POLYTHECNIC UNIVERSITY OF CATALUNYA

MASTER THESIS

# Accurate video object tracking using a region-based particle filter

*Author:*
Andreu Girbau Xalabarder

*Supervisors:*
Ferran Marqués Acosta
David Varas González

*A thesis submitted in fulfillment of the requirements for the degree of Master in Telecommunication Engineering*

*in the*

Image Processing Group
Signal Theory and Communications Department

February 6, 2017

# Abstract

Usually, in particle filters applied to video tracking, a simple geometrical shape, typically an *ellipse*, is used in order to bound the object being tracked. Although it is a good tracker, it tends to a bad object representation, as most of the world objects are not simple geometrical shapes. A better way to represent the object is by using a region-based approach, such as the *Region Based Particle Filter* (RBPF) [1]. This method exploits a hierarchical region based representation associated with images to tackle both problems at the same time: tracking and video object segmentation.

By means of RBPF the object segmentation is resolved with high accuracy, but new problems arise. The object representation is now based on image partitions instead of pixels. This means that the amount of possible combinations has now decreased, which is computationally good, but an error on the regions taken for the object representation leads to a higher estimation error than methods working at pixel level. On the other hand, if the level of regions detail in the partition is high, the estimation of the object turns to be very noisy, making it hard to accurately propagate the object segmentation.

In this thesis we present new tools to the existing RBPF. These tools are focused on increasing the RBPF performance by means of guiding the particles towards a good solution while maintaining a particle filter approach. The concept of *hierarchical flow* is presented and exploited, a *Bayesian estimation* is used in order to assign probabilities of being object or background to each region, and the reduction, in an intelligent way, of the solution space $\Omega$, to increase the RBPF robustness while reducing computational effort. Also changes on the already proposed co-clustering in the RBPF approach are proposed.

Finally, we present results on the recently presented DAVIS database [2]. This database comprises 50 High Definition video sequences representing several challenging situations. By using this dataset, we compare the RBPF with other state-of-the-art methods.

# Acknowledgements

First of all, I would like to thank professor Ferran Marques and PhD (at last!) David Varas for this 2 and a half years on the Image Processing Group. They have taught me a lot, had lot of patience, introduced me to this great world of investigation and to sushi. Thank you.

To UPC ETSETB in general, for this 6 years, to show me to overcome my own goals. Also to the Image Processing Group, specially to Albert, who helped me in everything computer-related I needed.

To my colleagues, to my friends, and professors. A list of all the people who is important to me in some way would be too extensive, so I would like to thank you all for these years. Also a special mention to *Els competitius* from Barcelona, who made this years a pleasure.

To my family. To Lluís, Eulàlia, Maria, Joan, Mercè, Joaquim, Anna, Pere, Xavier, Núria and Tomàs. Thank you. For everything.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **RBPF** | **R**egion **B**ased **P**article **F**ilter |
| **PDF** | **P**robability **D**ensity **F**unction |
| **MC** | **M**onte **C**arlo |
| **SIS** | **S**equential **I**mportance **S**ampling |
| **SIR** | **S**equential **I**mportance **R**esampling |
| **UCM** | **U**ltrametric **C**ontour **M**ap |
| **COB** | **C**onvolutional **O**riented **B**oundaries |
| **gPb** | **G**lobalized **P**obability of **B**oundary |
| **OF** | **O**ptical **F**low |
| **DAVIS** | **D**ensely **A**nnotated **VI**deo **S**egmentation |

To my family.

# 1 Introduction

## 1.1 Introduction

A classical approach on particle filters in video object tracking is to use geometrical shapes as particles to estimate an object position, typically ellipses. Although the use of simple geometrical shapes largely simplify the problem, particle filters still present high complexity in terms of the amount of possible solutions to the estimation problem, as their solution space $\Omega$ is really huge; $\Omega$ represents the parameterization of every possible ellipse that can be represented in a frame. Usually, this is palliated by convoluting a Gaussian kernel over the object estimation, to reduce the dimensionality of the problem, in order to keep the particles near the object of interest.

In our case, [1] defines a particle as a union of regions to improve the object estimation; not only to provide the object position but also to segment the object accurately. This new definition of particle increases the problem complexity, hence some steps of classical particle filters are put under revision as the behavior of the overall algorithm is different. For instance, it is typical in such filters to apply some Gaussian noise in order to perturbate the particles in a random manner, but for this case it is not possible, and new perturbation approaches are studied.

Applying the *Region Based Particle Filter* (RBPF) a better estimation of the object is achieved, but the problem of dimensionality remains. For each particle a world of possible solutions is presented as sets of regions instead of geometrical shapes. It is true that such space has been reduced in comparison to the classical particle filters, as the combination of all the regions is lower than the combination of all the possible representations of an ellipse, but, still, the amount of possible solutions is really huge. Also, the reduction of $\Omega$ has to be done in an intelligent manner, as removing a good possible solution of this space could be critical to the overall object estimation.

This thesis focuses on providing new tools to the existing RBPF. These tools are focused on increasing the RBPF performance by means of guiding the particles towards a good solution while maintaining a particle filter approach. *Hierarchical flow* and a *Bayesian estimation*, which lead to a reduction, in an intelligent way, of the solution space $\Omega$ dimensionality are some of the tools introduced to the RBPF in this thesis.

## 1.2 Contribution

The main contribution of this thesis is to provide new tools to the already presented *Region Based Particle Filter* (RBPF). These new tools are both to improve the accuracy

of this method and to reduce the solution space in order to provide an intelligent search to get to a good final object estimation. Moreover, the algorithm has been reprogrammed and optimized to fit the new tools to the RBPF.

Finally, this master thesis includes results on the newly proposed DAVIS dataset [2]. Also the code will be public on github in order to be evaluated and published at DAVIS web page between other state-of-the-art results.

## 1.3 Thesis outline

The structure of this thesis is organized as follows.

Chapter 2 provides state-of-the-art methods on object tracking while making inference on some computer vision applications related to our work.

Chapter 3 is devoted to the Region Based Particle Filter, and provides some tools used in such method. In this chapter the proposed changes to the existing RBPF are emphasized, as they are the main objective of this thesis.

Chapter 4 contains the evaluation of the DAVIS database. It is divided on 2 sections, training and evaluation. In this chapter, test results and performance comparison with the state-of-the-art methods are provided.

Finally, in chapter 5, the conclusions of the thesis are summarized and the future work is presented.

# 2 Background

## 2.1 Computer Vision applications

Computer vision applications may vary from a wide range. To provide examples of such applications we can state robotics, autonomous cars, surveillance... Our aim is the analysis of objects in video sequences. This computer vision application aims to provide information about an object of interest to a computer; to extract information about it at different levels. For us, such levels of object analysis are tracking and video object segmentation, to provide the object position and the object shape accurately. An approach on solving this kind of problems are the *particle filters*.

Generic particle filters [3] estimate an object as a geometrical shape, typically an *ellipse*. Although these methods are easily implemented and can work in real time, their accuracy may not be good enough for certain applications. In this thesis a method for doing both (tracking and video segmentation) is used in order to provide the position of the object and its accurate shape at the same time; this method is called *Region Based Particle Filter* (RBPF).

Based on exploiting the segmentation of a pair of images, RBPF [1] aims to propagate an object mask over a video sequence in an accurate manner, at a superpixel level.

## 2.2 State of the art

RBPF is based on segmenting an object of interest through a video sequence starting from an initial mask defined either by a user or a computer. Methods like this, which take in account an initial or several user interactions, are called *semi-supervised* methods. Some of these state-of-the-art methods, with which we compare to, are published on [2]. All of them have available public code.

A common approach to solve the video tracking and object segmentation problem is to perform an optimization of an energy defined over a graph structure [4]. To model long-range spatiotemporal connections along the video, some approaches use fully connected graphs [5]; others model the spatiotemporal consistency over a pair of frames optimizing over a graph structure of an image patch containing the object [6] or over non-successive frames [7].

Although these methods produce large coherence in the segmentation of objects, the computational complexity may be intractable in some cases. This complexity is efficiently reduced by means of minimizing the energy using a graph-cut over a bilateral space in [8], while the temporal consistency is maintained.

Another approach to model the temporal consistency is to make use of temporally consistent superpixels [9] in order to propagate them to the next frame while maintaining the temporal correlation. Our approach is to relate coherent combinations of regions from consecutive partitions in order to estimate the shape of the object with a single optimization.

Some of the presented methods make use of the full video sequence [5] [8], a subset of frames [4] or, like us, a frame-by-frame approach, where only the previous and the current frames are taken in account [1] [7] [6] [9].

# 3 Region-Based Particle Filter

## 3.1 System Overview

Many problems in science require estimation of the state of a system that changes over time. This state is not known or cannot be directly accessed (hidden state), so we need an estimation of it. This estimation is calculated via a sequence of noisy measurements. In the Bayesian approach to dynamic state estimation the goal is to construct the posterior *probability density function* (pdf) of the state based on a set of observed noisy measurements.

Particle filters are a generic type of *Monte Carlo* (MC) methods, which are characterized by obtaining numerical results by repeated random sampling. MC methods are mainly used in three distinct problem classes: optimization, numerical integration, and generating draws from a probability distribution. For our problem, the last usage of the MC approach is the one to follow.

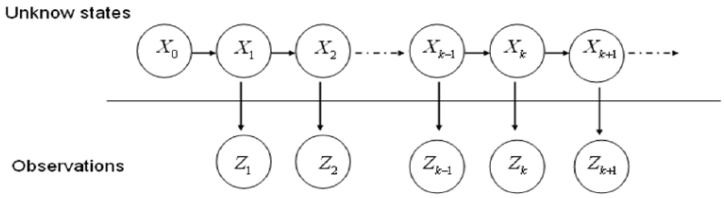


FIGURE 3.1: Bayesian estimation scheme. $x_k$ represents the hidden state that can be estimated via the measurement $z_{1:k}$

The Sequential Importance Sampling (SIS) algorithm is a recursive MC method that forms the basis of the generic particle filter algorithm. Recursive Bayesian filtering may also be implemented using this technique. The key idea of this algorithm is to represent the objective pdf $p(x_k|z_{1:k})$, where $x_k$ is the state estimation in time $k$ $\forall k \in \mathbb{N}$ and $z_{1:k}$ are the measurements from time 1 to time $k$, defined in the tracking problem by a set of random samples with their associated weights. The posterior $p(x_k|z_{1:k})$ can be approximated as:

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_s} w_k^{(i)} \cdot \delta(x_{1:k} - x_{1:k}^{(i)}) \tag{3.1}$$

where weights $w_k^{(i)}$ are chosen using *importance sampling* [10]. As the posterior is computed sequentially, weights can be expressed as [3]:

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(z_k|x_k^{(i)}) \cdot p(x_k^{(i)}|x_{k-1}^{(i)})}{q(x_k^{(i)}|x_{k-1}^{(i)}, z_k)} \tag{3.2}$$

where $q(x_k^{(i)}|x_{k-1}^{(i)}, z_k)$ is called *importance sampling*.

A typical particle filter is the color-based particle filter. In this classical approach, the state $x_k$ is parameterized as a geometrical shape, e.g. an ellipse with parameters $\{x, y, w, h\}$, being $(x, y)$ the position of the center and $(w, h)$ its width and height.

$$x_k = \{x, y, w, h\} \tag{3.3}$$

$$z_k = I_k \tag{3.4}$$

where $(x, y)$ is the object position, $(w, h)$ are the axis lengths of the geometrical shape, and $I_k$ is the image or frame at time $k$.

The most common particle filter used for tracking is the *Sampling Importance Resampling* (SIR) filter proposed by [11]. In this SIR filter, the choice of importance density is $q(x_k^{(i)}|x_{k-1}^{(i)}, z_k) = p(x_k^{(i)}|x_{k-1}^{(i)})$. This choice states that each particle at time $k$ is drawn from a function that only depends on the particle at $k-1$. If we substitute to equation 3.2:

$$w_k^{(i)} \propto w_{k-1}^{(i)} p(z_k|x_k^{(i)}) \tag{3.5}$$

Also a *resampling step* is applied at every time instant. A new set of particles $S_k' = \{x_k^{'(1)}, x_k^{'(2)}, ..., x_k^{'(N_s)}\}$ is created by resampling with replacement $N_s$ times from the approximate discrete representation of $p(x_{k-1}^{(i)}|z_{1:k-1})$. The result is an i.i.d sampling of the function introduced in equation 3.1. Then, the weights become to be represented by a uniform distribution $w_k^{(i)} = 1/N_s$, and the expression to compute the new weights in $k$ becomes:

$$w_k^{(i)} \propto p(z_k|x_k^{(i)}) \tag{3.6}$$

SIR method tracks objects comparing the histogram of the pixels that lay inside a geometrical shape, representing the object state and the histogram of the object model.

As it is explained in equation 3.3, a typical approach on classical particle filters is to get as the state representation a geometrical shape. Therefore, the solution space $\Omega$ turns to be really big, taking in account all the possible combinations a simple figure like an ellipse would have (in terms of positioning and shape). More detailed information on theory about classical particle filters can be found at [3] and [1].

### 3.1.1 State and measurement definition

Let us define a new representation of both the state and measurement for the tracking problem in terms of regions. In our work, states are formed by a union of regions from the image partition while measurements consider both the image and its associated partition.

$$x_k = \bigcup_{r}^{n_k^o} R_k^r \tag{3.7}$$

$$z_k = [I_k, Pk] \tag{3.8}$$

where $P_k = \bigcup_{r=1}^{n_k} R_k^r$ is a partition of the image $I_k$, $n_k$ is the number of regions that form the partition and $n_k^o$ is the number of regions that characterize the object with $n_k^o \leq n_k$.

Given that the state estimation $x_k$ is formed using a set of regions from an image partition $P_k$, several object representations that could be computed at pixel level are not allowed. This means that, in comparison with the classical particle filter approach at pixel level, the solution space $\Omega$ is largely reduced. If all the possible unions of regions are considered, the solution space size is:

$$|\Omega| = \sum_{n=1}^{n_k} \frac{n_k!}{n!(n_k - n)!} \tag{3.9}$$

where $n_k$ represents the amount of regions in the partition. These possible solutions are represented in a summation because, even the solution space is huge, it is still finite.

In this approach the SIR methodology is also used. Also the histogram comparison is used in order to represent the object state and the pdf of the object model. The color histogram of the pixels inside a mask generated by the union of regions is used as it is a direct mapping to a pdf, as in the Bayesian approach to dynamic state estimation the goal is to construct such pdf.

This method relies on estimating the state from the initial frame in time $k = 0$ to any possible $k$ inside the video sequence. For doing so, the initial object model $O_k$ for $k = 0$ is computed from taking the color histogram from an initial known state (defined by the user or the ground truth). The initial background model $B_k$ is also computed as it will be used in section 3.4. Both models will be updated at every time step if the mean weight $w_k^{(i)}$ of the particles $P_k^{(i=1:N_s)}$ is higher than a threshold $w_{th}$. The model update is settled as:

$$O_k = (1 - \alpha)O_{k-1} + \alpha h(A_{k-1}^M) \tag{3.10}$$

where $\alpha$ is the *forgetting factor* and it is defined as $0 \leq \alpha \leq 1$, $h(A_{k-1}^M)$ is the histogram of the estimated mask of the object state in equation 3.22. The background model $B_k$ is updated by means of a bounding box over $A_{k-1}^M$, taking the pixels outside the estimated mask.

## 3.2 Hierarchical segmentation

The state representation on RBPF leads to a discussion. In the area of segmentation, hierarchical techniques have proven to produce the best frameworks. A *hierarchy*

gives the representation of an image at different detail levels. In this work we use *Convolutional Oriented Boundaries* (COB) [12] to produce a hierarchical segmentation on every frame. In previous work, *Globalized Probability of Boundary* (gPb) [13] was used to provide the image partition. A proposal of this thesis is to use COB instead of gPB, as COB provides a better contour estimation to our problem; it focuses on the object outer boundaries while reducing inner contours, instead on evaluating every contour independently. This new approach leads to a possible solution on making the system faster and more robust, and on reducing the solution space $\Omega$ by means of *hierarchical flow*.

We define *hierarchical flow* as the movement through different hierarchical levels of a partition. This makes us able to access to the information on several hierarchical levels in order to, somehow, exploit it.

A question arises: which is the optimal hierarchy level to this estimation problem? Is there an optimal level? Several factors will have to be taken in account: which hierarchical level leads to the best state estimation? Which reduces the computational cost (traduced in time spent in the process)? And which provides best robustness to the system?

Some examples on this would be: the lower the hierarchy level, the best possible state estimation (as the resolution in terms of regions would be the highest possible), but more computational effort and noise in the system. On the other hand, the higher in the hierarchy, the worse the estimation will be but much less time will be spent to do the calculations, as the contour elements will be reduced in front of a lower-level partition of the hierarchy.

FIGURE 3.2: Different level of partition hierarchy. From left to right from the lowest hierarchy level $\lambda = 0$ to a higher one (rightmost) $\lambda = 0.15$ We can observe the amount of combinations (complexity) and the noise the lowest hierarchy offers and the segmentation errors at a higher (rightmost) one. The upper figures are the images with the partition overlay and the lower ones represent the segmentation hierarchy by means of the UCM map.

Our segmentation is built in the concept of *Ultrametric Contour Map* (UCM) [13]. This technique, combined with COB, generates contours with a certain strength or energy $\lambda$. We *flow* through the hierarchy by thresholding the UCM map by a $\lambda_{th}$, generating a new segmentation based on a *cut* ($\lambda \geq \lambda_{th}$) on the hierarchy.

In order to get to a good segmentation provided by a hierarchy level we build two new segmentations based on two cuts: $\lambda(J)$ and $\lambda(J, w)$, where $J$ corresponds to the Jaccard index and $w$ to the weight associated to a mask introduced below. These metrics provide a limit on the *hierarchical flow* in order to not to get to an invalid cut

that would generate a bad partition representation for a correct object estimation. Two segmentations are computed for robustness and computation purposes; the partition will be represented by the highest possible level on the hierarchy (typically achieved by $\lambda(J, w)$) unless this level leads to a bad propagation on the *co-clustering* step. Further details are explained over this section and co-clustering section 3.3.

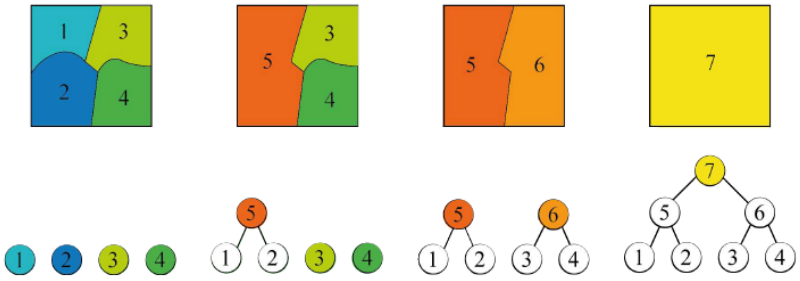


FIGURE 3.3: Partitions generated with mergings of regions from the initial leaves partition $P^{\lambda=0}$ (leftmost figure).

These new segmentations are computed using the Jaccard index $J$ and the mask $A^M_{k-1}$ weight $w_{k-1}$, where $A^M_{k-1}$ represents the estimated state in $k-1$. First, $A^M_{k-1}$ is propagated via optical flow (OF) to the next time step $k$.

$$M_k^{OF} = OF(A^M_{k-1}) \tag{3.11}$$

After this, the propagated mask $M_k^{OF}$ is adapted to the regions of the partition, as it was propagated on pixel level. This is done via thresholding on the amount of pixels propagated inside a region against the amount of pixels of the same region. The partition corresponding to this new mask $M_k^{R_\lambda}$ will be the leaves partition, which is the lowest-level hierarchy at cut $\lambda = 0$. The reference weight $w_{ref}$ is computed by means of comparing $M_k^{R_{\lambda=0}}$ color histogram to the object model $O_{k-1}$ using equations 3.20 and 3.21.

$$M_k^{R_\lambda} = P_k^{\lambda}(M_k^{OF}) \tag{3.12}$$

where $Pk^{\lambda}$ is the partition in time $k$ giving a cut on the hierarchy provided by $\lambda$.


FIGURE 3.4: Propagation from $k-1$ to $k$ by means of Optical Flow. Then, the fitting to the partition $P_k$ is performed.

Once the propagated-confined mask $M_k^{R_\lambda}$ is calculated, the Jaccard index of $M_k^{R_\lambda}$ against $M_k^{OF}$ will be denoted as the reference Jaccard index $J_{ref}$.

$$J(M_k^{OF}, M_k^{R_\lambda}) = \frac{|M_k^{OF} \cap M_k^{R_\lambda}|}{|M_k^{OF} \cup M_k^{R_\lambda}|} \tag{3.13}$$

This relation between masks will be used as our comparison metric for next steps. After this, $M_k^{R_\lambda}$ with $\lambda = 0$ will be used as a starting point. For each step $\lambda+\Delta\lambda$, $M_k^{R_\lambda}$ will be recalculated by means of thresholding using equation 3.12. The resulting mask $M_k^{R_\lambda}$ for a certain $\lambda$ will produce a Jaccard index $J_\lambda$ on its relation to $M_k^{OF}$ using equation 3.13. If the relation between Jaccards is $\frac{J_\lambda}{J_{ref}} < J_{th}$, where $J_{th}$ is the permitted difference between both Jaccards, then $\lambda(J)$ is settled as the previous $\lambda$, where the relation between Jaccards was higher than $J_{th}$.

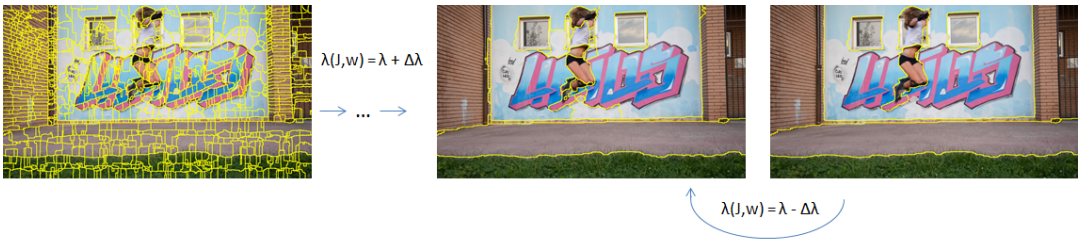


FIGURE 3.5: Representation of hierarchical movement. From a low hierarchy level $\lambda = 0$ at the left picture to higher ones. When the partition representation leads to a bad object representation the process stops and takes the last result as the partition.

For the second cut, $\lambda(J, w)$, the procedure is the same as $\lambda(J)$, but taking in account the weight $w_\lambda$ the mask would have in comparison with the object model $O_{k-1}$. If $\frac{w_\lambda}{w_{ref}} < w_{th}$, being $w_{th}$ the allowed difference between both weights, will mean that the mask represents a worse estimation than the $M_k^{R_\lambda}$ computed before and the process will stop, being $\lambda(J, w)$ equal to the previous $\lambda$. On the other hand, if $\frac{w_\lambda}{w_{ref}} > 1$ will mean that the estimation is better than the one that produced $w_{ref}$ and the process will continue, actualizing $w_{ref} = w_\lambda$. There is a condition on the Jaccard though. If the relation between Jaccards $\frac{J_\lambda}{J_{ref}}$ goes below a threshold $J_{th}^{'}$, where $J_{th}^{'} < J_{th}$, then the sequence will stop and $\lambda(J, w)$ will be equal to the previous $\lambda$.

## 3.3 Co-clustering

A huge difference between classical particle filters and RBPF is in the propagation step. Classical particle filters propagation is implicit in the perturbation step, as the particles are not constrained to an specific movement. However, these particles are a simple geometrical shape, and they can be perturbed in a quite easy way (just by applying noise to their position and shape).

In the RBPF approach a propagation step to relate regions from $P_{k-1}$ and $P_k$ is required, as a propagation by means of the classical particle filter perturbation is not possible. *Contour-based joint clustering* (co-clustering) [14] provides a joint segmentation between 2 different segmentations, and [1] a method to reduce the huge computational effort it requires. As in this case we tackle the tracking and segmentation problem at the same time step, we use this tool to propagate every particle from consecutive image pairs (from $k-1$ to $k$) in order to predict the object movement. For a detailed explanation on this algorithm [15] describes all the steps followed by the co-clustering method and provides examples on each one.

### 3.3.1 Particle Support Partition

To propagate all the particles at the same time, [1] presents a very nice approach. A new partition called *particle support partition* $P_{k-1}^S$ is defined. This support partition will gather the information of all the particles in order to propagate them in one step. $P_{k-1}^S$ is defined as the intersection between all the particles in time $k-1$.

$$P_{k-1}^S = \bigcap_{i=1}^{N_s} P_{k-1}^{(i)} \tag{3.14}$$

where $N_s$ is the number of particles.

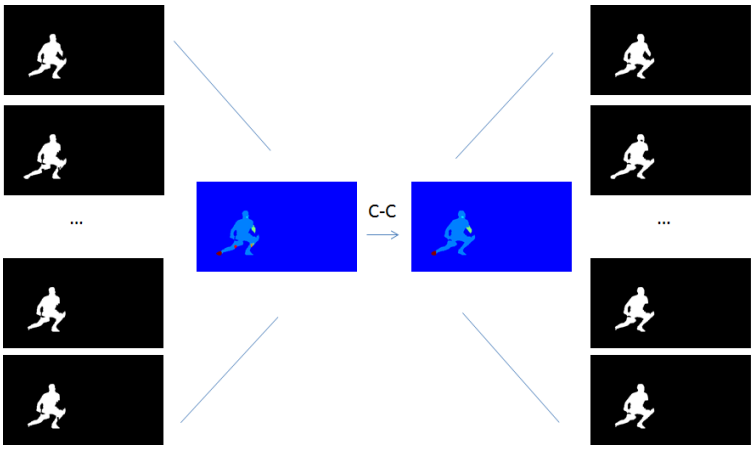


FIGURE 3.6: Particle support partition generated to do a propagation in one step. Some internal object regions fused during the co-clustering step.

### 3.3.2 Changes in co-clustering

The changes in the co-clustering algorithm proposed in this thesis are mostly focused on reducing its computational time and to increase its robustness. Also, a change in the co-clustering method was introduced in [16], which proposed a multi-resolution hierarchical co-clustering having in account the similarities between regions of the same partition $P_{k-1}$ (Intra similarities) and the similarities between partitions of a pair of images $P_{k-1}$ and $P_k$ (Inter similarities).

The proposed changes on co-clustering in this thesis are 2: introduce *motion estimation* on the co-clustering process and *flow through the segmentation hierarchy* in order to simplify the propagation problem.

Co-clustering matches regions from $P_{k-1}$ and $P_k$ by means of contour comparison. The motion estimation is done via *Optical Flow* (OF), and serves as a rough propagation of the pixels of interest. These pixels are the pixels conforming the contours of the regions of the partition $P_{k-1}$. This helps the co-clustering algorithm as it gives a first estimation on where a contour pixel on $P_{k-1}$ may find its correspondence at $P_k$, hence allowing us to reduce the amount of surrounding pixels to look for the corresponding contour pixel between $P_{k-1}$ and $P_k$.

Also, another proposal on co-clustering is to exploit the hierarchical information provided by COB [12]. In figure 3.2 it can be seen that the amount of calculations

is much higher at lower hierarchical levels, as more contour elements are present in the partition. These contour elements are erased by means of adjacent region fusion as the threshold on the contour energy $\lambda$ increases. 3 attempts on co-clustering will be realized at different hierarchy levels, on $\lambda = \lambda(J, w)$, $\lambda = \lambda(J)$, and $\lambda = 0$. The reasoning behind this is robustness on the co-clustering step, because, sometimes, an object region may fuse to background in $P_k$ on a high hierarchy level (for instance on the cut $\lambda = \lambda(J, w)$), leading to a bad propagation. If that happens, a new propagation try on a lower hierarchy level ($\lambda = \lambda(J)$) will be computed. If the problem persists, the partition with the most dense region representation ( $\lambda = 0$) will be used in order to propagate the particles from $P_{k-1}$ to $P_k$.

Finally, after the propagation, the partition $P_k^\lambda$ that will be used in further steps is computed having in account the different particles and a *step factor* ($\Delta\lambda$) that will impede a fusion between 2 very different consecutive energy levels. The propagated particle support partition $P_k^S$ will be used to do the hierarchical flow with the restriction on not being able to fuse any of its regions. This way a global hierarchy cut, having in account all the particles and its diversity, is done.

## 3.4 Prediction and perturbation

In this section we present both prediction and perturbation of the particles. The first step *prediction* tries to ensure a minimum quality of the particles estimation. Then, in the second step *perturbation*, randomness is introduced to provide diversity among particles.

After the co-clustering step, the object estimation represented by each propagated particle is formed by a set of regions from $P_k$. Each particle represents a point in the solution space $\Omega$. These points are the initial prediction to our estimation problem, but not the final solution.

The particles propagation provide an initial solution to the estimation problem, but it requires additional steps to improve it. Those steps are the *prediction*, which, in a deterministic way, tries to provide a first improvement to the propagation, and the *perturbation*, which tries to ensure diversity between particles via randomness or noise.

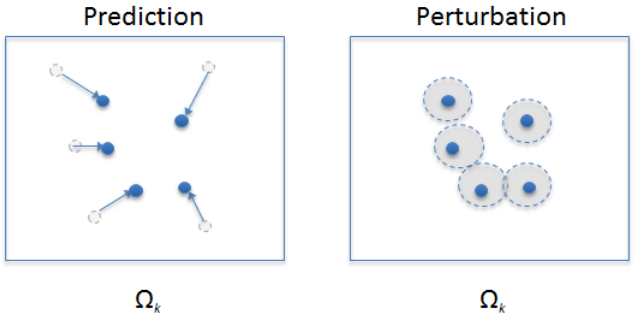


FIGURE 3.7: Each point on the space represents a particle. The left figure represents the prediction step, trying to guide the particles towards a good solution. The right figure represents the perturbation step, whose aim is to apply noise to the particles in order to provide diversity between them.

### 3.4.1 Bayesian estimation

Bayes theorem will be used to calculate both object and background probabilities. Such probability is stated as follows:

$$p(x_k \in \theta_k | \theta_{k-1}) = \frac{p(\theta_{k-1} | x_k \in \theta_k) \cdot p(x_k \in \theta_k)}{p(\theta_{k-1})} \tag{3.15}$$

where $x_k$ represents every pixel, $\theta_k$ the model (Object or Background), and $\theta_{k-1}$ is the previous time step distribution.

### 3.4.2 Prediction

The Prediction step is our proposal to do a first approach towards the final solution. Parting from the co-clustering result, some regions for every particle will be added/removed in a deterministic way. This reasoning comes from the fact that some regions, due to their color distribution, will be very likely to be part of the object, regarding the object model $O_{k-1}$, and some others to the background, regarding the background model $B_{k-1}$. These changes are proposed as we assume that they will increment the particle quality, leading it closer to a good solution.

Let us consider a particle $x_k^{(i)} = \bigcup_r^{n_k^o} R_k^r$, where $R_k^r$ is a region from the partition $P_k$. Then, considering both the object model $O_{k-1}$ and the background model $B_{k-1}$, the object and background probability of a region using Bayes Theorem in equation 3.15 will be:

$$p(R_k^r \in O_k | O_{k-1}) = \frac{p(O_{k-1} | R_k^r \in O_k) \cdot p(R_k^r \in O_k)}{p(O_{k-1})} \tag{3.16}$$

$$p(R_k^r \in B_k | B_{k-1}) = \frac{p(B_{k-1} | R_k^r \in B_k) \cdot p(R_k^r \in B_k)}{p(B_{k-1})} \tag{3.17}$$

For the object probabilities in equation 3.16: $p(R_k^r \in O_k)$ is the prior probability of the object, computed via *Optical Flow* and convolving the result with a Gaussian Kernel to provide a smoother prior with an upper-bound of 0.8 and a lower-bound of 0.2, $p(O_{k-1})$ is the prior probability of any pixel to be into the object, and $p(O_{k-1} | R_k^r \in O_k)$ is the probability of the object model obtained at the previous time instant assuming that the region $R_k^r$ under analysis belongs to the object. This last equation is computed as the average likelihood of the model considering all the region pixels:

$$p(O_{k-1} | R_k^r \in O_k) = \frac{1}{N_k^r} \sum_{i,j \in R_k^r} p(O_{k-1} | I_k(i,j) \in O_k) \tag{3.18}$$

where $N_k^r$ is the number of region pixels and $I_k$ is the image (frame) under analysis. The same statements above are applied to obtain the background probabilities.

Once the probability of belonging to object/background is computed for every region of the partition, a direct comparison for every propagated particle is done.

FIGURE 3.8: Left figure represents the probability to tag the regions as foreground (object) and the right one represents the probability of a region to be background. The brighter the value, the higher the probability of belonging to one or the other.

This way, the regions with very high/low probability of existing in the model object/background will be taken in account to be added/removed to each particle. To infer the relation between object and background probabilities for each region, Bayes factor $B$ is used [17]. It is computed as:

$$B_k^r = 2 \cdot \ln(\frac{p(R_k^r \in O_k|O_{k-1})}{p(R_k^r \in B_k|B_{k-1})}) \tag{3.19}$$

where $B_k^r$ is the Bayes factor of the region $r$ from partition $P_k$.

For each particle, all its nearby regions with a large $B$ are included into the particle. On the contrary, regions compounding the particle with very negative $B$ will be considered as background and removed from the particle. Also the particle weight variation $\Delta w_k^{(i)}$ will be taken in account for every particle change proposal. This protects the particle to several changes that fulfill the condition on resembling much more to a model (object/background) than to the other, but makes the overall weight decrease. This is due to the fact that, when doing the comparison between a model and a region in our Bayesian estimation approach, only that region is taken in account, making not possible to get more information about the overall particle distribution. This means that, for some regions, the probability of belonging to a model (object/background) may be high due to the region's color distribution, but that change (adding/removing the region) might be bad for the overall particle, because that amount of color inside the color histogram representing the model is already filled by another set of regions.

### 3.4.3 Perturbation

Next step is the perturbation of the updated particles. This step introduces diversity between particles in order to create multiple hypotheses, leading to a good estimation of the object when combined. Statistically, by means of random changes to the particles, a good solution to the estimation problem should be achieved.

For the RBPF this would suppose to work at the lowest partition hierarchy level (an erroneous region can be fatal in a higher cut on the hierarchy) and to use a huge amount of particles. Trying to keep a guidance on the algorithm in order to reduce the sample space $\Omega$ for providing better results and a reduction on the computation cost, in this work we look for random changes to a particle that make its weight

increase, assuming that this will lead to a not-so-diverse solution but to a good estimation of the state.

Each particle $x_k^{(i)}$ is perturbed as follows. First, a distance between the particle and each region of the partition is estimated. This distance is the Euclidean distance between the centroid of each region inside a particle and the surrounding regions (those regions which are not inside the particle). Regions which are closer than $D$ pixels will be considered as candidates to be added to the particle. This decision is made to reduce the amount of regions which are unlikely to be part of the final estimation (as they are far away from the main blob), and thus reduce undesired noise and computational time.

Applying both equations 3.16 and 3.17 with a prior $p(R_k^r \in O_k) = \frac{1}{2}$, a probability for each region to correspond to the object or to the background is calculated. Then, some regions are randomly chosen to be added or removed for each particle $x_k^{(i)}$, based on the object/background probabilities. Every selected region will be accepted or discarded with probability $1/2$.

FIGURE 3.9: Left: example on prediction result. The regions marked in red are the ones to remove deterministically for one particle; if some region was about to be added it would appear in green. Right: example on perturbation result. Marked regions are the proposed changes to one particle. Purple represent regions to remove and blue are the ones to add.

## 3.5 Evaluation

Each particle $x_k^{(i)}$ will have an associated weight $w_k^{(i)}$. The particle weight is computed as a distance measurement between distributions. In this work, we consider the Bhattacharya coefficient as the measure between distributions associated with the propagated mask of each particle and the model distribution. Distributions of particles and models are generated by normalized histograms of the corresponding RGB mask associated to the particle or the model.

Let $h(x_k^{(i)})$ be the histogram of those pixels belonging to the mask associated with the particle $x_k^{(i)}$, and $q_{k-1}$ be the object model $O_{k-1}$ histogram. Then, the Bhattacharya coefficient of the particle $i$ is computed as:

$$\rho_k^{(i)}(q_{k-1}, h(x_k^{(i)})) = \sum_{b=1}^{N_b} \sqrt{h^{(b)}(x_k^{(i)}) \cdot q_{k-1}^{(b)}} \tag{3.20}$$

where $N_b$ is the number of bins per channel. In this work, RGB space color has been used with $N_b = 20$ bins per channel.

By means of this distance, particles with color distribution resembling the model's distribution will have a lower Bhattacharya coefficient $\rho$ . Then, the weight $w_k^{(i)}$ of particle $x_k^{(i)}$ is denoted as:

$$w_k^{(i)} = e^{-\frac{\rho_k^{(i)}}{\sigma}} \tag{3.21}$$

where $\sigma$ is set to 0.4 in this work.

## 3.6 Resampling

The *Sequential Importance Sampling* (SIS) algorithm performs the recursive propagation of the particles and their associated weights as each measurement is received sequentially over time. A common problem with the SIS particle filter is that, after a few iterations, all but one particle will have negligible weight and one particle will concentrate all the probability mass. This is an undesired situation, as a single particle will not correctly represent the function $p(x_{1:k}|z_{1:k})$ in almost any case. This is called the *degeneracy problem* [3]. To avoid this, resampling of the particles is applied at every time step.

Given a set of $N_s$ particles $S_k = \{x_k^{(1)}, x_k^{(2)}, ..., x_k^{(N_s)}\}$, another set with the same number of particles $S_k^{'} = \{x_k^{'(1)}, x_k^{'(2)}, ..., x_k^{'(N_s)}\}$ is created. The new set of particles $S_k^{'}$ is created by randomly sampling the set $S_k$ with replacement on every particle. In this process, particles with high weights may be chosen several times in front of particles with lower weights.

## 3.7 Estimation

The estimation of the object is obtained as the average weighted state hypotheses, represented by the particles. Each particle has an associated mask $M_k^{(i)}$, which corresponds to the union of regions the particle contains. Then, the average mask $A_k^M$ is computed as:

$$A_k^M = \sum_{i=1}^{N_s} w_k^{(i)} \cdot M_k^{(i)} \tag{3.22}$$

where $N_s$ is the total number of particles and $w_k^{(i)}$ is the weight of the particle. Those pixel values that accomplish $A_k^M > T_o$ will be denoted as the final state estimation (or object estimation).


FIGURE 3.10: From left to right: particle stacking, represented by $A_k^M$, final state estimation $x_k$, represented by $A_k^M > T_o$, and overlay between the final estimated mask and the image $I_k$.

# 4 Results and Evaluation

## 4.1 DAVIS database

Our experiments are tested on the *DAVIS* (Densely Annotated VIdeo Segmentation) database [2], which consists on 50 video sequences spanning multiple occurrences of common video object segmentation challenges such as occlusions, motion-blur and appearance changes. Each video is accompanied by densely annotated, pixel-accurate and per-frame ground truth segmentation.

To compare our method with other state-of-the-art methods, the results are presented using the *Jaccard Index* between the annotation and the result generated by our method, as *DAVIS* provides the results on every method through this measure. We compare RBPF with other *semi-supervised* methods, which are the ones that require a human or machine interaction to provide the object mask in one or several frames. In our case, the first frame object segmentation is required.

## 4.2 Training

Several testing on *RBPF* has been made. As the method is time and computational resources consuming a subset of 5 sequences have been chosen in order to train the system. These sequences contain different attributes, such as non-linear deformation, edge ambiguities, motion blur or appearance changes among others. The following experiments were introduced by several questions: do we really need a particle filter approach? Does the randomness really improve the estimation, or could we just use the Bayesian approach in its deterministic form? Could it be that the co-clustering is good enough to estimate the object over the sequence with the new hierarchical flow we presented? Based on these questions we designed a set of experiments:

1. Co-clustering only: The initial mask is propagated through the sequence without any other change to the particles, only propagation via co-clustering.
2. Generic RBPF: Follows the initial thought on RBPF, which is propagating and, randomly, perturbating the particles.
3. Deterministic RBPF: Not considering the perturbation (random) step, only the propagation and the prediction (deterministic) step. This algorithm might not be considered a particle filter, but some tests on it were made as it is inspired on particle filters.
4. RBPF: The Region Based Particle Filter we propose with a combination between *propagation*, *prediction*, and *perturbation*.

The training consisted on taking a subset of sequences from DAVIS (*Dog, Rollerblade, Soapbox, Tennis, and Train*), and choosing the parameters by means of the obtained results.

In table 4.1 the results are presented as follows: *co-cl* is the mask propagation using solely the co-clustering technique all over the sequence, *rnd* refers to the generic RBPF, where is propagating and pertubating the particles, *det* is deterministically approach the object estimation, and *combo* is the combination between the deterministic and random techniques. As seen in this table, the combination between techniques is the one that produces the highest Jaccard. Further explanation on these results is:

1. Co-clustering only: By only propagating a mask over the sequence, the propagating errors (e.g. an undesired fusion between a pair of regions) will not be possibly corrected. This means that this error will be propagated all along the sequence. On the other hand, if several conditions are fulfilled, such as the object contours are well-defined all over the sequence, this may lead to a better result than the other approaches.

   The reason for this is the no-comparison between the object model and the mask to propagate. Co-clustering is based on comparing contours and trying to fit them on the next frame partition. If the object needs a faster model update due to abrupt changes on the object, the prediction and perturbation step might fail (they will try to find patterns of the previous model). On these situations, applying only the co-clustering step may lead to better results.

   The reason for this is that some sequences would need a faster model update. Most of the cases where co-clustering performs better than the other approaches are when the object begins being very small and increases its size abruptly, making it impossible for the model to be updated in time.

2. Generic RBPF: In order to properly test this approach, a span over several amount of particles may be needed. Due to computational limitations we could only test the whole database with a maximum amount of 50 particles, leading to a high variance in some sequences. Also, specific training would be needed in order to tune the parameters for this specific experiment.

   Furthermore, for this approach, working at a higher hierarchical cut may not lead to a good estimation in several cases. The reason for this is that the errors produced on a higher hierarchy level are much worse than the ones produced on a low hierarchy level.

   A solution to this is proposed on next chapter, where a multiple hierarchical levels segmentation is proposed as further investigation.

3. Deterministic RBPF: It proves that guiding particles towards a good estimation, leading to a reduction on the solution space $\Omega$, produces quite good results. Also, only a single particle is required, and even this method is not considered to be a particle filter, it is used to compare the RBPF to this approach.

   On the other hand, the lack of diversity among particles, provided by the randomness on a particle filter, might be a strong point against this approach. If a bad change is proposed by the deterministic step, there is no way that some of the particles change this decision. The results on this can be improved by means of randomness in order to introduce this diversity among particles.

4. RBPF: It can be seen that this approach has the best results overall the sequence for the 4 proposed tests. By means of combining the 3 other approaches the best results are obtained. We have led the algorithm to a good estimation while keeping diversity among particles.

Also tests on the amount of particles had been made. The final configuration to compare our method with other state-of-the-art methods in DAVIS was set to *50 particles* (as a tradeoff between results and time consumed), and combining *co-clustering*, *prediction*, and *perturbation*.

## 4.3 Evaluation

The evaluation of the whole database has been made choosing the best configuration for the training sequences. This configuration is the combination between deterministic RBPF and generic RBPF. We compare our method (RBPF) with the *semi-supervised* state-of-the-art methods benchmark in the DAVIS web page. These methods are TSP [9], SEA [6], HVS [4], JMP [7], FCP [5] and BVS [8].

### 4.3.1 Qualitative results

Some qualitative results from the DAVIS database are presented.

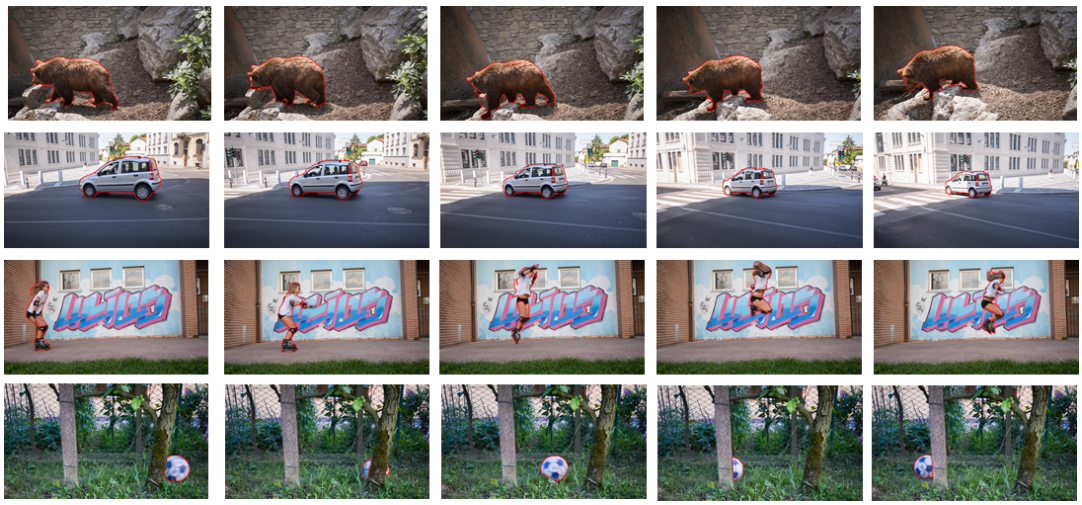

FIGURE 4.1: Some qualitative results on the dataset. The sequences names are *Bear*, *Car-shadow*, *Rollerblade*, and *Soccerball*.

### 4.3.2 Evaluation

The average evaluation between the *semi-supervised* methods presented on DAVIS is showed in table 4.2. We can state than, on average, RBPF performs better than all the other *semi-supervised* methods presented in DAVIS. On table 4.3 detailed evaluation on *Jaccard index* for every sequence is provided.

TABLE 4.1: Training: Testing between the 4 proposed techniques on DAVIS. Jaccard index, higher is better.

| *Sequence* | *co-cl* | *rnd* | *det* | *combo* |
|---|---|---|---|---|
| Bear | 0.846 | 0.850 | 0.892 | **0.923** |
| Blackswan | 0.914 | **0.919** | 0.911 | 0.910 |
| Bmx-Bumps | 0.140 | 0.132 | 0.312 | **0.314** |
| Bmx-Trees | 0.080 | 0.239 | 0.252 | **0.308** |
| Boat | 0.021 | **0.659** | 0.573 | 0.530 |
| Breakdance | 0.020 | 0.198 | **0.527** | 0.407 |
| Breakdance-Flare | 0.173 | 0.661 | 0.735 | **0.762** |
| Bus | 0.553 | 0.673 | 0.683 | **0.689** |
| Camel | 0.586 | 0.590 | **0.611** | 0.609 |
| Car-Roundabout | 0.752 | 0.810 | **0.922** | 0.916 |
| Car-Shadow | 0.891 | 0.937 | **0.952** | 0.951 |
| Car-Turn | 0.937 | 0.934 | **0.949** | 0.949 |
| Cows | 0.767 | 0.862 | 0.891 | **0.893** |
| Dance-Jump | 0.391 | 0.418 | **0.684** | 0.584 |
| Dance-Twirl | 0.141 | 0.354 | **0.357** | 0.353 |
| Dog | 0.327 | 0.370 | **0.757** | 0.684 |
| Dog-Agility | 0.288 | 0.290 | 0.346 | **0.542** |
| Drift-Chicane | **0.781** | 0.340 | 0.580 | 0.489 |
| Drift-Straight | 0.510 | 0.486 | 0.566 | **0.568** |
| Drift-Turn | 0.898 | **0.900** | 0.895 | 0.895 |
| Elephant | 0.326 | 0.334 | 0.806 | **0.806** |
| Flamingo | 0.734 | **0.820** | 0.807 | 0.807 |
| Goat | 0.720 | 0.769 | **0.780** | 0.780 |
| Hike | 0.668 | **0.891** | 0.890 | 0.890 |
| Hockey | 0.121 | 0.276 | **0.585** | **0.585** |
| Horsejump-High | 0.487 | 0.476 | **0.660** | 0.634 |
| Horsejump-Low | 0.413 | 0.487 | **0.592** | 0.581 |
| Kite-Surf | 0.442 | **0.618** | 0.550 | 0.589 |
| Kite-Walk | 0.695 | 0.703 | 0.747 | **0.751** |
| Libby | 0.250 | 0.454 | 0.447 | **0.615** |
| Lucia | 0.817 | **0.847** | 0.825 | 0.800 |
| Mallard-Fly | 0.558 | 0.579 | 0.586 | **0.587** |
| Mallard-Water | 0.561 | 0.843 | **0.868** | 0.864 |
| Motocross-Bumps | **0.629** | 0.286 | 0.303 | 0.241 |
| Motocross-Jump | **0.505** | 0.340 | 0.347 | 0.318 |
| Motorbike | 0.232 | **0.603** | 0.476 | 0.573 |
| Paragliding | 0.857 | 0.859 | **0.864** | 0.863 |
| Paragliding-Launch | 0.588 | 0.597 | 0.590 | **0.609** |
| Parkour | 0.143 | 0.267 | **0.589** | 0.508 |
| Rhino | 0.625 | 0.676 | **0.681** | 0.680 |
| Rollerblade | 0.194 | 0.553 | 0.628 | **0.848** |
| Scooter-Black | 0.571 | 0.591 | 0.628 | **0.683** |
| Scooter-Gray | 0.329 | 0.320 | 0.198 | **0.374** |
| Soapbox | 0.573 | 0.417 | 0.650 | **0.820** |
| Soccerball | 0.092 | **0.908** | 0.886 | 0.907 |
| Stroller | 0.258 | 0.806 | 0.775 | **0.807** |
| Surf | **0.897** | 0.888 | 0.895 | 0.893 |
| Swing | 0.112 | 0.404 | 0.586 | **0.679** |
| Tennis | 0.267 | 0.635 | **0.820** | 0.777 |
| Train | 0.023 | 0.341 | 0.641 | **0.810** |
| MEAN | 0.474 | 0.584 | 0.662 | **0.679** |

| | | TSP | SEA | HVS | JMP | FCP | BVS | RBPF |
|---|---|---|---|---|---|---|---|---|
| $\mathcal{J}$ | Mean $\mathcal{M}\uparrow$ | 0.358 | 0.556 | 0.596 | 0.607 | 0.631 | 0.665 | **0.679** |
| | Recall $\mathcal{O}\uparrow$ | 0.388 | 0.606 | 0.698 | 0.693 | **0.778** | 0.764 | 0.760 |
| | Decay $\mathcal{D}\downarrow$ | 0.385 | 0.355 | 0.197 | 0.372 | **0.031** | 0.260 | 0.191 |
| $\mathcal{F}$ | Mean $\mathcal{M}\uparrow$ | 0.346 | 0.533 | 0.576 | 0.586 | 0.546 | 0.656 | **0.658** |
| | Recall $\mathcal{O}\uparrow$ | 0.329 | 0.559 | 0.712 | 0.656 | 0.604 | **0.774** | 0.723 |
| | Decay $\mathcal{D}\downarrow$ | 0.388 | 0.339 | 0.202 | 0.373 | **0.039** | 0.236 | 0.183 |

TABLE 4.2: Overall results of region similarity ($\mathcal{J}$) and contour accuracy ($\mathcal{F}$) for each of the semi-supervised methods. The results show that, on average, our RBPF approach preforms better than the other semi-supervised state-of-the-art methods presented on DAVIS.

TABLE 4.3: Detailed comparison between *semi-supervised* methods on DAVIS. Jaccard index, higher is better.

| *Sequence* | *OURS* | *BVS* | *FCP* | *JMP* | *HVS* | *SEA* | *TSP* |
|---|---|---|---|---|---|---|---|
| Bear | 0.923 | **0.955** | 0.906 | 0.929 | 0.938 | 0.912 | 0.778 |
| Blackswan | 0.910 | **0.943** | 0.908 | 0.930 | 0.916 | 0.933 | 0.872 |
| Bmx-Bumps | 0.314 | **0.434** | 0.300 | 0.336 | 0.428 | 0.198 | 0.290 |
| Bmx-Trees | 0.308 | **0.382** | 0.248 | 0.229 | 0.179 | 0.113 | 0.095 |
| Boat | 0.530 | 0.644 | 0.613 | 0.705 | 0.782 | **0.793** | 0.656 |
| Breakdance | 0.407 | 0.500 | **0.567** | 0.478 | 0.550 | 0.329 | 0.056 |
| Breakdance-Flare | **0.762** | 0.727 | 0.723 | 0.430 | 0.499 | 0.131 | 0.040 |
| Bus | 0.689 | **0.863** | 0.832 | 0.668 | 0.809 | 0.752 | 0.515 |
| Camel | 0.609 | 0.669 | 0.734 | 0.640 | **0.876** | 0.649 | 0.654 |
| Car-Roundabout | **0.916** | 0.851 | 0.717 | 0.726 | 0.777 | 0.708 | 0.614 |
| Car-Shadow | **0.951** | 0.578 | 0.723 | 0.645 | 0.699 | 0.775 | 0.636 |
| Car-Turn | **0.949** | 0.844 | 0.724 | 0.834 | 0.810 | 0.909 | 0.323 |
| Cows | 0.893 | **0.895** | 0.812 | 0.756 | 0.779 | 0.707 | 0.595 |
| Dance-Jump | 0.584 | **0.745** | 0.522 | 0.490 | 0.680 | 0.662 | 0.132 |
| Dance-Twirl | 0.353 | **0.492** | 0.471 | 0.444 | 0.318 | 0.117 | 0.099 |
| Dog | 0.684 | 0.723 | **0.774** | 0.673 | 0.722 | 0.581 | 0.313 |
| Dog-Agility | 0.542 | 0.345 | 0.453 | **0.699** | 0.457 | 0.354 | 0.079 |
| Drift-Chicane | **0.489** | 0.033 | 0.457 | 0.243 | 0.331 | 0.119 | 0.018 |
| Drift-Straight | 0.568 | 0.402 | **0.668** | 0.618 | 0.295 | 0.513 | 0.198 |
| Drift-Turn | **0.895** | 0.299 | 0.606 | 0.717 | 0.276 | 0.667 | 0.162 |
| Elephant | 0.806 | **0.850** | 0.655 | 0.750 | 0.742 | 0.553 | 0.666 |
| Flamingo | 0.807 | **0.881** | 0.717 | 0.530 | 0.811 | 0.583 | 0.666 |
| Goat | **0.780** | 0.661 | 0.677 | 0.731 | 0.580 | 0.535 | 0.444 |
| Hike | **0.890** | 0.755 | 0.874 | 0.664 | 0.877 | 0.776 | 0.679 |
| Hockey | 0.585 | **0.829** | 0.647 | 0.677 | 0.698 | 0.714 | 0.413 |
| Horsejump-High | 0.634 | **0.801** | 0.676 | 0.586 | 0.765 | 0.638 | 0.236 |
| Horsejump-Low | 0.581 | 0.601 | 0.607 | **0.663** | 0.551 | 0.498 | 0.291 |
| Kite-Surf | **0.589** | 0.425 | 0.577 | 0.500 | 0.405 | 0.486 | 0.366 |
| Kite-Walk | 0.751 | **0.870** | 0.682 | 0.509 | 0.765 | 0.498 | 0.447 |
| Libby | 0.615 | **0.776** | 0.316 | 0.295 | 0.553 | 0.226 | 0.070 |
| Lucia | 0.800 | **0.901** | 0.801 | 0.836 | 0.776 | 0.626 | 0.377 |
| Mallard-Fly | 0.587 | **0.606** | 0.541 | 0.536 | 0.436 | 0.557 | 0.200 |
| Mallard-Water | 0.864 | **0.907** | 0.687 | 0.751 | 0.704 | 0.865 | 0.623 |
| Motocross-Bumps | 0.241 | 0.401 | 0.306 | **0.761** | 0.534 | 0.470 | 0.133 |
| Motocross-Jump | 0.318 | 0.341 | **0.511** | 0.583 | 0.099 | 0.387 | 0.123 |
| Motorbike | 0.573 | 0.563 | **0.713** | 0.506 | 0.687 | 0.451 | 0.340 |
| Paragliding | 0.863 | 0.875 | 0.866 | **0.951** | 0.907 | 0.863 | 0.735 |
| Paragliding-Launch | 0.609 | **0.640** | 0.571 | 0.589 | 0.537 | 0.577 | 0.301 |
| Parkour | 0.508 | **0.756** | 0.322 | 0.342 | 0.240 | 0.121 | 0.070 |
| Rhino | 0.680 | 0.782 | 0.794 | 0.716 | **0.812** | 0.736 | 0.694 |
| Rollerblade | **0.848** | 0.588 | 0.450 | 0.726 | 0.461 | 0.138 | 0.098 |
| Scooter-Black | 0.683 | 0.337 | 0.504 | 0.626 | 0.624 | **0.793** | 0.378 |
| Scooter-Gray | 0.374 | **0.508** | 0.483 | 0.123 | 0.433 | 0.241 | 0.133 |
| Soapbox | **0.820** | 0.789 | 0.449 | 0.758 | 0.684 | 0.783 | 0.247 |
| Soccerball | **0.907** | 0.844 | 0.820 | 0.097 | 0.065 | 0.653 | 0.029 |
| Stroller | **0.807** | 0.767 | 0.597 | 0.656 | 0.662 | 0.464 | 0.369 |
| Surf | 0.893 | 0.492 | 0.843 | **0.941** | 0.759 | 0.821 | 0.814 |
| Swing | 0.679 | **0.784** | 0.648 | 0.115 | 0.104 | 0.511 | 0.098 |
| Tennis | **0.777** | 0.737 | 0.623 | 0.765 | 0.576 | 0.481 | 0.074 |
| Train | 0.810 | 0.872 | 0.841 | **0.873** | 0.846 | 0.854 | 0.648 |
| MEAN | **0.679** | 0.665 | 0.631 | 0.607 | 0.596 | 0.556 | 0.358 |

# 5 Conclusions and next steps

*Region Based Particle Filter* has proven to be competitive against state-of-the-art methods. Also, we have proven that, by guiding the algorithm towards a good solution (or what we assume is a good solution), better results are generated. On most cases this guidance is translated into a reduction on the solution space of possibilities $\Omega$.

This algorithm, however, does not run in real time. This means that its applications would be set as post-processing tools. Some applications for this technique would be in *cinema* or *advertisements*, e.g. to focus a region of interest, modify the light or color and propagate this change over the video sequence, or from *2D to 3D layer conversion*, e.g. track every layer in 2D and map it to the 3D space in a video sequence.

Some possible next steps to take in consideration (or considered during the thesis) are:

**Multiple hierarchal cuts for partition representation**: In [16] the concept of multiple cuts over the same partition hierarchy appears. A very nice approach to provide a tradeoff between complexity and a good estimation in a RBPF would be having *multiple resolution* in the same partition. An example to this would be a partition where most of the interior of the main object is fused, while having a crown of more dense superpixels over its outlying contours, reducing the solution space $\Omega$ while providing a better possible estimation. This technique would offer a blend cut (multiple $\lambda$) over the partition tree in contrast to the method used in this thesis, which is a rigid cut (a single $\lambda$) over a hierarchical tree to generate the partition.

**Region mapping on partition hierarchies**: Mapping between 2 partitions is done by means of co-clustering. It would be very interesting to construct a hierarchy tree containing the region and contour information. That way, both partitions could be related by means of directly clustering the nodes on their hierarchy trees. Furthermore, if the previous step to take in consideration *Multiple hierarchal cuts for partition representation* is applied in combination with this proposal, a fast and accurate partition relation method could be generated.

**Temporal contour consistency**: Some problems on the co-clustering come from the energy inconsistency of some contours. Each segmentation for every frame produced by COB [12] is done independently from each other. If a temporal window on these partitions was introduced, maybe the temporal inconsistency would disappear. On our side, we could detect such contours and modify their energy on the hierarchical tree.

**Model update depending on sequence characteristics**: As it can be inferred from the *co-clustering only* approach conclusions in the training section 4.2, the model update might be dependent on the type of sequence the algorithm is facing, as some sequences may need a high-paced model update, while others may need a slower

one. A proposed next step on this would be finding a metric to determine if the forgetting factor $\alpha$ in equation 3.10 for the model update should be high or low, depending on the sequence characteristics.

**CNN descriptors**: RBPF is based on color descriptors. This is due their invariance in rotation or scaling and that color histograms can be directly mapped onto a pdf. As particle filters try to estimate the state pdf it is a straightforward solution. However, color features are useful in some scenarios, where the background and foreground are quite different or the object model does not change abruptly, but on many others fail. A considered next step is to change to *Convolutional Neural Nets* (CNN) features, as the information that can be extracted from them is much richer than the solely color information. Some tests were made using *GoogLeNet* [18]. They were unsuccessful, but gave us an idea where to aim if we wanted to use such descriptors.

**Personal evaluation**: Nowadays we can see a boom in Deep Learning for every application. In the next CVPR (2017) several new tracking algorithms will be presented, outperforming, I'm sure, the current state-of-the-art. A discussion on Neural Nets arises. Those statistical methods are as blind as a Particle Filter would be, but are able to handle that big solution space by means of huge amount of data and training. Will those methods need in a near future some intelligent guidance as we provided to the RBPF or will they overcome in this Big Data world the future presents us?

# Bibliography


[1] D. Varas and F. Marques, "Region-based particle filter for video object segmentation", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3470–3477.

[2] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation", in *Computer Vision and Pattern Recognition*, 2016.

[3] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking", *IEEE Transactions on signal processing*, vol. 50, no. 2, pp. 174–188, 2002.

[4] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph based video segmentation", *IEEE CVPR*, 2010.

[5] F. Perazzi, O. Wang, M. Gross, and A. Sorkine-Hornung, "Fully connected object proposals for video segmentation", in *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[6] S Avinash Ramakanth and R Venkatesh Babu, "Seamseg: Video object segmentation using patch seams", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 376–383.

[7] Q. Fan, F. Zhong, D. Lischinski, D. Cohen-Or, and B. Chen, "Jumpcut: Non-successive mask transfer and interpolation for video cutout", *ACM Transactions on Graphics (Proceedings of SIGGRAPH ASIA 2015)*, vol. 34, 2015.

[8] N. Maerki, F. Perazzi, O. Wang, and A. Sorkine-Hornung, "Bilateral space video segmentation", in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] J. Chang, D. Wei, and J. W. F. III, "A Video Representation Using Temporal Superpixels", in *IEEE Computer Vision and Pattern Recognition Conference on Computer Vision*, 2013. [Online]. Available: http://people.csail.mit.edu/jchang7/pubs/publications/chang13_CVPR.pdf.

[10] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering", *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[11] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation", in *IEE Proceedings F (Radar and Signal Processing)*, IET, vol. 140, 1993, pp. 107–113.

[12] K. Maninis, J. Pont-Tuset, P. Arbeláez, and L. V. Gool, "Convolutional oriented boundaries", in *European Conference on Computer Vision (ECCV)*, 2016.

[13] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation", *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[14] D. Glasner, S. Vitaladevuni, and R. Basri, "Contour-Based Joint Clustering of Multiple Segmentations", 2011.

[15] A. Girbau Xalabarder, "Region-based particle filter", B.S. thesis, Universitat Politècnica de Catalunya, 2015.

[16] D. Varas, M. Alfaro, and F. Marques, "Multiresolution hierarchy co-clustering for semantic segmentation in sequences with small variations", in *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[17] R. E. Kass and A. E. Raftery, "Bayes factors", *Journal of the american statistical association*, vol. 90, no. 430, pp. 773–795, 1995.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.