data, as well as codifying a strong prior term using high-order potentials.

We have generalised the initial tracking formulation to diverse tracking scenarios, since we believe that most of the tracking works in the literature are designed for very specific applications. The problem of data association becomes more complicated when considering all possible relationships between targets and observations, together with target occlusions and target-target interactions. We introduce a new *graph of hypothesis* that, using the tracklets generated in our matching process as a starting point, tackles the problem of data association (solving the ambiguity of corresponding targets and observations) that can be applied in different scenarios.

After investigating the problem of point (feature-less) matching in the context of tracking applications, we move towards dense (strongly featured) matching in the context of image segmentation and region matching. We proposed a hierarchical contextual image representation that shows several benefits when applied to the problem of many-to-many matching. We propose a multi-layered image model, that arranges the image elements and structure into two levels according to their semantic value. To perform image matching we proposed a high-order potential that encodes a shape similarity between the semantic elements of the images, and at the same time enforces a coherent matching between the two levels of the model hierarchy. Our experimental results showed that adding these high-order terms benefits the matching accuracy as well as improving the speed of the inference convergence.

By experimenting with the idea of region matching we foresaw other interesting applications that could benefit from this concept of *dense* matching. We explored the problem of image co-segmentation, understood as labeling as foreground or background a set of input images. We proposed an unsupervised iterative algorithm that starting from a inaccurate estimation, dynamically models the foreground and background appearance of a set of images, and labels the pixels using a Markov random field framework. The main problem of such approach is the labeling expansion towards a trivial solution due to a weak initial estimation. We proposed the use of region matching in order to *bound* the figure/ground separation and avoid its continuous expansion each iteration. We show successful results on the iCoseg database, as well as in the MSRC database, that are comparable to state-of-the-art *supervised* methods. The explicit modeling of both foreground and background appearance also allowed overcoming common problems of the co-segmentation literature, such as requiring a high variability on the background appearance among the different input images.

Finally, we extended our work on image co-segmentation to video co-segmentation using the same principles of appearance modeling of foreground and background. However in this case we designed a different multi-video model, with video volumes as labeling elements that were described using motion features. To the best of our knowledge, we are the first to introduce the co-segmentation of multiple videos. We also proposed the use of heterogeneous videos extracted from the internet (Youtube) that was used to improve the segmentation results of a given test video, simply by providing additional videos containing the same or similar instance of the moving object.

As a future work, in the context of headlight tracking, we would like to explore the generation of motion features from the tracking results in order to improve the headlight classification accuracy. Our experiments on high-order region matching presented in Chapter 5 show interesting results regarding the convergence of the inference algorithm. We would like to investigate the convergence properties of belief propagation in such high order experimental set-up, and explore the theoretical explanation behind the faster inference rates shown in our experiments. The idea of region matching between images can also be extended to image categorization, by building a distance kernel using an image similarity measure, extracted form the matching itself or even using the residual energy after the optimization, as in [30]. Regarding the last part of the thesis, we believe that our approach to video-cosegmentation has a high margin of improvement. We would like to apply video-cosegmentation to other applications in which we think it may be beneficial, such as action recognition or video retrieval and classification. We would like to introduce the first benchmarking video co-segmentation database with annotated figure/ground ground-truth masks, and provide extensive validation and a comparative results with other video segmentation algorithms.

List of Publications

This dissertation has led to the following communications:

Journal Papers

- Jose C. Rubio, Joan Serrat, Antonio Lopez, Dani Ponsa. Multiple target tracking for intelligent headlight control. *IEEE Transactions in Intelligent Transportation Systems 2011.*

Conference Contributions

- Jose C. Rubio, Joan Serrat, Antonio Lopez. Video co-Segmentation. *submitted to the Asian Conference on Computer Vision 2012.*
- Jose C. Rubio, Joan Serrat, Nikos Paragios. Image Contextual Representation and Matching through Hierarchies and Higher Order Graphs. *International Conference on Pattern Recognition 2012.*
- Jose C. Rubio, Joan Serrat, Antonio Lopez, Nikos Paragios. Unsupervised co-segmentation through region matching. *IEEE International Conference on Computer Vision and Pattern Recognition 2012.*
- Jose C. Rubio, Joan Serrat, Antonio Lopez. Multiple target tracking and identity linking under split, merge and occlusion of targets and observations. *International Conference On Pattern Recognition Applications and Methods 2012.*
- Jose C. Rubio, Joan Serrat, Antonio Lopez, Dani Ponsa. Multiple target tracking for intelligent headlight control. *IEEE International Conference in Intelligent Transportation Systems 2010.*

Bibliography

[1] Traffic safety facts. White Paper, 2000.

[2] Use of high-beam headlamps. White Paper, 2006.

[3] *Learning Graph Matching*, 2007.

[4] Pablo Alcantarilla et al. Night time vehicle detection for driving assistance lightbeam controller. In *Intelligent Vehicles Symposium, 2008 IEEE*, 2008.

[5] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *IEEE Computer Vision and Pattern Recognition*, 2010.

[6] Nicolas Alt, Christopher Claus, and Walter Stechele. Hardware/software architecture of an algorithm for vision-based real-time vehicle detection in dark environments. In *DATE 08: Proc. of the conf. on Design, Automation and Test i Europe*, 2008.

[7] Dhruv Batra, Adarsh Kowdle, Devi Parikh, Jiebo Luo, and Tsuhan Chen. icoseg: Interactive co-segmentation with intelligent scribble guidance. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[8] Regis Behmo, Nikos Paragios, and Veronique Prinet. Graph commute times for image representation. *IEEE Computer Vision and Pattern Recognition*, 2008.

[9] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis Machine Intelligence*, 2002.

[10] Ben Benfold and Ian Reid. Stable multi-target tracking in real-time surveillance video. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[11] Bernard. *Density Estimation for Statistics and Data Analysis (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman and Hall/CRC, 1986.

[12] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *J. Image Video Process.*, 2008.

[13] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 1st ed. 2006. corr. 2nd printing edition, 2006.

[14] Tiberio S. Caetano, Terry Caelli, and Dante A. C. Barone. Graphical models for graph matching. *in IEEE Computer Vision and Pattern Recognition*, 2004.

[15] Tibério S. Caetano, Terry Caelli, Dale Schuurmans, and Dante Augusto Couto Barone. Graphical models and point pattern matching. *IEEE Trans. Pattern Analysis Machine Intelligence*, 2006.

[16] Kai-Yueh Chang, Tyng-Luh Liu, and Shang-Hong Lai. From co-saliency to co-segmentation: An efficient and fully unsupervised energy minimization model. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[17] Yen-Lin Chen, Chuan-Tsai Lin, Chung-Jui Fan, Chih-Ming Hsieh, and Bing-Fei Wu. Vision-based nighttime vehicle detection and range estimation for driver assistance. In *SMC 08: Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics*, 2008.

[18] Minsu Cho, Young Min Shin, and Kyoung Mu Lee. Co-recognition of image pairs by data-driven monte carlo image exploration. In *European Conference on Computer Vision*, 2008.

[19] Fan Chung and S.-T. Yau. Discrete green's functions. *J. Comb. Theory Ser. A*, 2000.

[20] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Analysis Machine Intelligence*, 2002.

[21] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Arti cial Intelligence*, 2004.

[22] Antonio Criminisi, Geoffrey Cross, Andrew Blake, and Vladimir Kolmogorov. Bilayer segmentation of live video. In *IEEE Computer Vision and Pattern Recognition*, 2006.

[23] Rita Cucchiara, Massimo Piccardi, and Paola Mello. Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Transactions on Intelligent Transportation Systems*, 2000.

[24] Tracking videos web page, 2010. `http://www.cvc.uab.es/adas/IHC/ITSC2010/`.

[25] Paul J. Davis, Elizabeth A. Kosmacek, Yuansheng Sun, Fiorenza Ianzini, and Michael A. Mackey. The Large-Scale digital cell analysis system: an open system for nonperturbing live cell imaging. *Journal of Microscopy*, 2007.

[26] R. DeFauw, S. Lakshmanan, and K.V. Prasad. A system for small target detection, tracking, and classification. In *ITSC 99: Proc. of then IEEE Int. Conf. on Intelligent Transportation Systems*, 1999.

[27] R. Delgado-Gonzalo, N. Dénervaud, S. Maerkl, and M. Unser. Multi-target tracking of packed yeast cells. In *Proceedings of the Seventh IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI 10)*, 2010.

[28] Sven Dickinson. The evolution of object categorization and the challenge of image abstraction, 2009.

[29] Sven J. Dickinson, Ali Shokoufandeh, Yakov Keselman, M. Fatih Demirci, and Diego Macrini. Object categorization and the need for many-to-many matching. In *DAGM-Symposium*, 2005.

[30] Olivier Duchenne, Armand Joulin, and Jean Ponce. A graph-matching kernel for object categorization. In *International Conference on Computer Vision*, 2011.

[31] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 2004.

[32] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 2005.

[33] Jonas Firl, Marko Hoerter, Martin Lauer, and Christoph Stiller. Vehicle detection, classification and position estimation based on monocular video data during night-time. In *ISAL 09: Proc. of the 8th Int. Symposium on Automotive Lighting*, 2009.

[34] Andrea Fossati, Patrick Schönmann, and Pascal Fua. Real-time vehicle tracking for driving assistance. *Machine Vision and Applications*, 2010.

[35] Andrew C. Gallagher and Tsuhan Chen. Clothing cosegmentation for recognizing people. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[36] Daniel Glasner, Shiv Naga Prasad Vitaladevuni, and Ronen Basri. Contour-based joint clustering of multiple segmentations. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[37] S. Göormer, D. Müller, S. Hold, M. Meuter, and A. Kummert. Vehicle recognition and ttc estimation at night based on spotlight pairing. In *ITSC 09: Proc. of the IEEE Intelligent Transportation Systems Conf.*, 2009.

[38] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan A. Essa. Efficient hierarchical graph-based video segmentation. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[39] J. M. Hammersley and P. Clifford. Markov field on finite graphs and lattices. 1971.

[40] Varsha Hedau, Himanshu Arora, and Narendra Ahuja. Matching images under unstable segmentations. *IEEE Computer Vision and Pattern Recognition*, 2008.

[41] Dorit S. Hochbaum and Vikas Singh. An efficient algorithm for co-segmentation. In *International Conference on Computer Vision*, 2009.

[42] Yuchi Huang, Qingshan Liu, and Dimitris N. Metaxas. Video object segmentation by hypergraph cut. In *IEEE Computer Vision and Pattern Recognition*, 2009.

[43] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Analysis Machine Intelligence*, 1998.

[44] Amirali Jazayeri, Hongyuan Cai, Jiang Yu Zheng, and Mihran Tuceryan. Vehicle detection and tracking in car video based on motion model. *IEEE Transactions on Intelligent Transportation Systems*, 2011.

[45] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[46] Armand Joulin, Francis R. Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[47] Zia Khan, Tucker Balch, and Frank Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *European Conference on Computer Vision*, 2003.

[48] Jaechul Kim and Kristen Grauman. Asymmetric region-to-image matching for comparing images with generic object categories. *IEEE Computer Vision and Pattern Recognition*, 2010.

[49] Alexander Kläser, Marcin Marszalek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *In British Machine Vision Conference*, 2008.

[50] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Clustering via lp-based stabilities. In *Neural Information Processing Systems*, 2008.

[51] Adarsh Kowdle, Dhruv Batra, W.C. Chen, and Tsuhan Chen. imodel: interactive co-segmentation for object of interest 3d modeling. In *European Conference on Computer Vision*, 2010.

[52] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 2001.

[53] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 1955.

[54] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 1955.

[55] Tinne De Laet, Herman Bruyninckx, and Joris De Schutter. Shape-based online multitarget tracking and detection for targets causing multiple measurements: Variational bayesian clustering and lossless data association. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2011.

[56] Xiangyang Lan, Stefan Roth, Daniel P. Huttenlocher, and Michael J. Black. Efficient belief propagation with learned higher-order markov random fields. In *European Conference on Computer Vision*, 2006.

[57] Yong Jae Lee, Jaechul Kim, and Kristen Grauman. Key-segments for video object segmentation. In *International Conference on Computer Vision*, 2011.

[58] Marius Leordeanu and Martial Hebert. A spectral technique for correspondence problems using pairwise constraints. In *International Conference on Computer Vision*, 2005.

[59] Alex Levinshtein, Cristian Sminchisescu, and Sven J. Dickinson. Optimal contour closure by superpixel grouping. In *European Conference on Computer Vision*, 2010.

[60] Alex Levinshtein, Cristian Sminchisescu, and Sven J. Dickinson. Spatiotemporal closure. In *ACCV*, 2010.

[61] Kang Li, Mei Chen, and Takeo Kanade. Cell population tracking and lineage construction with spatiotemporal context. In *Proceedings of the 10th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2007.

[62] Yin Li, Jian Sun, and Heung-Yeung Shum. Video object cut and paste. *ACM Trans. Graph.*, 2005.

[63] Ying Li and Sharath Pankanti. Intelligent headlight control using camera sensors. In *UCVP 09: Proc. of the Workshop on Use of Context in Vision Processing*, 2009.

[64] M. Liu, A.K. Roy Chowdhury, and G.V. Reddy. Robust estimation of stem cell lineages using local graph matching. In *MMBIA09*, 2009.

[65] Antonio López et al. Nighttime vehicle detection for intelligent headlight control. In *Int. Conf. on Advanced Concepts for Intelligent Vision Systems*, 2008.

[66] Antonio López et al. Temporal coherence analysis for intelligent headlight control. In *Proc. of the IROS 08 2nd Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, 2008.

[67] Michael Maire, Pablo Arbelaez, Charles Fowlkes, and Jitendra Malik. Using contours to detect and localize junctions in natural images. In *IEEE Computer Vision and Pattern Recognition*, 2008.

[68] Julian John McAuley, Tibério S. Caetano, and Marconi S. Barbosa. Graph rigidity, cyclic belief propagation, and point pattern matching. *IEEE Trans. Pattern Analysis Machine Intelligence*, 2008.

[69] Julian John McAuley, Tibério S. Caetano, and Alexander J. Smola. Robust near-isometric matching via structured learning of graphical models. In *Neural Information Processing Systems*, 2008.

[70] Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 2010.

[71] Lopamudra Mukherjee, Vikas Singh, and Chuck R. Dyer. Half-integrality based algorithms for cosegmentation of images. In *IEEE Computer Vision and Pattern Recognition*, 2009.

[72] Lopamudra Mukherjee, Vikas Singh, and Jiming Peng. Scale invariant cosegmentation for image groups. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[73] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *In Proceedings of Uncertainty in AI*, 1999.

[74] Peter Nillius, Josephine Sullivan, and Stefan Carlsson. Multi-target tracking - linking identities using bayesian network inference. In *In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2006.

[75] Ronan O'Malley, Edward Jones, and Martin Glavin. Rear-lamp vehicle detection and tracking in low-exposure color video for night conditions. *Trans. Intell. Transport. Sys.*, 2010.

[76] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[77] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *In IEEE Computer Vision and Pattern Recognition*, 2007.

[78] Huaijun Qiu and Edwin R. Hancock. Graph simplification and matching using commute times. *Pattern Recognition*, 2007.

[79] Deva Ramanan. Learning to parse images of articulated bodies. In *Neural Information Processing Systems*, 2006.

[80] Julien Rebut, Benazouz Bradai, Julien Moizard, and Adrien Charpentier. A monocular vision based advanced lighting automation system for driving assistance. In *ISIE 09: IEEE Int. Symp. on Industrial Electronics*, 2009.

[81] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 1979.

[82] Mark A. Rizzo, Michael W. Davidson, and David W. Piston. Fluorescent protein tracking and detection: Fluorescent protein structure and color variants. *Cold Spring Harb Protoc*, 2009.

[83] Carsten Rother, Pushmeet Kohli, Wei Feng, and Jiaya Jia. Minimizing sparse higher order energy functions of discrete variables. In *IEEE Computer Vision and Pattern Recognition*, 2009.

[84] Carsten Rother, Vladimir Kolmogorov, Tom Minka, and Andrew Blake. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs. In *IEEE Computer Vision and Pattern Recognition*, 2006.

[85] Jose C. Rubio, Joan Serrat, Antonio L'opez, and Daniel Ponsa. Multiple-target tracking for intelligent headlights control. In *ITSC 10: Proc. of then IEEE Int. Conf. on Intelligent Transportation Systems*, 2010.

[86] Jose C. Rubio, Joan Serrat, and Antonio M. López. Multiple target tracking and identity linking under split, merge and occlusion of targets and observations. In *International Conference on Pattern Recognition Aapplications and Methods*, 2012.

[87] Jose C. Rubio, Joan Serrat, Antonio M. López, and Nikos Paragios. Image contextual representation and matching through hierarchies and higher order graphs. *International Conference on Pattern Recognition*, 2012.

[88] Jose C. Rubio, Joan Serrat, Antonio M. López, and Nikos Paragios. Unsupervised cosegmentation through region matching. *IEEE Computer Vision and Pattern Recognition*, 2012.

[89] Jose C. Rubio, Joan Serrat, Antonio M. López, and Daniel Ponsa. Multiple-target tracking for intelligent headlights control. *IEEE Transactions on Intelligent Transportation Systems*, 2012.

[90] Eli Shechtman and Michal Irani. Matching local self-similarities across images and videos. In *IEEE Computer Vision and Pattern Recognition*, 2007.

[91] Erik B. Sudderth, Antonio Torralba, William T. Freeman, and Alan S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *In IEEE Intl. Conf. on Computer Vision*, 2005.

[92] Narayanan Sundaram and Kurt Keutzer. Long term video segmentation through pixel level spectral clustering on gpus. In *International Conference on Computer Vision Workshops*, 2011.

[93] Charles Sutton and Andrew Mccallum. Improved dynamic schedules for belief propagation. In *Conference on Uncertainty in Arti cial Intelligence (UAI)*, 2007.

[94] T.Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, 1983.

[95] Fabricio Tiburzi, Marcos Escudero, Jesús Bescós, and José Mar´a Mart´nez Sanchez. A ground truth for motion-based video-object segmentation. In *International Conference on Image Processing*, 2008.

[96] Sinisa Todorovic and Narendra Ahuja. Region-based hierarchical image matching. *International Journal of Computer Vision*, 78:47–66, 2008. 10.1007/s11263-007-0077-5.

[97] Lorenzo Torresani, Vladimir Kolmogorov, and Carsten Rother. Feature correspondence via graph matching: models and global optimization. In *European Conference on Computer Vision*, 2008.

[98] Alexander Toshev, Jianbo Shi, and Kostas Daniilidis. Image matching via saliency region correspondences. *IEEE Computer Vision and Pattern Recognition*, 2007.

[99] David Tsai, Matthew Flagg, and James M. Rehg. Motion coherent tracking with multi-label mrf optimization. In *British Machine Vision Conference*, 2010.

[100] Hwann tzong Chen, Horng-Horng Lin, and Tyng luh Liu. Multi-object tracking using dynamical graph matching. In *In IEEE Computer Vision and Pattern Recognition*, 2001.

[101] Joost van de Weijer and Cordelia Schmid. Coloring local feature extraction. In *European Conference on Computer Vision*, 2006.

[102] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Cosegmentation revisited: Models and optimization. In *European Conference on Computer Vision*, 2010.

[103] Sara Vicente, Carsten Rother, and Vladimir Kolmogorov. Object cosengmentation. In *IEEE Computer Vision and Pattern Recognition*, 2011.

[104] Shiv Naga Prasad Vitaladevuni and Ronen Basri. Co-clustering of image segments using convex optimization applied to em neuronal reconstruction. In *IEEE Computer Vision and Pattern Recognition*, 2010.

[105] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *In AISTATS*, 2003.

[106] Zeng-Fu Wang and Zhi-Gang Zheng. A region based stereo matching algorithm using cooperative optimization. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2008.

[107] Zheng Wu, Thomas H. Kunz, and Margrit Betke. Efficient track linking methods for track graphs using network-flow and set-cover thechniques. *IEEE Computer Vision and Pattern Recognition*, 2011.

[108] Mikhail Zaslavskiy, Francis Bach, and Jean-Philippe Vert. Many-to-many graph matching: a continuous relaxation approach. In *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, 2010.

[109] Yun Zeng, Chaohui Wang, Yang Wang, Xianfeng Gu, Dimitris Samaras, and Nikos Paragios. Dense non-rigid surface registration using high-order graph matching. *IEEE Computer Vision and Pattern Recognition*, 2010.