Hierarchical Visual Content Modelling and Query based on Trees

Arief Setyanto

A thesis submitted for the degree of

Doctor of Philosophy

at the

School of Computer Science and Electronic Engineering

University of Essex

May 2016

Abstract

In recent years, such vast archives of video information have become available that human annotation of content is no longer feasible; automation of video content analysis is therefore highly desirable. The recognition of semantic content in images is a problem that relies on prior knowledge and learnt information and that, to date, has only been partially solved. Salient analysis, on the other hand, is statistically based and highlights regions that are distinct from their surroundings, while also being scalable and repeatable. The arrangement of salient information into hierarchical tree structures in the spatial and temporal domains forms an important step to bridge the semantic salient gap. Salient regions are identified using region analysis, rank ordered and documented in a tree for further analysis. A structure of this kind contains all the information in the original video and forms an intermediary between video processing and video understanding, transforming video analysis to a syntactic database analysis problem.

This contribution demonstrates the formulation of spatio-temporal salient trees the syntax to index them, and provides an interface for higher level cognition in machine vision.

Acknowledgements

This thesis would not have been possible without the support and valuable contributions of many people throughout my studies. First of all, I would like to express my gratitude to my supervisors, Dr John C Woods and Profesor Mohammed Ghanbari for their valuable suggestions, critical comments, and support in dealing with the intricacies of my research. I am grateful for their help from the inception of this work to this final stage. My thanks also go to Professor Steve Sangwine and Dr. Adrian Clark for discussion and comments on my work as board members.

I am also thankful to my Indonesian friends in Colchester including Hakimul Ikhwan, Lia, Zeynita Gibbons, Atu Rosalina, Agung Mandala, Wati Halesworth, Amika Wardana, Maharani Yusro, Monika, Miftahuddin, Yusro bin Samiran, Idham Ananta, Vogy, Rudi Kusdiantara, Yuslenita, Rendra, Ajeng, Susi, Dr. Hadi Susanto, Dr. Murniati and all others. I also thank my colleagues Felix, Femi, Ali, Khabir, Juliet, Zardad, Sulaiman and many others at the University of Essex for their support and solidarity.

I owe a huge debt to the Indonesian Directorate General of Higher Education for the scholarship during my study at Essex, as well as to the STMIK AMIKOM Yogyakarta, Indonesia for financial and professional support.

Finally, my greatest thanks are to my family, including my extended family. I owe a huge debt to my father Arief Sukirto, the greatest motivator I ever had, and my mother, Daryati, the true teacher of my life. My gratitude is also due to my mother and father-in-law, Rusmiyatun and Mujahid, as well as my sisters and brothers, Ari Setyani, Widi Astuti , Ami Supriani and Febiyanto. Finally, my very special and utmost thanks to my wife Sri Utami Zuliana, my Sons Faiq Muhammad Arif and Lukito Lukman Arif, and my daughter Aisyah Arfiana for their love and patient support during my studies. To them I devote this thesis.

Contents

List of Figures

List of Tables

Abreviations

AV audio visual.

BPT Binary Partition Tree.

GT Ground Truth.

MPEG Moving Picture Experts Group.

MS Mean Shift Algorithm.

RAG Region Adjacency Graph.

RGB Red Green Blue.

SLIC Simple Linear Iterative Clustering.

SQL Structured Query Language.

STRAG Spatio Temporal Region Adjacency Graph.

VAG Volume Adjacency Graph.

WS Watershed Algorithm.

Symbol

$C(M)$ catchment basin with minimum M.

$F(x, y)$ vector function at point x, y.

$F(x, y, t)$ vector function at point (x, y, t).

$Hist^{u}_{R_i}$ The histogram value of bin (level) u in the Region R_i.

Ig grascale image.

Img Image.

R Region/superpixel a group of neighbouring pixels in image/single frame. It is a vector with three mean colour components, size, and centroid coordinates.

V Volume/supervoxel a group of neighbouring voxels in subsequent frames of a video. It contains a vector of three colour components, size centroid of the first and last frame.

Vid Video.

$\delta_h(R_i, R_j)$ Quantified Colour Histogram Distance between Region i and j.

$\delta_s(R_i^t, R_j^{t+1})$ Inter frame region size distance of region i in current frame (t) and region

j in the upcoming frame $(t+1)$.

$\delta_v(V_i, V_j)$ distance between supervoxel/volume i and j.

$\delta_{\vec{ct}}(R_i^t, R_j^{t+1})$ Inter frame centroid distance of region i in current frame (t) and region j in the upcoming frame $(t+1)$.

$\delta_{a\vec{c}}(R_i, R_j)$ Absolute Colour Distance between Region i and j.

$\delta_{as\vec{c}}(R_i, R_j)$ Size proportional Absolute colour distance between Region i and j..

$\delta_{cts}(R_i^t, R_j^{t+1})$ Inter frame region distance, considering colour, centroid and size of region i in current frame (t) and region j in the upcoming frame $(t+1)$.

$\delta_{e\vec{c}}(R_i, R_j)$ Euclidean Colour Distance between Region i and j.

$\delta_{es\vec{c}}(R_i, R_j)$ Size proportional Euclidean colour distance between Region i and j..

$\gamma(k)$ evolution function of node at k level on certain path in the BPT.

$\mathcal{E}$ Edge in the Graph.

$\mathcal{G}$ graph.

$\mathcal{V}$ Vertex in the Graph.

φ Fitness Value.

$\vec{c}$ colour vector, consist of colour components.

$\|\|_2$ L_2-norm or Euclidean distance.

a size of a partition (superpixel/supervoxel).

$dir(V_i)$ the direction of centroid movement.

$f(x, y)$ scalar function at point (x, y).

$g(x, y)$ thresholding result at point (x, y).

t temporal axis/frame number.

$vel(V_i)$ the movement speed of the centroid.

x horizontal axis.

y vertical axis.

Chapter 1

Introduction

1.1 Motivation

The development of electronic imaging devices that are now embedded in most hand-held devices has led to an explosion in the amount of video and image data in the last couple of years. The visual data are archived either in personal or online digital storage. For example, the well-known video sharing website YouTube (www.youtube.com) reports that 300 hours of video are uploaded every minute and that the site has more than 1 billion users all over the globe. Even more video is produced individually by the new habit of recording pictures and videos in everyday life as a private collection.

Currently, video content description is usually produced by human effort with a small proportion such as the time stamp, location (geographic-tagging) and device being produced automatically by the capturing devices. Due to the fast growth in video data, human annotation is no longer feasible, however, many people leave their video collection without any description at all. This lack of annotation will make searching for information increasingly difficult. Automated video description is therefore required in order to minimize the user task.

Fully automated description is difficult to achieve due to machine limitations. Human visual systems have an effective mechanism to filter and understand the scene. In this regard, saliency detection has been extensively explored as a way to filter the important information and suppress the rest. Understanding the scene is a cognitive process, and prior knowledge plays a significant role in this task. On the other hand, machine records colour every point (pixel) on the visual space in a certain density. The other information such as texture, edge and motion is derived from pictorial information. Empowering the computer with an ability to mimic the human tasks of understanding video and providing a description automatically has become a challenge in video research in the past decade. A general-purpose solution to describe video content remains unavailable.

There are a number of domain-specific solutions such as pedestrian, sport, cooking, news and learning video where the sets of semantic objects and activities are limited. Domain specific solutions are usually implemented for particular purpose analysis when there is prior knowledge of the target objects. In circumstances where the video inputs vary widely across domains, limitation of the target objects cannot be implemented. A possible option for dealing with that issue provides an intermediate-level description of the scene that can bridge the human-machine gap. The description is not directly meaningful for the human subject; however, a semantic description can be derived from it.

A video signal is generated as an output of a camera, by scanning in two-dimensions of the scene over time. A moving scene is a collection of individual images (frames), in time series with a particular number of frames for every single second [1]. The key characteristic of video is associated with spatial and temporal information that delivers a semantically coherent narrative. Temporally consecutive frames have explicit spatial constraints with region inheritance, spatial correlation and motion information. A spatial and temporal coherency represents a regional evolution during a video sequence. Although every single frame delivers different sets of data, the content is redundant because most of the region in the current frame is inherited from the previous frame with gradual changes. In other words, some part of the frame stays consistent without any significant change, while another part of the frame experiences considerable changes. The changes can happen due to movement, rotation and region growth, or a combination of these.

In order to prepare an intermediate-level description of video signal, a hierarchical segmentation is considered to be the reliable way to represent video content. A tree structure allows the recording of the detail of the coarse abstraction of the information. Naturally, visual information contains multi-scale information. For example, a face image is supposed to record eyes and nose as part of the face as the part of a human body. The hierarchical structure offers the capability of recording in that way, even though in practical implementation not all the information stored at every level is meaningful for human. Metadata records the node features of the hierarchical tree.

1.2 Objective

The objective of this research is to provide intermediate level metadata of the video content which allows the user to search the visual content, and which keeps the metadata creation process automatically with minimum human intervention. In order to achieve that goal, through the course of this thesis, segmentation, hierarchical binary partition tree (BPT) creation, tree simplification, metadata creation and retrieval are demonstrated.

The problem is defined in section 1.3, and the solution overviews are discussed in section 1.4.

1.3 Problem Statement

Generating video metadata manually with human intervention is no longer feasible because of the huge amount of video data. There are a number of domain-specific automatic video descriptions that have been provided with a limited set of semantic objects. Limitations to the semantic objects cannot, however, be implemented in the circumstances where the video inputs vary widely across domains. Therefore, an intermediate-level description is a possible option to generate metadata automatically. Available unsupervised segmentation algorithms are fast, but they suffer from over-segmentation issues. In order to reduce the over-segmentation rate, a merging task needs to be performed. A merging rule and similarity measures need to be formulated to dictate the merging process. In a complete merging system, merging task will be carried out as long as a pair of partitions are available. The merging history can be recorded in a binary partition tree (BPT) structure. The BPT archives all partitions and their merging results. Therefore, the BPT becomes very complex and consists of thousands of nodes, where a number of salient nodes are formed during the evolution. The salient partitions are expected to be correlated to the meaningful object to human subject (semantically meaningful). The tree can be simplified by identifying the salient nodes and cutting the tree under these nodes. BPT structures that record the object candidates have to be translated into metadata. Attributes of the nodes need to be translated into human recognized terminology in order to provide intermediate level metadata. Finally, a mechanism has to be formulated in order to allow the users to express their requests to the visual content archived in the metadata.

1.4 Solution Overview

In order to provide metadata, a number of tasks need to be carried out, namely: segmentation, partitions merging, evolution analysis, tree simplification, metadata creation, and visual content retrieval. Segmentation is one of the early important processes in defining of visual content descriptions. According to [2] and [3], complete segmentation aims at dividing image/video into some semantically meaningful objects. On one hand, it usually needs prior knowledge to drive the segmentation process. On the other hand, partial segmentation aims at getting a number of partitions that satisfy a particular homogeneity criterion. These techniques are fast and do not need prior knowledge (un-

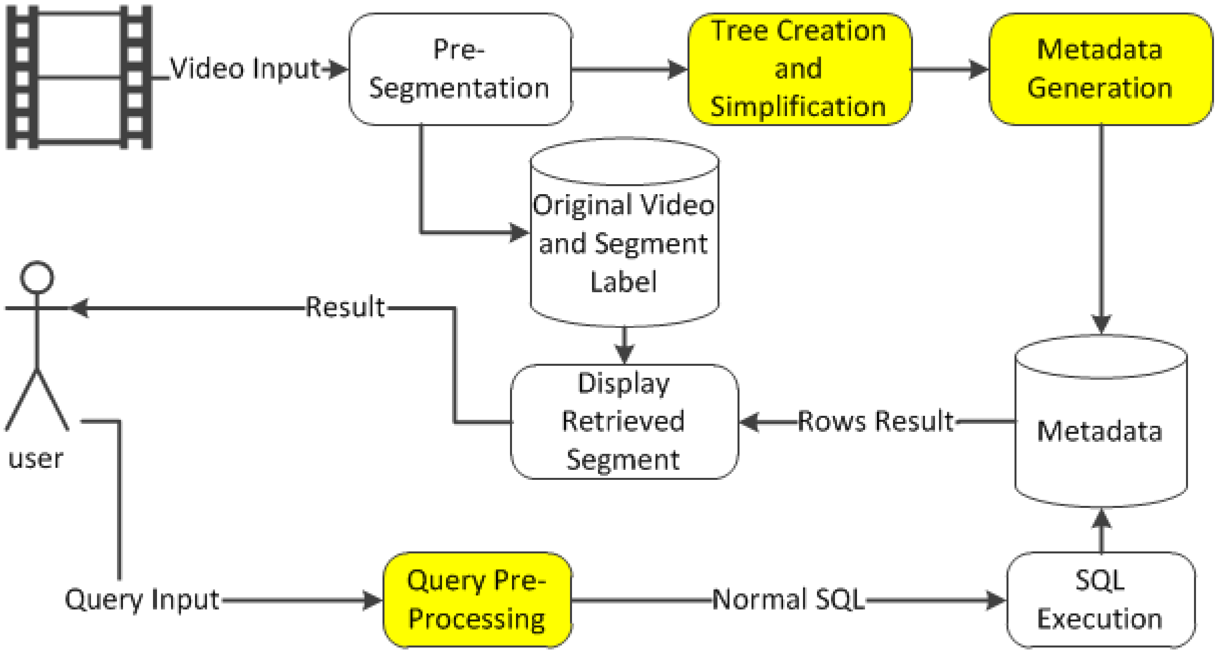


Figure 1.1: Building block of the work (Yellow blocks indicates the contribution of this thesis)

supervised), but the results are usually over-segmented. So, post-processing is needed.

Partial segmentation, hereinafter refers to as pre-segmentation, produces non overlapping partitions. In a single frame/image, it prepares as regions/'superpixel' and volume/'supervoxel' in a multi-frame video. Initial segmentations are produced by Watershed, mean shift and SLIC.

The merging task is carried out to get greater partitions and are expected to be more meaningful. A merging rule is responsible for selecting the merging pairs of neighbouring partitions. Similarity is the inverse of a distance measure formulation that determines the merging order. Formulations of absolute, Euclidean and histogram distance measures are evaluated against the segment quality and computational speed. The merging task is recorded in a binary partition tree (BPT) structure adopted from [4] and extended in the spatio temporal by [5]. BPT is adopted to record both the superpixel merging history of a single frame in the spatial approach and the supervoxel merging history in the volumetric approach. On one hand, the spatial approach puts a video in a number of frames and segments individually. On the other hand, the volumetric approach puts a stack of frames in three-dimensional matrices, and segmentation is carried out directly to these matrices.

The evolution of each initial small partition towards the global partition that contains the entire image/video is well documented in BPT structure. The evolution analysis is carried out over all branches of the tree to determine important nodes where the tree can be pruned to get a simpler BPT. The saliency rate of every node on the tree is determined by its distance to its parent and rank ordered to get a salient node list. It is arguable that salient partitions are formed during the evolution. The salient partitions are expected to be correlated to the meaningful object to human subject. If the tree is pruned in the salient nodes, it is expected that the final segmentation consists of salient partitions only, and a simpler tree with fewer nodes is obtained. A rule is applied to decide pruning nodes so that the simpler tree can be obtained without losing too much detailed information. The simplification is proposed at the multi levels based on how far it is the child-parent distance across the branches of the trees. Three levels of simplification are prepared according to the saliency rank on each branch of the tree. The three-level simplification offers flexibility and avoids losing detailed information by assuming a single answer for the partition set. The simplification result is evaluated against the available ground truths in [6] data set.

Pre-segmentation, merging and simplification operations yield some attributes to the nodes on the tree. All nodes on the tree correspond to particular partitions on the original video. The complete process above gives some attributes to every node that is recorded in the metadata. Some extra attributes are calculated in order to make the metadata closer to human requirements. The node and its neighbourhood relationship are recorded in the metadata. Some extended keywords are defined to work with the designed metadata in order to deal with a spatio-temporal information request. The building blocks of the work can be seen in Figure 1.1.

1.5 Contributions

- Provide metadata as an intermediate abstraction level in order to separate low level video processing and high level content analysis. The metadata on spatio-temporal binary partition trees, based on 3D SLIC, archives supervoxels and spatio-temporal neighbourhood properties.
- Provide unsupervised segmentations and tree simplification, thereby avoiding user intervention and the domain limitation for a video sequence.
- Comparison between three pre-segmentation algorithms for individual frames, namely mean shift, modified K-Means called SLIC, and Watershed in terms of providing initial segments for generating BPT. It is identified that SLIC gives the most stable performance and number of segments.
- Provide a mechanism to identify salient nodes in the tree, which is extended from [7]. Tree simplification is carried out by pruning the tree at the salient nodes. The top rank of salient nodes from the entire tree are identified and demonstrated to

be close to the available ground truth objects.

- Comparison between tree simplification techniques in terms of different initial segmentation and distance measures on a single frame BPT. It is identified that the most reliable output is produced by the SLIC with Euclidean distance measure, considering speed and final segmentation quality compared to the available ground truth.
- Comparison between two pre-segmentation algorithms for the entire video in 3D matrix representation, namely 3D SLIC and 3D Watershed in terms of providing initial supervoxel for generating spatio-temporal BPT and tree simplification. It is identified that 3D SLIC performs better than Watershed both in execution time and final partition quality compared to the available ground truth.
- Develop a query language extension to retrieve video content based on designed metadata. An evaluation of the retrieval operation is carried out on colour, motion, and the top-most saliency. It is identified that some colour-related queries get better answers from initial partition set. The motion-related query gets better results from the simplification set. The salient-related query shows that the the top-most salient partitions in the tree are strongly correlated to the objects in the ground truth on the video test.

1.6 Thesis Organization

This section briefly details the organization of the document. The remaining thesis is divided into seven chapters:

Chapter 2, presents a comprehensive literature review, the state of the art in segmentation algorithms in image and its extension in video are discussed, as well as the current trends in video database and evaluation methods.

Chapter 3, discusses the general framework for reference throughout the rest of the thesis including Binary Partition Tree (BPT) construction, general formula for merging and simplification, as well as the evaluation method. BPT construction was derived from previous work [4], evolutionary analysis for tree simplification was adopted from [7] and its extension in matching and volumetric approaches. The general metadata structure was designed in order to set the main destination of data obtained in all segmentation and simplification tasks.

Chapter 4, is a discussion about segmentation of a single frame, a comparison of three well-known pre-segmentation algorithms, namely the Watershed Algorithm (WS), Mean Shift Algorithm (MS) and modification of K-means called Simple Linear Iterative Clustering (SLIC), is carried out. In term of the merging process, various merging order formula are tested in order to identify the most reasonable formula. The evolutionary analysis proposed in [7] is carried out in three pre-segmentation algorithms and three merging order formulae. The spatial Region Adjacency Graph (RAG), binary partition tree for a single frame, is obtained from this chapter. It is recorded in region metadata for the designated database tables.

Chapter 5 is dedicated to discussing the identification of inter-frame region/superpixel correlation. This needs to be identified in order to record matching region in the current, previous and future frames. The temporal relationship is recorded in the metadata.

Chapter 6 is dedicated to discussing a different approach to segmenting video data. Instead of segmenting individual frames in time, a three-dimensional matrices is prepared to accommodate the entire video, and segmentation is carried out to that volumetric data. The segmentation result has three axes: namely spatial (x,y) and temporal (t); it is called supervoxels/volume. By performing this operation, the matching task discussed in Chapter 5 is no longer needed, but it requires that the video data is available entirely at the beginning of the segmentation task. The merging task is performed on the supervoxel with an additional parameter of merging order to take into account the direction of the supervoxel. Merging history is recorded in binary partition tree structure, however each node no longer represents a single region in a particular frame. Every node represent regions in a subsequent frame with consistent labels. The results of pre-segmentation, merging and simplification are recorded in the supervoxel and $svEdge$ table for the Volume Adjacency Graph (VAG).

Chapter 7, discusses utilization of recorded metadata obtained in a series of operations presented in Chapters 4, 5 and 6. A list of extended keywords, query execution strategy and a set of functions in order to decode particular extension Structured Query Language (SQL) keywords of spatio temporal operations are discussed. The results of the spatial, temporal and spatio temporal operations are presented.

Finally, Chapter 6 presents the conclusion, and possible future works.

The list of publications related to this thesis is presented in Appendix A.

Chapter 2

Literature Review

2.1 Introduction

This chapter discusses work related to this research. In the course of this thesis, segmentation, simplification, visual content metadata creation and retrieval will be discussed and an extensive exploration of segmentation techniques, visual content metadata archiving and retrieval will be carried out.

Multimedia content indexing and retrieval has received substantial attention from the research community. For instance, an initiative from National Institute of Standards and technology has been sponsoring annual text retrieval conference video evaluation (TRECVid) to promote research in video analysis and retrieval since 2003 [8]. They provide a large dataset of test video and lots of researchers participate in implementing their algorithms on video retrieval.

Video object segmentation is an important task to enable more complex video analysis.

The demand for object segmentation of visual content has emerged since the object video coding standard was proposed. Moving Picture Experts Group (MPEG)-7 enables an object annotation in a video. MPEG-7 does not, however, have standardized (automatic) extraction of audio visual (AV) descriptions/features, and a standard search mechanism [9]. Since then, much work has been done and a lot of progress made in video object segmentation, classification, description and searching.

Content based video modelling needs a preliminary task to divide content into smaller units of visual content. Small unit visual content is obtained from the segmentation task and highly desired to be semantically meaningful. However, It is hardly possible for an algorithm to produce a consistently meaningful partition for general purposes (this will be demonstrated in Chapter 4). Metadata modelling for video content, however, benefits greatly from a 'good' segmentation. There have been numerous attempts to define the metadata content of video, with some of the proposals employing automatic segmentation results as input, while others employ a human annotation task.

Some video segmentation techniques are extended from image segmentation methods. For example, [10] builds on the efficient graph-based image segmentation introduced by[11], extending it in three-dimensional space in video. Some techniques utilize the output of image segmentation carried out to every frame within the video. For instance, [12] introduce a video segmentation by acquiring ranked foreground object proposals using [13]. The top ranks of objects proposal are considered as the foreground. It is followed by searching the objects proposal in subsequent frames throughout the video. Video and image segmentation are discussed here as interconnected topics.

There are some review papers on segmentation such as [14, 3, 15] and [16]. [14]

focuses on classifying temporal video segmentation. This class of segmentation aims to identify a group of frames belonging to one shot, which is determined by the transition from one shot to another, called a 'cut'. Even though temporal segmentation has a different goal, in practice, object video segmentation often needs to process a group of frames between two cuts, or in one temporal segment as the result of temporal segmentation work.

A comprehensive review of moving object segmentation in video was written by [3]. They divided the moving object segmentation into two major classifications: motion based and spatio-temporal. Motion based is divided into two sub-categories, which are two and three dimensional motion. According to their work, the drawback of motion based segmentation is overcoming the problem of noise-sensitivity and inaccuracy. In order to deal with these issues, a number of works have proposed a combination of spatial and motion based methods called spatio temporal.

The recent work in object video segmentation review was published in 2013 [15]. The review focuses on dynamic/moving object segmentation in video. Inference based and feature based classification are the two main proposed classifications. Inference based is broken down into background subtraction and energy minimization, while feature based is broken down into depth information, motion and histogram.

[16] in Chapter 1 classify image/video segmentation into seven categories. These consist of data based, interaction based, feature based, inference based, space based, class based and semantic specific. The data-based mode considers the data types used in the segmentation tasks, such as nature, human or medical videos. The interaction-based mode is divided into two main categories: supervised and unsupervised. The

feature-based mode relies on a selection of features, such as colour, texture, intensity, shape or motion. The inference-based mode is divided into bottom-up (based on low level features such as colour, texture etc.) and top-down (based on high level concepts as a result of human annotation). The space-based mode is carried out according to motion or spatial and motion information. The class-based mode extracts a specific class of objects from video such as face, cars, buildings, etc. The last category is a semantic specific mode that aims to divide an image/video into meaningful segments associated with some semantics.

The purpose of this literature review is to provide an overview of the work related to every task in this thesis. Firstly, some classes of segmentation techniques in respect to spatial, temporal (motion) and spatio-temporal are explored. Secondly, the applications of binary partition trees as hierarchical segmentation of image and video are explored. Thirdly, the role of prior knowledge in segmentation is discussed in supervised and unsupervised segmentation techniques. The state of the art of video content indexing and retrieval are explored in later sections. Finally, there is a discussion of the existing technique compared to the solution proposed in this thesis.

2.2 Formal Definition

According to [2] partial segmentation is splitting a scene into non-overlapping partitions with respect to certain homogeneity criteria. A single frame of video is equal to a still image. A video is a sequence of images in a certain order. In two-dimensional space, a greyscale image can be defined as $f(x, y)$ when x and y are coordinates in horizontal and vertical directions. It can be extended for colour images $F(x,y) = \vec{c}$, with $\vec{c}$ as colour components, for instance, in Red Green Blue (RGB) colour model vector $\vec{c}$ consist of R, G or B elements. In video data, the definition can be extended to $f(x,y,t)$ for greyscale video and $F(x,y,t)$=$\vec{c}$ for colour video, where t denotes the temporal variable.

Segmentation of an image Img, is a finite set of regions $R_1, ..., R_n$

$$Img = \bigcup_{i=1}^{n} R_i\,, \quad R_i \cap R_j \ = \emptyset,\ for\ i \neq j \tag{2.1}$$

where Img denotes entire images, n is the number of partitions, R$_i$ denotes region i^{th} as a segmentation result, and every pixel belongs to specific regions. This formulation can be extended to three dimensional spatio temporal data:

$$Vid = \bigcup_{i=1}^{n} V_i\,, \quad V_i \cap V_j \ = \emptyset,\ for\ i \neq j \tag{2.2}$$

where Vid denotes the entire video, n is the number of partitions, V$_i$ denotes volumes/supervoxel i^{th} as the segmentation result, and every voxel (pixel in three-dimensional space) belongs to a unique partition. The main extension from regions to volumes is the temporal axis in three-dimensional data.

2.3 Spatial Segmentation Technique

2.3.1 Thresholding

Thresholding technique is a simple but effective way to divide data into some categories [2]. The main concept of thresholding is to select a threshold, where all pixels with the value greater or equal to the threshold belonging to a certain partition, while the rest are assigned to a different partition. The basic concept of the thresholding technique follows the formula 2.3.

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) \geq T \\ 0 & \text{if } f(x,y) < T \end{cases} \tag{2.3}$$

g is the output of thresholding of the original image f. Threshold T is applied to every pixel in $f(x,y)$ and assigns $g(x,y)$ to 1 or 0. Where (x,y) are horizontal and vertical axis on an image plane respectively. If $f(x,y)$ is above the threshold (T), $g(x,y)$ is set to 1, otherwise 0. It can be extended to multiple thresholds by defining n number of thresholds.

The correct selection of the threshold is critical for successful segmentation. Distribution of the grayscale value in a histogram can be utilized to guide a reasonable threshold. In ideal conditions, when the histogram is bimodal, the population of the foreground and the background forms separated peaks, and a minima exists in the valley between them. It is reasonable to select the minima as a threshold. The distribution may not be sufficiently uniform, however, in such cases applying a single threshold for the entire image is often unsuccessful [2]. A dynamic threshold needs to be implemented where the image is divided into a set of sub-images and local threshold is calculated for each sub-image.

In Otsu's thresholding method, introduced by [17], an analysis of the histogram is carried out to determine all possible thresholds, followed by calculating the variance of the foreground and background separated by the threshold. The calculation is carried out for each threshold in order to select the best value. [18] works on defining a threshold on an image with uni-modal distribution where the histogram may only contain one obvious peak. Threshold is determined by finding a corner in the histogram plot.

Dynamic threshold introduced in [19]. In this approach, the image is divided into a regular array of overlapping sub-images, and histograms are computed for each sub-images. It is possible in some sub-images for the histogram to be uni-modal; meaning that thresholds cannot be determined. The threshold for those sub-images is interpolated from neighbouring sub-images. In order to arrive at an effective way to select thresholds, a hierarchical technique is applied such as in [20, 21]. [21] proposes an iterative cluster merging using a dendogram in order to identify multiple thresholds in an image.

In video, thresholding has been applied in many ways. It can be applied for motion data as well as colour and the texture of the voxel in a video sequence. $f(x, y)$ is defined as a motion vector in the coordinate x, y and the threshold is applied as the certain value to distinguish between moving and stationary parts of the pixel in the current frame. For instance, [22] proposed an optimal threshold for frame difference, in order to determine segmentation of moving objects, while [23] proposed an adaptive thresholding in video segmentation in order to detect moving objects.

2.3.2 Edge Based

Edge segmentation is an early technique that remains important. It relies on and edge detection operator such as Robert, Laplace, Prewits, Sobel [2] operators etc. All operators aim to find the border between adjacent regions by detecting the discontinuities in the image. These can be in the greyscale, colour, texture, etc. The outcome of these operators cannot directly produce segmentation results due to the open boundary. Therefore, an additional task is needed to produce border construction of a region. Common problems in this technique are the existence of an edge where there is no real border, or a real border exists but no edge is detected. Both problems cause poor segmentation quality. Some proposed techniques devote significant effort to overcoming this drawback.

In video segmentation, detection of edges in the reference frame is usually followed by tracking the boundary in the subsequent frames. A trajectory of the boundaries is drawn along the temporal domain by tracking them using motion information. In [24] a binary model for the object of interest is derived from its edges, followed by tracking in the subsequent frames. [25] proposed a structured random forest to detect edges in a real time frame rate.

2.3.3 Graph-Based

This class of algorithm is derived from popular mathematical graph theory. A graph $\mathcal{G}$ contain two components which are vertices $\mathcal{V}$ and edges $\mathcal{E}$. The graph is defined as following formula:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}) \tag{2.4}$$

Some researches aim to obtain a single final set of partitions using some concepts from graph theory. Some examples of methods in this class are graph cut [26, 27], normalized cut [28, 29], and recursive shortest spanning tree (RSST) [30].

Graph cut is proposed to separate the foreground and background. In [26] using graph cut to segment the optimal separation of the foreground from its background, an interactive scenario is proposed, with a human giving a seed of the foreground and the algorithm. In a normalized graph [29], for example, proposed natural image segmentation by performing mean shift to get the initial over-segmentation, before then representing all regions in a graph and perform normalized cut algorithm to get globally optimized clustering. [28], meanwhile, proposed an algorithm to segment the most prominent moving group of pixels over frames. A graph of connected pixel in a spatio temporal neighbourhood is thereby formed. A segmentation problem is thus transformed to a graph partitioning problem. A normalized graph cut is proposed to eliminate the undesired bias of the minimum cut algorithm.

[10] proposed an extension of the efficient graph-based segmentation [11] in three-dimensional space. [31] proposed another extension of graph-based image segmentation, enhancing processing speed and minimizing memory consumption. They work in streaming mode, which only considers the previous frame to decide the segmentation of the current frame segmentation. They claimed a better performance of between one and four frames per second compared to one frame per second [10].

2.4 Motion Based Segmentation

The goal of motion-based segmentation is to segment a video sequence into multiple coherently moving objects. Motion information gives a lot of information about moving parts on the scene. If the goal of segmentation is to divide a scene into moving and static parts, motion is a prominent cue. In some computers vision, the moving part is assumed as the foreground and the static part as the background. There are three main issues: region support, motion modelling and segmentation criteria [3]. Region support could be individual point/pixel, corner, line or regions. Motion modelling could be two-dimensional or three-dimensional. Segmentation criteria can be Hough transform, expectation and maximization (EM) or maximum of posteriori (MAP).

There are some works on motion based segmentation such as [32, 33]. [32] propose a new motion modelling methods called motons, which builds motion models and gathers the motion in the video according to its motion model. [33] proposes a motion similarity criteria and uses the similarity to guide region merging.

2.5 Spatio Temporal Segmentation

The main idea of spatio-temporal segmentation is combining spatial and temporal information to obtain good segmentation quality. By combining both motion and spatial information, these techniques intend to overcome the over-segmentation problem in image segmentation, and to overcome the noise-sensitive and inaccuracy problems in motion-based segmentation [3]. There are four different groups of methods: background modelling, projection, matching and volumetric approaches. The last three groups of methods are also discussed in [34, 35].

2.5.1 Background Modelling

In the scene that was acquired by a fixed camera, a popular method to distinguish the motion part and the static part is performed by subtracting each individual frame from the background. The method is called background subtraction. The background is modelled, and the moving object is detected by subtracting the current frame and the background model. A result greater than a specific threshold will be determined as a moving object. The variation in this class is how to model the background and how to update the background model.

[36] models each pixel as a mixture of Gaussians and uses an online approximation to update the model. The Gaussian distributions of the adaptive mixture model are then evaluated to determine which are most likely to result from a background process. Each pixel is classified based on which Gaussian distribution represents it.

[37] models the background using codebooks in order to represent structural background variation in an efficient way and consume less memory. Each pixel is quantized into codebooks which represent a compressed form of background model for a long sequence. [38] proposed a background model according to accumulated frame difference information. A background registration technique is used to construct a reliable background image.

2.5.2 Projecting approach

In this approach, each frame is represented as two-dimensional matrices. The reference frame is segmented individually, and each region is tracked in the remaining frames. The algorithms in this method vary in terms of how to segment the reference frame, and how to make a projection to the upcoming frames. The region trajectory is often used in this approach.

[39] used the two-dimensional (2D) watershed in a series of frames, and projects the initial partition into upcoming frames considering motion information. The label estimation is conducted using Markov random field (MRF). [40] proposed iteratively merging over-segmented regions to form meaningful objects based on a mutual spatio-temporal similarities measure, which is a combination of temporal and spatial information in a statistical framework as a hypothesis test.

[5] proposed a trajectory tree as an object-oriented representation of a video. A binary partition tree of the first frame of a video is used as a reference to estimate the segmentation of the upcoming frames. It is followed by enforcing consistent labels for similar regions. A new node label is issued for unprecedented regions in the current frame. A single BPT makes it hard to represent a video with dynamic motions, however.

2.5.3 Matching Approach

In matching approaches, spatial segmentations for each frame are performed independently. Region features such as colour, shape, location, texture, or their combinations are used to match regions across frames. A temporal correlation between regions across frames is obtained by linking regions in the subsequent frame according to some criteria.

Generally this approach uses individual spatial segmentation for every frame throughout the entire video. A matching task is carried out to solve inter-frame region correlation. Inter-frame regional similarity plays an important role in this stage. The relationship between regions may be one to one, but there can also be many-to-many. This procedure can avoid the motion estimation task in the projection approach but, the lack of motion information may cause difficulty in determining moving objects, and may also need user intervention. This approach needs robust spatial segmentation to produce stable regions and may simplify partitions. An over-segmented and unstable spatial segmentation can cause many-to-many inter-frame regional correlation, which is computationally very costly.

There are some approaches to the processing of spatio temporal video sequences. Some researchers employed frame-by-frame processing, followed by region matching [41, 42]. In our work [43], we use the identification of salient regions in each frame to solve many-to-many relationship matching across frames. [44] extracts the visual object plane by matching 2D binary model.

[45] uses a graph-based matching procedure to establish temporal correlation between regions across frames. A hierarchy of nested partitions is used to resolve topology conflicts in graph matching. Their graphs are also endowed with a memory component to account for completely occluded regions which may reappear in the scene. Shape matching through the parametrization of region contours is explored in [46] for tracking applications.

[12] proposed a forms of video segmentation by making an assumption that a re-

gion that moves differently from its surroundings and appears frequently throughout the video will probably be salient or the main interest. They work in binary segmentation which divides a video into two regions: foreground regions and background region. The drawback of this approach can be seen when it is applied to a video containing many moving objects.

[47] proposed a tracking method, according to accurately segmented object boundaries. The first step of the proposed method is to model the object and background using a Gaussian mixture model (GMM), and extract a rough contour according to the object edge features. An elastic shape (modelled according to [48]) matching method is then applied to extract the exact contour.

[49] attempt to solve the problem of partial shape matching. They transform shapes into sequences and utilize an algorithm that determines a subsequence of a target sequence that best matches a query. They map the problem of the best matching subsequence to the problem of a cheapest path in a directed acyclic graph (DAG).

[50] represent the object to be tracked using a hierarchy of regions, each of which is described with a combined feature set of popular SIFT descriptors [51] and colour histograms. They formulate the tracking process as a graph matching problem using an energy minimization function. They use a graph updating mechanism to adapt the object evolution over time.

[52] introduces a region trajectory generation model based on graph clustering. They use Watershed to cluster each frame and employ a spectral embedding framework to cluster region trajectory and obtain meaningful objects. Affinities are computed based on motion similarities between point trajectories associated with the region trajectories.

The previous works [53] discuss the application of a genetic algorithm to establish matching nodes between selected partitions in a BPT of the current frame to regions in the next frame, while the [43] discuss selective top-most salient nodes in the current frames to the top salient nodes in the upcoming frame.

2.5.4 Volumetric Approach

A multiple frames video segmentation problem is considered as a spatio-temporal volume and solves a three-dimensional (3D) segmentation problem, in order to avoid frame-to-frame region matching. Every voxel (a pixel in 3D space) in the matrix have neighbours in the spatial and temporal direction. This can be achieved with an assumption that a complete video is available at the beginning of processing. The benefit of avoiding tracking regions, however comes with higher memory requirements due to the large data size. In [10], for example, 2.2 GB of memory is required to process 193,000 edges for one second of video of 25 frames.

[54] uses a 3D watershed segmentation by processing the pre-segmentation of individual frames using 2D segmentation. In order to avoid over-segmented initial partition, a topological simplification is achieved by removing particular local minima. Final segmentation is obtained by merging 3D watersheds in the spatio-temporal domain using a Markov random field (MRF) framework. [55] proposed a hierarchical mean-shift on a space time dimension. Every pixel is put in 3D space $f(x, y, t)$, with each pixel containing seven feature dimensions: three colour component (RGB), two motion angles, and two motion distances. The edge vector for each voxel is computed by the colour feature on the spatio-temporal volume proposed by [35]. It is followed by performing

spatio-temporal watershed over the topological surface defined by the edge vectors. Active surfaces have been implemented with a level set methodology for 3D segmentation [56]. In their work, segmentation accuracy is improved by combining occluded volumes and motion models for the object and the background.

[10], proposed a spatio temporal video segmentation on a graph-based image segmentation algorithm adapted from [11]. They extended two-dimensional region space segmentation of 9 neighbourhoods into three-dimensional space time on 26 neighbourhood.

[57] discuss volumetric hierarchical segmentation. This starts with an initial supervoxel prepared by the pre-segmentation algorithm. It is followed by an iterative merging task and recorded in a binary partition tree structure. All nodes represent a corresponding supervoxel in subsequent frames of the original video. A simplification algorithm considering spatial properties and motion speed and direction is also discussed.

2.6 Hierarchical Segmentation

2.6.1 Binary Partition Tree on Image

A binary partition tree (BPT) was used for image segmentation representation. There are other forms of tree such as max-tree, min-tree and quad-tree. Quad-tree is a tree structure in which every node has four leaf nodes, and in image processing is usually implemented to partition an image from within the entire image, whereby, in every iteration, the partition is divided into four partitions until a homogeneity criterion is achieved.

Max-tree, meanwhile, is a tree where the leaf nodes are at a maximum distance to the parent nodes. The inverse is true in the min-tree algorithm. Max-tree, min-tree and quad-tree have been demonstrated to represent image segmentation, as in [58]. Compared to these alternatives, however, BPT has a simpler structure and is therefore simpler to implement with a consequent lack of location information in the tree structure. Binary partition trees have been implemented in many areas of still image processing.

Since it was proposed in [4] as a framework for video and image segmentation, some work has used BPT for efficient representation. BPT creation consists of three main steps: pre-segmentation, merging and binary tree construction. An implementation of colour based segmentation, region merging and a user interface was developed in [7]. A framework of semantically meaningful image segmentation was proposed using an evolutionary analysis. They also provide a user interface for efficient browsing to determine a semantically meaningful object in the tree. Although this work successfully presents a good segmentation and, sometimes, a complete object that is semantically meaningful identified on one of the node, some objects are represented by some disconnected nodes on the separate branches of the tree. An object with salient colour dissimilar to the background is usually well segmented, but an object with a low colour distance to its background leads to a miss-merge. Close colour distance between semantic objects and the background leads to an object becoming fragmented into nodes on a number of parts of the tree.

The BPT was extended into the multi-dimensional Binary Partition Tree (MBPT) in [59] to convey not only colour but also texture edge and motion in order to produce more robust segmentation results. The algorithm significantly improves the segmentation result for images with good colour distribution compared to the previous approach which only uses colour feature. MBPT, however, cannot achieve better results in an image with poor colour distribution. Texture and edges can improve the final segmentation when the initial segmentation is good, but for poor initial segmentation, texture and edge cannot help that much. This could be because MBPT was developed using initial pre-segmentation based on colour distribution, while another feature is used in the region merging step.

[60] introduced object detection in a binary partition tree. In this work, shape information of the target object is needed. The additional shape information leads to intensive user participation, which can improve human subjectivity and labour cost. An extension of this work in [61] introduced descriptor to represent an object. Although their work is well presented, it is limited to finding objects that have been predefined such as sky, text signs or a face. Another approach proposed a contour detection in binary partition tree representation [62], using the difference between two regions as the boundary of a region. If the difference is high enough, the probability of it being the contour boundary is higher than for low differences.

[63] used a recursive spanning tree algorithm to split and merge a region in an image. This work use BPT to record the history of region merging. In order to identify an object, they define a stopping criterion, which is compared to a definition of an object. [64] proposed an algorithm to segment an object in a video sequence, using a modified recursive spanning tree algorithm (MRSST) and then binary partition tree representation as the result of MRSST for each frame. In the first frame, the user is allowed to identify

the foreground object, then, in the next step, the algorithm will track the foreground in the node of the BPT of the next frame .

2.6.2 Binary Partition Tree on Spatio Temporal Domain

Binary partition trees can be used either with spatial or spatio-temporal data. [65] proposed a motion based binary partition tree for video object segmentation. [5] segments the initial frame and then projects the region to the subsequent frame according to motion vectors. [66] examines the problem of the segmentation and tracking of video objects for a content-based information retrieval context. Initially, they use an active contour model that progressively refines the selection by fitting the natural edges of the object followed by object refinement using a BPT with a marker and propagation approach. The video object is tracked by using a hybrid structure alternately combining a hierarchical mesh for the motion estimation between two frames and a multi-resolution active contour mode.

2.7 Simplification

Simplification of partitions needs to be performed in order to achieve larger area segment in the visual space. This process is essentially needed when the primitive regions are too small. For example, table 2.1 shows a comparison of the number of primitive regions compared to the number of expected objects in the ground truth. It can be seen that, on average, a region in ground truth will be formed by around two hundred initial watershed regions [67]. Mean shift [68] and SLIC [69] shows smaller over-segmentation

Table 2.1: Pre-segmentation and comparison to the ground truth

Video	Resolution	Method	Partitions	Ground Truth (GT)	Over Seg
Soccer	288 x 352	watershed	4179	21	199
		SLIC	166		7.9
		mean shift	326		15.5
Stefan	240 X 352	watershed	4707	18	258
		SLIC	319		17.7
		mean shift	1043		57.9

rates.

The merging task commences with the initial partitions prepared by pre-segmentation algorithms. In doing so, stopping criteria and selection issues have to be considered. Stopping criteria dictate the decision of when the merging iteration should be terminated. Selection rule is responsible for deciding the pairwise partitions that have to be merged in a particular iteration.

In a complete tree, iteration is terminated whenever a root node is achieved. The root node represents the entire image/video; therefore, no significant information is carried by the root. This is because the purpose of segmentation is to break down the content into multiple objects. The important partitions are expected to be located in between the initial partitions (the lowest level child nodes) and the root.

A number of studies have been carried out in order to identify the important regions between the smallest unit (initial partitions) and the greatest root nodes. Researchers have made a significant effort to define merging and stopping criteria [70, 71]. [7], and [63] define a formulation of stopping criteria in order to avoid excessive merging between different objects into a single partition. They proposed summing up the cumulative merging cost during region evolution, stopping when the value reached a threshold based on a uni-modal threshold calculation [18]. [72] proposed a merging based on maximal similarity, they used LAB histogram distance as the merging criteria with help of user input of the background and foreground part of the image.

Many formulations of selection techniques have been explored and formulated as similarity measures. [4] initially proposed iteratively merging pairwise primitive regions using a colour similarity criterion. [58] introduced a proportional colour and partition size to eliminate small regions in around big regions. [59] modify the similarity criteria by including texture in order to get a better merging result. In order to improve the outcome, [73] propose an adjacency degree and area to be considered as a similarity criterion.

The task of merging aims to achieve the object candidate. For instance [74] proposed a bottom-up segmentation. In this approach, the initial segmentation is obtained by recursive shortest spanning tree(RSST), followed by a merging task using a combination of spatial configuration properties called $syntactic features$. Demster-shafer's theory is an alternative of generalization. Bayesian probability is used to decide pairwise region merging. [7] proposed a method to identify where the excessive merging occurs. It is performed by identifying any discontinuity of region evolution from the child nodes upwards to the root. [71] proposed an identification of salient regions by surrounding saliency measures of every nodes against global colour.

2.8 Availability of Prior Knowledge

2.8.1 Supervised

Semantic segmentation, or complete segmentation [2], is unlikely to be achieved without the availability of prior knowledge. Supervision can come from user intervention or a particular scenario. The presence of humans in the loop can give a guidance to the segmentation algorithm where the human interest is located in the scene. In video segmentation, a user intervention scenario was proposed, such as in [75, 76, 60], while in image segmentation user supervision was proposed, for instance in [26, 77, 78]. Some segmentation methods are proposed to extract specific objects such as humans, cars, roads or vegetations from input images/videos [79, 80]. For instance, in vehicle segmentation in traffic video surveillance scenarios, the shape can be observed from samples, therefore the algorithm can decide efficiently where the vehicle region is. One may argue that this is classified as a different category rather than supervised. In [16], this scenario is classified as a class-based mode. Some researchers also claim their approach as semi-supervised due to the limited amount of supervision [81, 82]. They proposed a segmentation seed at the beginning of video, followed by posterior inference of unlabelled pixels from the tree-structured model in the remaining of frames.

Some work that involves humans in the loop in order to supervise the segmentation varies in terms of intervention intensity. [78] proposed a supervised video segmentation by minimizing user interaction. The first frame is segmented using the mean shift algorithm and by letting the user spot meaningful partitions. The algorithm then tracks the partition in the rest of the frame. [83] uses the same scenario, with an improvement in inter-frame region matching. [76] proposed user intervention to identify the edge of the region in the first frame and then to track this over time in the entire video. [84] proposed a user-assisted split and merge segmentation based on long term motion affinity. They proposed an affinity measure on a temporally disjointed track.

Generally, supervised segmentation produces accurate, or even semantically meaningful, results. Post processing is not needed and the speed of computation can be improved. On the other hand, it requires human intervention which causes extensive labour work. Much effort has been devoted to minimizing the level of intervention. Another supervised segmentation scenario is to limit the target objects, giving the system knowledge about the features of the target. While this works in certain scenarios, it cannot be applied for general use since the algorithm will probably fail to cope with a different set of the target objects whenever the input video is changed.

2.8.2 Unsupervised

The absence of prior knowledge can broaden application possibilities. Fully automatic processing and a general scope without any limitation of specific scenarios can be achieved in an unsupervised scheme. Due to the lack of prior knowledge, the outcomes are likely to be fragmented into small semantically meaningless partitions, and therefore, the result may need post-processing to make a strong correlation with a semantic object in the real world. In some applications, where semantic partition is not the main goal, such as video compression, post-processing does not need to be performed.

A segmentation can be without any specific purposes. In this class of segmentation, the input is arbitrary video and the algorithm works solely based on the information inside the video. [85] proposed an analysis of point trajectories in this context, while [86] proposed fast unsupervised video segmentation according to motion feature.

[12] introduced an unsupervised video segmentation by assuming a region that moves differently from its surroundings and appears frequently throughout the video will probably be main interest. This work identify the object proposal as foreground regions and a background region on the first frame. Object proposal is obtained by rank the regions in intra-frame appearances. The main drawback of this approach is when the video contains many moving objects.

Some works in graph-based segmentation such as [10, 11], are work based on unsupervised scenario. The vertices in the graph iteratively merge based on internal and external variations. They keep the segmentation to be unsupervised, which lets the algorithm decide the regions without prior knowledge. The segmentation results rely on colour, texture only, moreover optical flow is added to get a better result. They implemented a hierarchical segmentation rather than tuning the parameters in order to keep the small homogeneous regions in the result.

In general, unsupervised segmentation can work without human intervention, and therefore fully automated. It can also work in an unlimited class of the object so it does not need a training phase. As a consequence, the accuracy of the result cannot directly conform with the semantic concept in the real world. In case the application needs a semantic object, post-processing often needs to be carried out. Some applications for semantic content such as summarization, recognition and other content based multimedia services, need post-processing tasks before the segmentation result can be effectively used. Some applications do not really need an accurate semantic object such as video coding and compression, and in these circumstances post-processing may not be needed.

2.9 Visual Content Indexing and Retrieval

Video content extraction, indexing and retrieval has attracted significant attention in the research community. Video carries rich information, entailing massive raw data with high redundancy. In general, the effort categories in dealing with video content retrieval are structure analysis, feature extraction, data modelling and retrieval.

2.9.1 Video Structure Analysis

Video can be temporally structured into a scene, shot and frame. A shot consists of consecutive frames captured in a single camera event. It begins and terminates with a shot boundary or ‘cut’. There are various ways to detect shot boundaries, such as utilizing colour histograms [87], block colour histograms, edge change ratios, motion vectors [88], or graph distance [89]. In order to discover shot boundaries, a threshold [90, 91] or statistical learning methods are utilized [92].

Within a shot, frames are highly redundant: i.e most of the content in the current frame is inherited from the previous frame. A key frame is particularly selected to reflect the rest of the members in a single shot. Many methods have been introduced to extract such key frames [93].

The scene is a higher level of structure that consists of many consecutive shots, and contains a single semantic story. In much of the research, different criteria are proposed for scenes. For example, in [94] a scene is defined as shots with similar key frames, whereas [95] determine a group of shots as a scene if their backgrounds have some degree of uniformity.

2.9.2 Visual Feature Extraction

Feature extraction can be carried out to the key frame, the object in the video and the temporal information. Key frame feature extraction can be performed with static features such as colour distribution [96], texture, shape or edge histograms [97]. Once the feature is extracted, it can be stored in the database, and retrieval can be performed according to the feature.

Object feature extraction is performed to detect particular objects appearing in the video. For example, [98] detect a face in the video and index the face in order to serve a searching mechanism. [12] extract an object proposal and operate this across frames throughout the video.

Motion is an essential characteristic of video that differs from the image. Statistical features of motions can be extracted such as in [99]. Trajectory based motion features are extracted by modelling object trajectories such as in [100].

2.9.3 Query and Retrieval

Following feature extraction, the feature can be stored in a database and content retrieval can be carried out. [101] proposed an object video database system (OVID) that is based on an object oriented database management system (OODBMS). In their model, a video consists of objects and every object can have varied attributes. They focus on defining the video data object model, while the content description is produced by human. [102] shows a spacial query in order to cope with Geographic Information Systems (GIS) data in a spacial database. [103] proposed a spatio temporal data modelling and query. Although it was not intended to deal with the object in the video, it was designed to manage moving data in the database management system(DBMS).

The Informedia projects use speech recognition, image processing, and natural language understanding automatically to produce metadata for video libraries [104]. According to their report, although the speech, vision and language processing are imperfect, metadata with some degree of inaccuracy still can be very useful for information retrieval.

[105] discusses spatio temporal queries and introduces some specific spatial keywords such as AREA, INSIDE and temporal keywords such as DURATION, CONTAIN, and MOVING DISTANCE. [106] develops a video database management system called "Billvideo" equipped with rule based queries to deal with spatio temporal video databases [107]. An extension of Billvideo proposed in [108] supports the MPEG-7 XML standard. In Billvideo, human interaction is needed to define the content.

[109] introduces spatio temporal region graph query languages (STRGQL) for a video database. This is based on their concept of spatio temporal region graph indexing [110] Their work builds on the frame based segmentation that is performed in Mean shift [68]. Every frame has a particular spatial region adjacency graph (RAG) and a relation between nodes in the graph is identified as a temporal graph. They use Standard

Query Language with some extra keywords in order to perform graph functions: graph similarity measure (GDM), graph edits distance (GED) and summary.

Recently, [111] introduced a database management system based on postgreSQL to manage multimedia data called ADAM. They empower the SQL with an additional distance measure in order to deal with similarity searching in a multimedia database. Feature extraction is carried out using SIFT [51]. They focus on image collection in their database.

2.10 Discussion

Visual content indexing and retrieval has received a lot of attention from the research community. A good partition is a key success factor for object-based video indexing. A number of segmentation algorithms have been introduced, but no general purpose algorithm can produce human quality segmentation. Given the rapid growth in the amount of video data, there is significant demand from the perspective of video management to store, annotate and retrieve data efficiently. Human made annotations are no longer practical, and therefore an automatic annotation tool is highly desired.

To develop automatic annotation tools, a minimum user intervention algorithm is needed. To provide a general purpose tool, a domain limitation has to be alleviated. This requirement can only be fulfilled by an unsupervised approach. The lack of prior knowledge, however, leads to poor quality of segmentation that is far from the semantically meaningful partitions, which are desired in an annotations tool.

Semantic is not a single concept, there is always multi-scale information contained in it. For example, a human consists of a head, a body and feet, moreover a head can be divided into eyes, mouth, nose and so on. Therefore, partitioning a visual space into a single concept is unreliable, this highlighting the need for a hierarchical representation.

Reliance on unsupervised segmentation techniques to provide content metadata is unrealistic, however, in general, the segmentation algorithms produce too many segments compared to what is needed, and simplification tasks have to be undertake to resolve this issue. When simplification is carried out, for example by setting the number of segments that are expected [4, 10], this leads to a loss of some detailed content. Another approach to simplification is to identify the important parts of the visual space, such as in [12, 13, 112]. This opens up the possibility of loss of information from the background or non-salient parts of the scene; therefore, there is a necessity to identify the essential parts whilst at the same time keeping the detailed information in a single representation. A binary tree representation and simplification offers both properties. The lowest level of the tree enables a detailed boundary to be stored, while the upper level of the tree saves the more general content. Although the BPT cannot store real object trees as demanded by the semantic concept, it does offer the possibility to store the detail and the global information in multiple levels in a single structure. Moreover, the important object candidates can be detected and recorded in the hierarchical structure. A multiple level of saliency will be recorded as a property of the node of the tree.

This research aims to extract objects candidate from the video. Current research provides feature extraction from key frame [96, 97] or limited object from the video [98]. This research offer a general solution to store all objects candidate and keep temporal information in a hierarchical representation.

Finally, the set of nodes of the hierarchical structure will be recorded in metadata. A necessary conversion, such as RGB code to colour conversion, polar coordinate direction can be provided. The metadata is expected to provide an intermediate visual content representation. Compared to available video databases, such as Billvideo [106, 108], where human intervention is needed to provide semantic descriptions, this works totally independent from user intervention. It is also free from semantic limitation, but it is not designed to answer exact semantic requests.

Chapter 3

General Framework

3.1 Introduction

In order to allow users to request the video content through metadata, there are several tasks that need to be carried out. This chapter discusses the methods and formulations explored in the remaining chapters. Firstly, a detailed discussion of pre-segmentation and partition merging is provided. Secondly, the benefits of salient node detection over Binary Partition Tree, tree pruning and simplification is outlined. Finally, a metadata and content query language are defined. The detailed implementation and the potential variations are discussed later in Chapters 4, 5, 6, and 7.

There are several tasks to be carried out in the thesis as a whole. These are:

- Pre-segmentation
- Set up Partition Adjacency Graph

- Partition Merging and recording in a binary partition tree
- Salient Partition Identification
- Tree Simplification
- metadata modelling
- SQL-like formulation and execution strategy

The expected final result is a partition database which can be accessed using an SQL-like query language for spatio temporal data. The database is expected to serve as further research in region space processing for video. A schematic diagram of all the tasks required can be seen in the Figure 3.1.

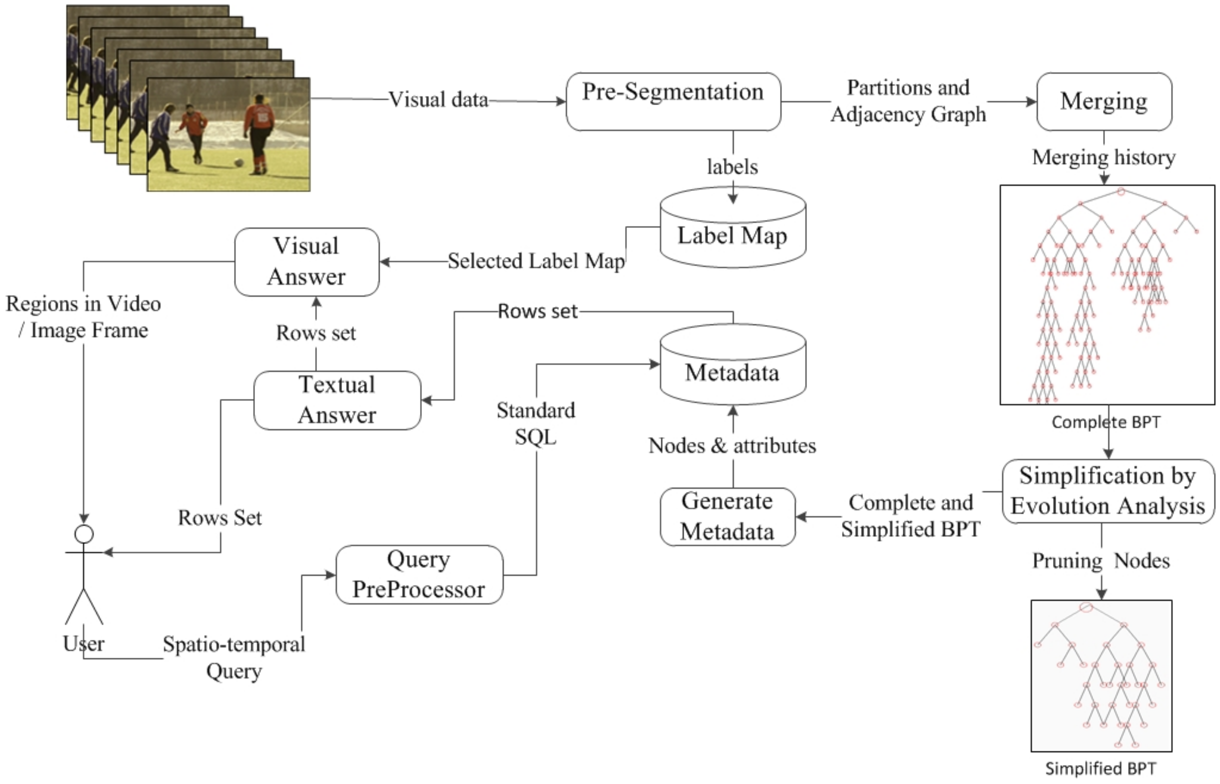
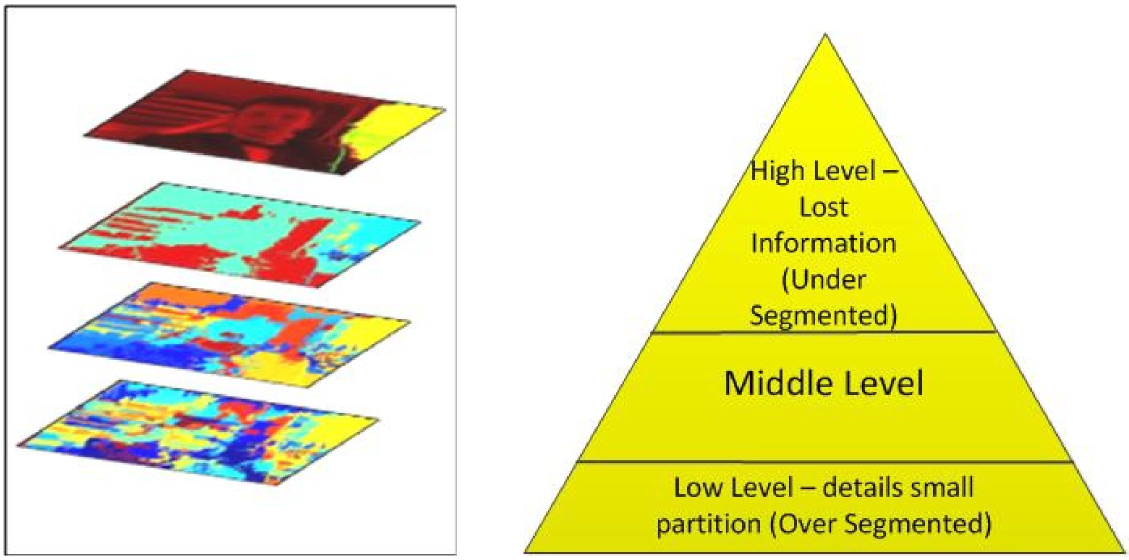


Figure 3.1: Proposed Framework

3.2 Hierarchical Segmentation

This work is based on general split and merge segmentation theory [2]. The first task is to split the visual space into homogeneous partitions. This task is followed by a merging process in order to obtain greater partitions that are expected to be closer to semantically meaningful partitions. Producing a semantic partition is difficult task, particularly when there is a lack of prior knowledge. Instead of making an effort to provide perfect segmentation, a multi-scale segmentation is considered to be a better option. It is arguable that in multi scale segmentation, detail level segmentation still being kept while at the same time less detail can be provided. Figure 3.2 illustrates the idea of multi level segmentation.


High Level – Lost Information (Under Segmented)
Middle Level
Low Level – details small partition (Over Segmented)

(a) Multi-scale Segmentation

(b) Multi-scale segmentation pyramid

Figure 3.2: Illustration of Multi-scale Segmentation

In the implementation, the lowest level of the segmentation is produced by a pre-segmentation algorithm. The higher level of segmentation is obtained by merging the selected pairwise partitions at every step of the iteration. At every step of the iteration, therefore, a pair of partitions is allowed to merge, then the merging history is recorded to form a BPT. Based on the BPT structure, a metadata of the video is generated.

3.2.1 Pre-Segmentation

The purpose of the pre-segmentation is to provide the initial partitions for the merging task. Closed boundary and non-overlapping partitions are expected. Over-segmentation is acceptable to some extent, because further merging processes will combine them to form a larger partition. Low boundary recall in the initial partitions is, however, highly undesirable. These methods are expected to work in two-dimensional and three-dimensional spaces for image and video data.

Segmentation is a well-researched area, and numerous methods have already been proposed by researchers. These can be classified into categories such as edge-based, region-based, and graph-based thresholding.

Thresholding classifies pixels within an image based on their values compared to a specified threshold. The threshold approach is simple, so it has low computing complexity. There are many ways to identify the threshold, although the optimal threshold is usually obtained by histogram analysis. Generally, thresholding works well for the simple images, in which objects can be clearly separated by their features. Thresholding fails to work in the noisy conditions that are normally present at natural images. Otsu [17] method is one example of this class. The computation complexity of Otsu method is

$O(L^{m-1})$ where L is number of gray levels and m is number of applied thresholds [113].

Edge-based segmentation utilizes the edge detection algorithms. The image region is defined by its border in this method, but the closed boundary partitions cannot be defined straightforwardly since the segmentation result is produced in the disconnected boundary. Hence, post-processing is needed to join the edges to form a closed boundary region. The advantage of this type of algorithm is that it works in a way that is similar to the human process in responding to visual signals. In general, this method works successfully in segmenting high-contrast images, but it suffers from noise. Hence, such a method fails to produce good results on images with smooth transitions and low contrast.

Graph-based algorithms use vertices to represent the pixels with the similarity between pixels being defined as the edges. Partitions are created by minimizing a cost function defined over the graph. Normalized cut (N-CUT) [114] recursively divides the graph to minimize the global cost function using contour and texture cues. The number of vertices is initially equal to that of pixels, causing a significant memory and computational requirements, and making the performance of the algorithms to be slow. The complexity of this algorithm is $O(N^{3/2})$, in which N is the number of pixels [115]. It is designed to work with images to produce regions/superpixels.

Then, an efficient, graph-based (EGB) approach is proposed by [11]. This run in $O(N \log N)$ time for N pixels image. The algorithm selects the merging edges to be exactly those that would be selected by Kruskal's algorithm for constructing the minimum spanning tree (MST) of each component. The algorithm works for images and produces superpixels.

An extension of [11] in three-dimension data is proposed by [10]. The algorithm is called efficient, hierarchical, graph-based video segmentation (EGBHV). This algorithm is working on video data with complexity $O(N \log N)$. Since video data comprises a huge number of pixels, the memory requirement of this algorithm is very high.

The region growing method class begins with specific seeds in the image and iteratively includes the pixels near to the seed. The iteration will be stopped whenever a convergence criterion is achieved. This class has a lot of variations in defining the seed and the technique to include the neighbouring pixels to a certain segment. Mean shift, SLIC and watershed are categorized in this class.

The Watershed algorithm [67] is a fast region growing algorithm. The seeds are defined as local minima in the image, whereas the regions are formed by joining the neighbouring pixels inside the catchment basin until neighbouring catchment basins started to merge. Watershed is fast with complexity $O(NlogN)$. It works on an image in two-dimensional space and video in three-dimensional space time. Despite its over-segmentation issue, it is a good initial segmentation to start merging process due to high speed, and boundary recalls rate.

The mean shift is a clustering algorithm based on a seed randomly chosen centroid of the cluster. This is gradually moved (shifted) to the new value until the value attains stability. Like the Watershed, it works independently without the need to set the number of expected segment. The classic mean shift suffers from over-segmentation issues, however. [68] proposed edges as an additional control of the shift in order to generate more reasonable segments. According to [69], the complexity is $O(N^2)$, and therefore, it is slow.

K-Means is classified as a clustering method, in which k number of data points (pixels) are selected as the cluster means. The pixels around the proposed cluster means are evaluated and assigned to the closest ones. When all pixels are assigned to some clusters, the means of each cluster are calculated, and the assigning process is repeated. The iteration will be stopped when the means achieve a stable value. The problem with classic K-Means is complexity because of the evaluation of all pixels against means candidates. Computational complexity is $O(KNI)$, where K is expected number of the partitions and I is the number of iterations. SLIC [69] is an extension of the K-Means algorithm with a limited searching window. As a result, the complexity can be reduced to $O(N)$, and it is invariant to image complexity. This algorithm is fast, and it works on both image and video. Although the need for K as an input can be considered to be a disadvantage, it can be eliminated by setting the value which is high enough for an over-segmented result to be expected in process of the initial merging.

Pre-segmentation is theoretically a segmentation, but in this thesis, the term pre-segmentation is used to distinguish it from the hierarchical segmentation that will be discussed later. The term pre-segmentation is used because the result needs further processes in order to obtain better partition. The goal of pre-segmentation is to cluster pixels into homogeneous partitions. The pre-segmentation algorithms are most likely to produce over segmented partitions because of the absence of prior knowledge. The absence of prior knowledge is necessary to keep the algorithm working independently without human intervention or limitation on account of any assumption.

The comparisons of some segmentation algorithms are presented in Table 3.1. Some algorithms work on 2D data only and the other work on both spatial and volumetric data. Thresholding, in general is very fast, for example, Otsu method depends on the

Table 3.1: Pre-Segmentation Algorithm Comparison

No	Category	Algorithm	Complexity	Spatial	Volumetric
1	Thresholding	Otsu	$O(L^{m-1})$	yes	-
2	Graph based	NCUT	$O(N^{3/2})$	yes	-
3	Graph based	EGB	$O(N \log N)$	yes	-
4	Graph based	EGBHV	$O(N \log N)$	-	yes
5	Region Growing	Watershed	$O(N \log N)$	yes	yes
6	Region Growing	SLIC	$O(N)$	yes	yes
7	Region Growing	mean shift	$O(N^2)$	yes	yes*

* Mean shift for 2D and 3D data provided in different implementations

N = number of pixels

L = number of grey levels

m = number of applied threshold

number of gray levels and thresholds. EGB and EGBHV provide a hierarchical segmentation where a number of levels are prepared with different granularity. In the pre-segmentation task, however, the expected result is a single final partition. Closed boundary is needed as initial partitions to start merging process, therefore, edge based segmentation is not considered. Watershed, mean shift and SLIC algorithms are selected mainly because they are operated on spatial and volumetric data, and provide closed boundary final partitions. Watershed and SLIC are operated on spatial and volumetric video data in single implementation, while mean shift on spatial [68], and volumetric in [116].

Video is considered in two different ways, framed-based and volume-based. Firstly, a frame-based approach considers video as a stack of frames. A frame is equal to an image in the horizontal (x) and vertical (y) axis. Segmentation is carried out on each frame, followed by identification of temporal links across frames. In the second approach, a video is considered as data in three-dimension matrices in the horizontal (x), vertical (y) and temporal (t) axis. Segmentation is performed directly to the three-dimension matrices. This approach avoids temporal link calculation across the frame.

3.2.1.1 Watershed

The watershed concept was first applied by [117] to segment bubbles on scanning electron microscopy (SEM) metallography pictures. It is brought to image processing in the field of mathematical morphology. The immersion model was proposed by [67], and the topographical distance approach was proposed in [118]. The Watershed algorithm works fast enough, but it yields a vast number of small regions. The illustration of this algorithm for one-dimensional data can be seen in Figure 3.3. The concept is also expanded into two-dimensional (2D) and three-dimensional (3D) data. Watershed produces superpixel forms in 2D spaces, and supervoxels in 3D space for video.

A 2D watershed for a gradient image of 288 x 352 pixels can be seen in Figure 3.4, and they consist of 4471 superpixels. This work will not focus on what is going in the watershed transformation, but we use the result for further processing. Further explanation refers to [67], [118].

Although the concept of watershed is quite simple, the algorithm is complex. [117] introduce an algorithmic description of watershed. Let us consider Ig to be a grayscale

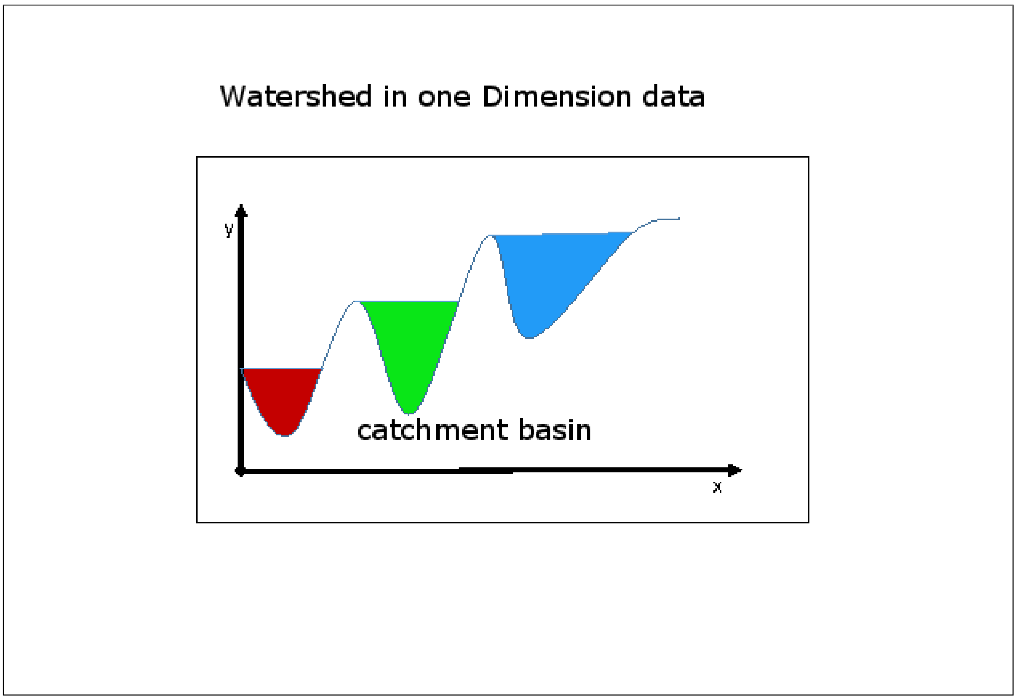


Figure 3.3: Ilustration of The Watershed of one-dimensional data

image whose definition of domain is denoted as $D_{Ig} \subset Z^2$. The value of Ig is discrete in a certain range $[0, N]$ and N is a positive integer, for example, 255.

$$Ig(p) = D_{Ig} \subset Z^2 \mapsto \{0, 1, ..., N\} \tag{3.1}$$

Let Gr denote a particular digital grid, Gr is a subset of $Z^2 \times Z^2$. A path Pa of length l between two pixels r and q in image Ig is a $(l + 1)$ number of pixels $(p_0, p_1, ..., p_{l-1}, p_l)$ such that $p_0 = r$ and $p - l = q$ and $\forall i \in [1, l], (p_{i-1}, p_i) \in G$.

Let $l(Pa)$ be the length of path Pa and $Ng(p)$ are the neighbours of pixels p. Consider M is minimum value of Ig at altitude h and to be a connected plateau of pixels with value h in which it is impossible to reach a point of lower altitude without having to climb.

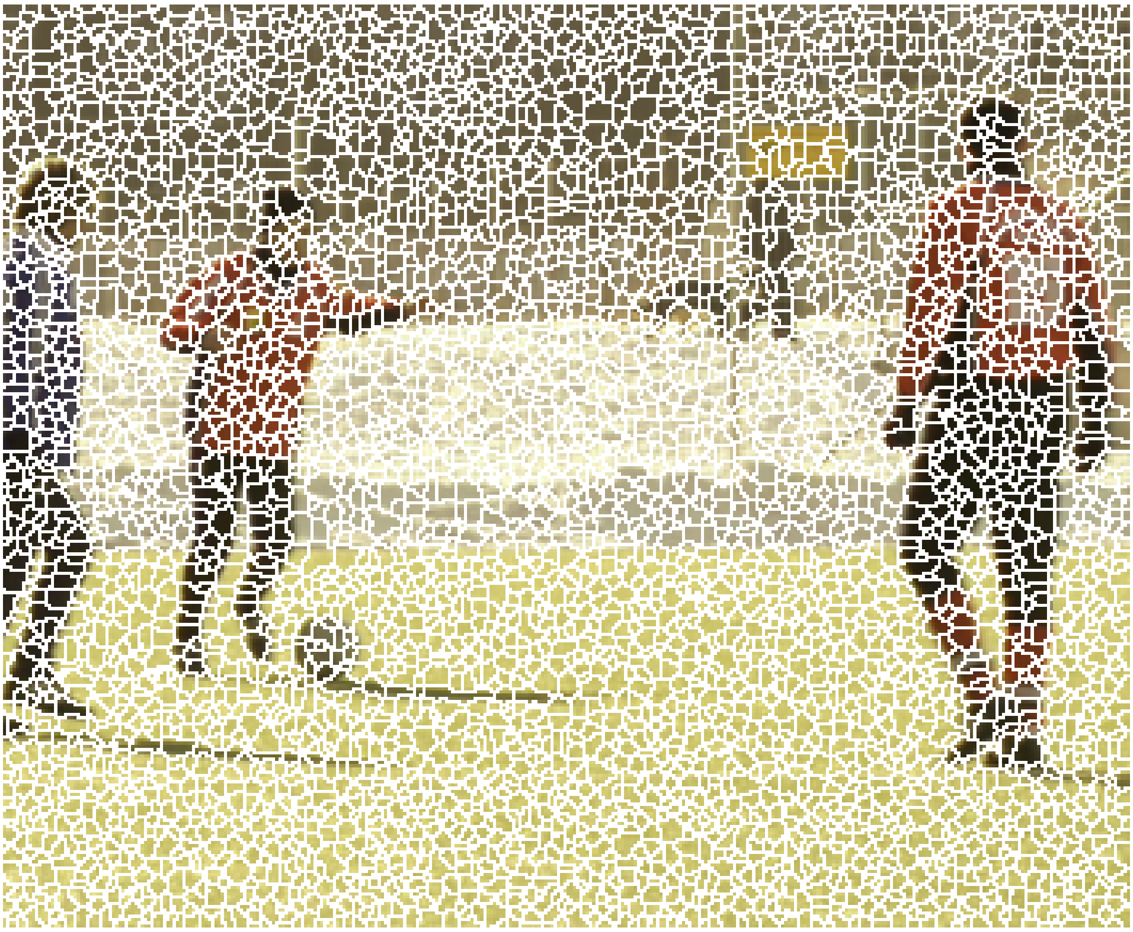


Figure 3.4: Example Result of The Watershed Algorithm of Frame 1 of the ‘Soccer’ Video

$$\forall p \in \forall r \notin M, \text{such that } Ig_r \leq Ig_p \tag{3.2}$$

$$\forall Pa = (p_0, p_1, .., p_l)\text{such that } p_0 = r\text{and } p_l = q \tag{3.3}$$

$$\exists i \in [1, l]\text{such that } Ig(p_i) > Ig(p_0) \tag{3.4}$$

The immersion process can be formulated as:

$$Th(Ig) = \{p \in D_{Ig}, Ig(p) \leq h\} \tag{3.5}$$

Where $Th(Ig)$ denote threshold of Ig in level h. The Catchment basin is defined in 3.6.

$$Ch(M) = \{p \in C(M), Ig(p) \leq h\} = C(M) \cap Th(Ig) \tag{3.6}$$

Where $C(M)$ denotes a catchment basin with minimum M, $Ch(M)$ is the subset of $C(M)$ which is less than or equal to h. The immersion process starts from Th_{min} as the smallest minima. The threshold is gradually increased and, as a result, a set of catchment basins are formed. The watershed lines are a set of pixels that do not belong to any catchment basin at the end of iteration.

3.2.1.2 Mean Shift

The third algorithm tested is the mean shift. No parameter input is needed to execute the algorithm. [68] introduced an implementation of the mean shift algorithm with considers edges to improve the output, called 'EDISON'. This is publicly available, and demonstrates good performance in speed and quality of the output. The example of first frame 'Soccer' video can be seen in Figure 3.5. It is still slightly over segmented, especially in parts of image with high variation such as the player. That part consists of a combination of red, black and grey regions.

Consider an image formulated as feature vectors, an image $Img((\vec{x}))$ has a feature

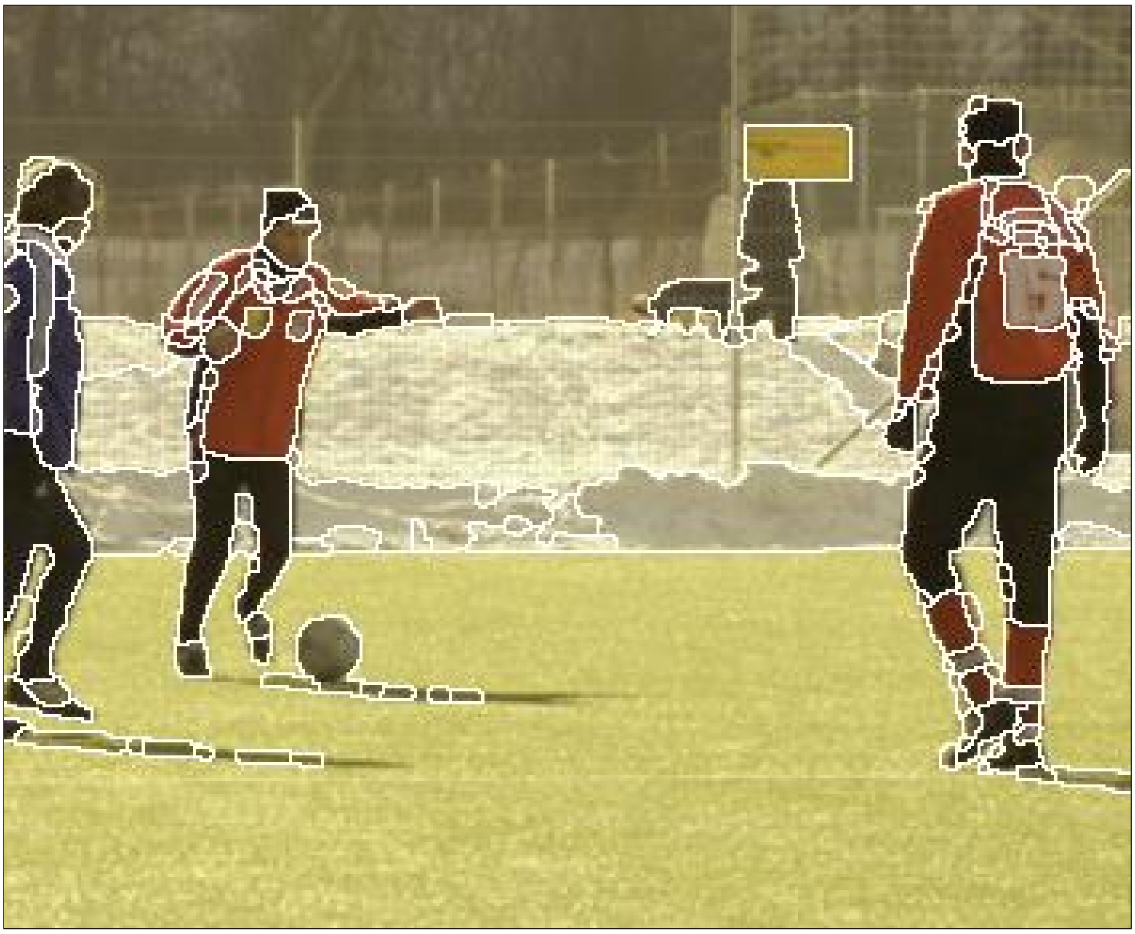


Figure 3.5: Example Result of Mean Shift Algorithm of Frame 1 of 'Soccer' Video

vector $\vec{F}(\vec{x})$ which is formulated in 3.7.

$$\vec{F}(\vec{x}) = \begin{pmatrix} \vec{x} \\ Img(\vec{x}) \\ L(\vec{x}) \end{pmatrix} \tag{3.7}$$

$\vec{x}$ is a pixel inside the image, $L(\vec{x})$ is local image features, such as bandpass filter response. The segmentation algorithm is operated to cluster the image features into reasonable homogeneous partitions.

The mean shift segmentation algorithm considers the probability density function

(PDF) of feature vector $\vec{F}(\vec{x})$ from formulation 3.8 computed from the image. Kernel-density estimates are used with respect to the following equation:

$$pK(\vec{F}) \equiv \frac{1}{|X|} \sum_{\vec{x} \in X} K(\vec{F}g - \vec{F}(\vec{x})), \text{ with } \vec{F} \in R^D \tag{3.8}$$

where $\vec{x}$ is the pixel in the image and $|X|$ is the number of pixels, while $K(\vec{e})$ is a kernel.

The mean shift alone produces highly over-segmented regions, and in order to combat this problem, in EDISON implements the salient edges to control the weighting parameters. The example in Figure 3.5 is a result of mean shift implementation considering salient edges.

3.2.1.3 Simple linear iterative clustering (SLIC)

The K-means algorithm is successful in many clustering applications, but at high computational cost. In order to reduce this, a modification is proposed by [69] reducing the size of the search window. The modified algorithm is known as simple linear interactive clustering (SLIC). It produces a controlled size of partition and keeps the boundary. The example segmentation result for the first frame of the 'Soccer' video can be seen in the Figure 3.6, it consists of 500 regions in almost equal sizes.

Besides working in 2D data, SLIC can also work in 3D data and produces supervoxels. It will be implemented in 3D video segmentation to complement 3D watershed.

3.2.2 Adjacency Graph

The pre-segmentation step produces a vast number of partitions. Every partition has its neighbours; in graph $\mathcal{G}$ theory, each partition can be considered as vertex $\mathcal{V}$, the re-

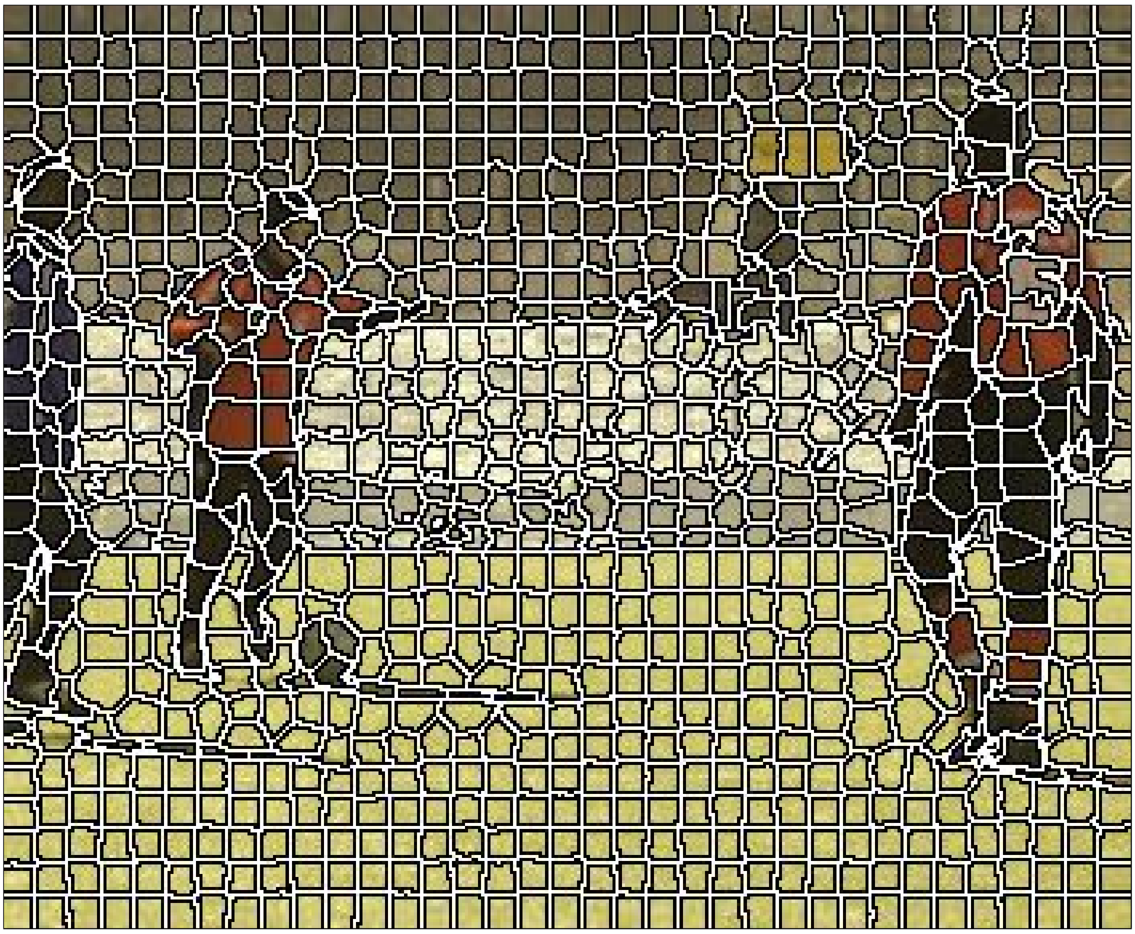


Figure 3.6: Example Result of SLIC Algorithm of Frame 1 of 'Soccer' Video $K = 500$, Window 10 x 10

lationship between a pairwise neighbouring partition can be drawn as an edge $\mathcal{E}$. The relationship can be colour similarity, area difference, common boundary, histogram similarity, texture similarity, moving direction and velocity and many others.

Graph representation is adopted to draw the relationship among partitions either in image or video. There are three adjacency graphs which are:

- Region adjacency graph (RAG) for 2D segmentation of a single frame.

- Volume adjacency graph (VAG) for 3D segmentation of a sequence of frames.

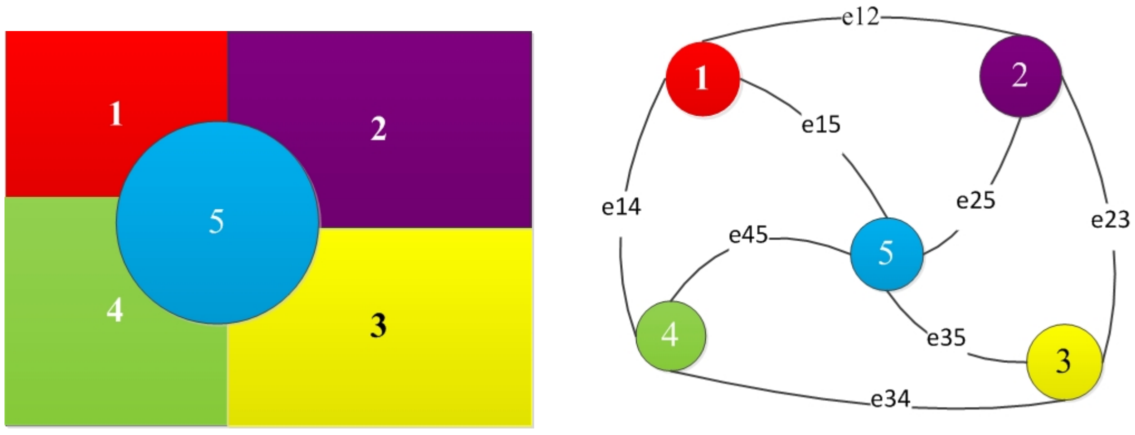
- Spatio Temporal Region Adjacency Graph (STRAG) for recording spatial neighbourhoods among regions in a frame and temporal correlation across frames.

The general formulation of a graph is given in the equation below:

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \psi) \tag{3.9}$$

Where ψ is a function to produce the attributes of the edge such as colour distance, neighbour orientation, histogram difference and other required attributes.

3.2.2.1 Region Adjacency Graph



(a) Segmented Image

(b) Corresponding RAG

Figure 3.7: Illustration of Image Segmentation and Corresponding Region Adjacency Graph

A region adjacency graph for a single frame records the spatial relationships between regions. Pairwise similarity is recorded as edges in this graph in order to guide the merging task in BPT creation of an individual frame. Figure 3.7 illustrates the region adjacency graph. As can be seen in Figure 3.7(b) there are five vertices that represent regions in Figure 3.7(a). Every edge has a unique identity and value, that calculated depends on a particular similarity measure.

The illustration of the data referred to in Figure 3.7 is described in Tables 3.2 and 3.3. Every region is recorded, and is assigned a unique node number. Every neighbouring relationship is illustrated in Figure 3.7 and is recorded as a unique edge. Table 3.2 not only record the initial regions as the result of pre-segmentation tasks, but also new regions obtained in the merging task. As can be seen in Table 3.2, a new issued region number is 6 (yellow highlighted), which is not exist in Figure 3.7(a). This is the merging result between regions 1 and 2, because the edge between them (e12) has the smallest distance. As can be seen in table 3.3, there are two edges e12 and e25 have the smallest distance at 180.03, but because in every iteration only one pair is allowed to merge, e12 is selected to merge to produce region 6. The features of region 6 are a combination of those of 1 and 2, therefore the area of region 6 is an accumulation of regions 1 and 2. When region merging occurs, the original region parent are assigned, therefore, regions 1 and 2 share a similar parent which is region 6 (yellow highlighted in parent column).

3.2.2.2 Volume adjacency graph (VAG) for 3D segmentation

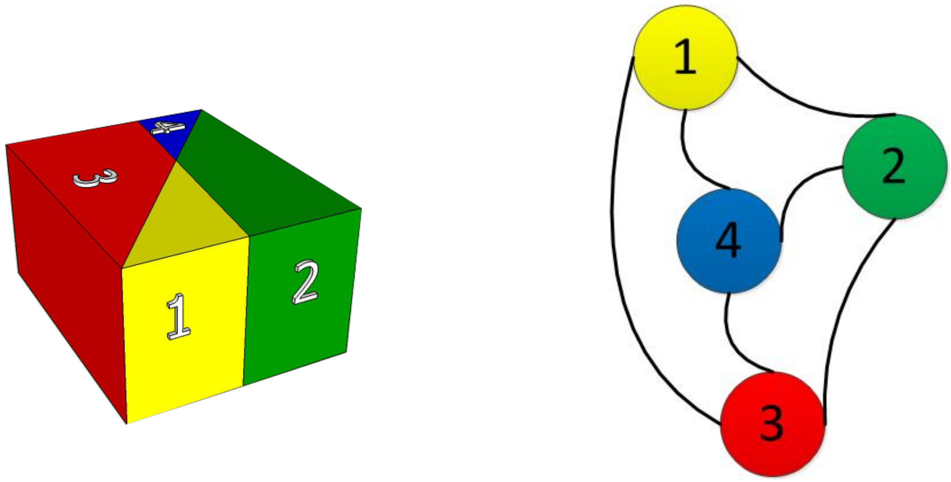
There are two scenarios that could occurs in terms of video availability. In the first the entire video is available at the beginning. The second possible condition is that only a frame is available at one time. This condition is applied to a video streaming scenario where the user receives a single bit of data as part of a frame on a regular basis at a particular bit rate.

Table 3.2: Region Table

No	**colour**	**R**	**G**	**B**	**area**	**left leaf**	**right leaf**	**parent**
1	Red	255	0	0	3947.5	0	0	6
2	Purple	128	0	128	2447.5	0	0	6
3	yellow	255	255	0	3947.5	0	0	
4	green	0	255	0	2447.5	0	0	
5	blue	0	0	255	2209.8	0	0	
6	mixed Red and Purple	191.5	0	64	6395.1	2	1	

Table 3.3: Region Adjacency Table

No	**node1**	**node2**	**RGB distance**	**Valid**
e12	1	2	180.3	yes
e14	1	4	360.6	yes
e15	1	5	360.6	yes
e25	2	5	180.3	yes
e23	2	3	312.3	yes
e35	3	5	441.7	yes
e34	3	4	255.0	yes
e45	4	5	360.6	yes



(a) Segmented Video in Volumetric plot (b) Corresponding VAG

Figure 3.8: Illustration of Video Segmentation and the Corresponding Volume Adjacency Graph

An illustration of the first condition can be seen in Figure 3.8. The input of the segmentation algorithm is considered go be the three-dimensional matrix with vertical(y), horizontal (x) and temporal (t) axis. Each piece of pictorial information lies in three-dimensional space, and is called a voxel. The segmentation produces a number of super voxels which have members across frames. The vertex represents a number of coherence regions across frames, while the edge represents the relationship between supervoxels.

3.2.2.3 Spatio Temporal Region Adjacency Graph

In the streaming situation, representing the entire video as a three-dimensional matrix is no longer possible. Every single frame needs to be segmented individually and the relationship among regions across frames has to be calculated. The regional neighbour-

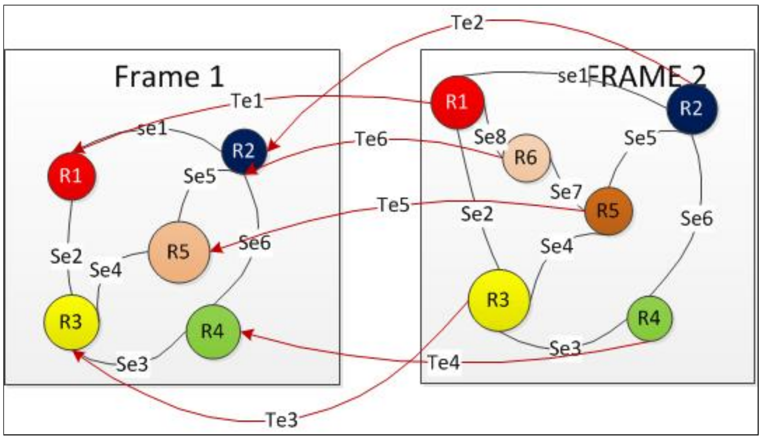


Figure 3.9: Spatio Temporal Edge in Streaming Scenario

hood for every individual frame is recorded in a spatial adjacency graph as illustrated in Table 3.3. Instead of spatial edges, a temporal edge is defined in order to record the region coherency across frames. An illustration of that condition can be seen in Figure 3.9

As can be seen in Figure 3.9, there is a new data structure-temporal edge (abbreviated by Te). Temporal edges connect a region in the previous frame to its pair in the current frame; with the relation potentially being be many-to-many. The data structure of a temporal edge is defined below.

```
class temporalEdge{
    int TemporalEdgeId;
    int CurrentFrameNo;
    int nodeOriginId;
    int nodeCurrentId;
    double Similarity;
    double areaDistance;};
```

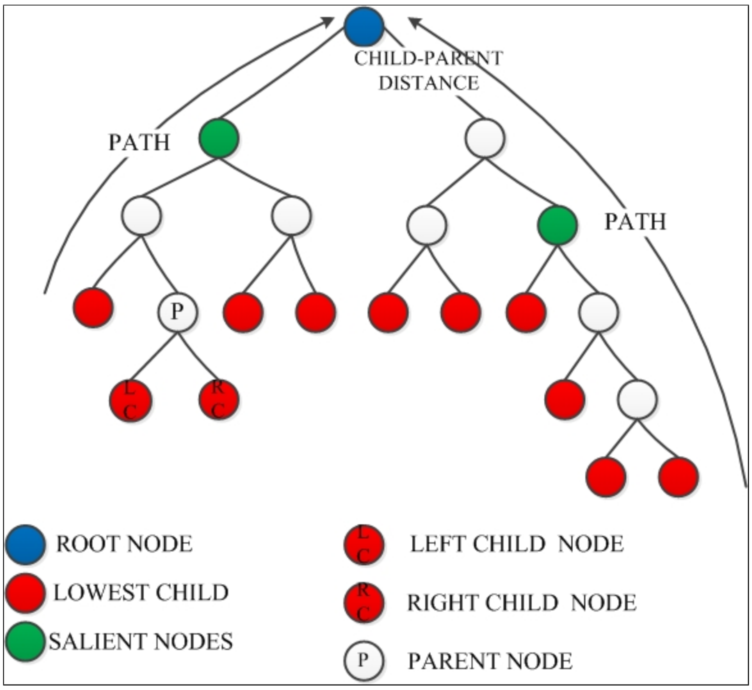


Figure 3.10: Ilustration of Binary Partition Tree and Definition

3.2.3 Binary Partition Tree (BPT)

A BPT is a tree structure where each node other than the lowest child has $left\ child$ and $right\ child$. The root of the tree represents the entire original data. The original data can be an image or a video sequence. BPT represents the evolution from the small to large partitions as can be seen in figure 3.2. The evolution begins from an initial partition as a result of the pre-segmentation task. The merging task is then performed sequentially with respect to a particular merging order rule.

Figure 3.10 illustrates a binary partition tree. The red nodes are denoted as a set of lowest child/leaf nodes, representing the initial partitions. The illustration of relationship between a parent and the left and right child nodes is depicted by RC, LC, and P nodes. All nodes on the tree have a right and left child except the lowest child nodes. A parent node is a result of a merging process between a left child and right child. The green nodes illustrate salient nodes, which are identified in evolution analysis. These green nodes are utilized to cut the tree under them in order to get to a simpler BPT. The blue node at the top of the BPT is a root node. It represents the entire image/video as it is a result of merging all the nodes in the whole tree. Path is defined as all possible tracks to achieve the root node from every lowest child node. Child-parent distance is denoted by the line connecting the parent to the child node. This distance is utilized in evolution analysis to identify the salient node (i.e green nodes). A node represents a region/superpixel in a BPT for a single frame/image. In a BPT for video, a node represents a volume/supervoxel which exists in a number of frames.

The merging order is designed to control which pairwise neighbouring partition needs to be merged to allow the most similar pair of neighbouring nodes to get higher priority. The most similar is represented by the lowest value of edge in the adjacency graph.

3.2.3.1 Similarity Measure and Merging Order

Similarity measure is a value that quantifies the similarity between a pair of neighbouring partitions. In data grouping theory, similarity measure is the opposite to distance; therefore, the greater the distance; the smaller the similarity value [119]. Let us consider a pair of points a in coordinates $(u_{a1}, u_{a2}, ..., u_{an})$ and b in $(u_{b1}, u_{b2}, ..., u_{bn})$. The distance between them can be calculated using Euclidean or absolute distance.

$$\delta_e(a, b) = \sqrt{(u_{b1} - u_{a1})^2 + (u_{b2} - u_{a2})^2 + ... + (u_{bn} - u_{an})^2} \tag{3.10}$$

A shorter formula of Euclidean distance can be seen in 3.11.

$$\delta_e(a,b) = \sqrt{\sum_{k=1}^{n} (u_{bk} - u_{ak})^2} = \left\|\vec{u}_a - \vec{u}_b\right\|_2 \tag{3.11}$$

Where $\delta_e(a,b)$ is the Euclidean distance between a and b.

In absolute distance $\delta_e(a,b)$, the square and square root is replaced by an absolute operation 3.12.

$$\delta_a(a,b) = \sum_{k=1}^{n} \left|(u_{bk} - u_{ak})\right| \tag{3.12}$$

$$\delta_a(a,b) = \left|(\vec{u}_b - \vec{u}_a)\right| \tag{3.13}$$

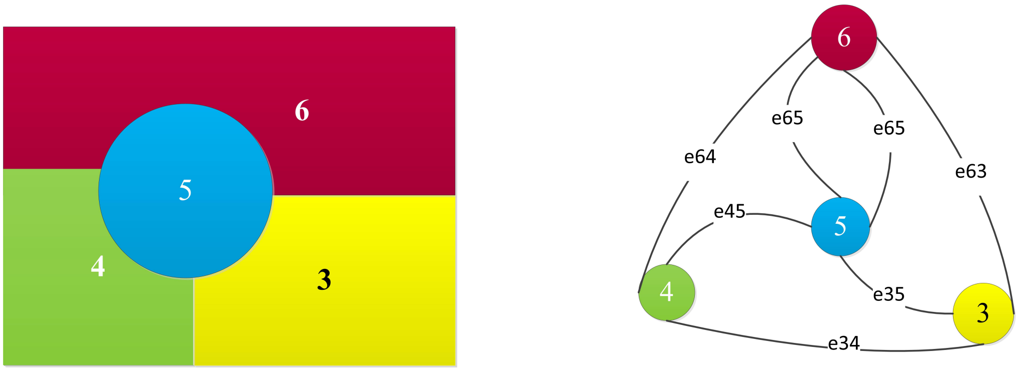
There are many possible parameters to define distance measures, such as colour mean, histogram, combination colour mean and area. As reported previously, colour model also has a great impact on the result. LAB colour space is reported to be close to human visual perception and produces the best result [120]. In three-dimensional video signals, temporal features, which are motion direction and speed, are considered in the similarity measure.

Merging strategy plays an important role in achieving a good segmentation result. The main aim of the merging strategy is to allow the most similar pairs of partitions to get the highest merging priority in every iteration. Merging order is controlled by the similarity measure. For example, the similarity in Table 3.3 is dynamically updated in every single iteration. Merging between two regions affects the entire region adjacency table because the edges of all regions connected to the selected regions have to be updated. The processed edge has to be deleted from the list of edges. The smallest distance for the next iteration has to be calculated among all the remaining edges.

An illustration of the original image and region adjacency graph after the first merging can be seen in Figure 3.11. The region adjacency graph in Table 3.3 is updated as a consequence of updating, deleting and the new minimum distance after merging. The region adjacency table after merging is updated and can be seen in Table 3.4. After the first merging iterations, where e12 is selected as a processed edge, regions 1 and 2 are merged and a new region number (6) is generated. Every edge between any nodes connected to 1 or 2 therefore has to be set as invalid because regions 1 and 2 no longer exist. Instead of being connected to 1 and 2, they are now connected to the region 6; therefore, new edges have to be issued between 4,5 and 3 to 6, meaning the new distance between them must be calculated. The remaining valid edge, highlighted in green, is also shown in Table 3.4.

3.3 Simplification

Generally, the pre-segmentation algorithm produces an over-segmented result. In fact, all evaluated segmentation algorithms give more than expected partitions in the available ground truth. The pre-segmentation task produces too small partitions compared to objects in the real world, meaning that need to be merged in order to get closer to the real-world objects. Excessive merging, however, result in under-segmentation. This occurs when partitions belonging to different objects join in one segment.



(a) Segmented Image

(b) Corresponding RAG

Figure 3.11: Illustration of Image Segmentation and Corresponding Region Adjacency Graph after the first merging iteration

Binary partition Trees (BPT) created from the initial segmentation are cluttered with thousands of nodes. Every node corresponds to a partition in the image or video. Among those nodes, although some are meaningful and closely related to the real-world object or ground truth, the majority of them are not. A simplification of the BPT is therefore important if a simpler tree need to be achieved.

Simplification of BPT is an activity to prune the branches from the tree in order to cut out unnecessary small partitions. This result in a simpler tree composed of nodes representing greater partitions and expected to be closer to the ground truth. Identifying the node on the tree where the branch should be cut is a critical decision.

The merging event between a pair of partitions can be categorized as 'normal' and 'critical' merging. Normal merging is defined as merging between pairwise partitions with small feature distances. This usually happens between two different partitions belonging to a single object. Critical merging, meanwhile, is defined as merging between

Table 3.4: Region adjacency table after first iteration merging, corresponding to figure 3.11

No	node1	node2	RGB distance	valid
e12	1	2	180.3	not
e14	1	4	360.6	not
e15	1	5	360.6	not
e25	2	5	180.3	not
e23	2	3	312.3	not
e35	3	5	441.7	yes
e34	3	4	255.0	yes
e45	4	5	360.6	not
e64	6	4	325.3	YES
e65	6	5	270.5	YES
e63	6	3	270.5	YES

two neighbouring partitions with high feature distance. This occurs when two partitions belonging to different object start to merge. Critical merging has to be avoided in order to minimize the under-segmentation error.

An evolution analysis is carried out along the branch of the tree in order to observe the merging history from the small initial partitions to the root of the tree. Critical merging is expected to be identified during the partition's evolution. The identified critical merging is classified as a pruning node candidate where the branch of the tree would be cut. An evaluation of this technique is demonstrated in Chapters 4 and 6.

The initial segmentation produces vast numbers (around thousands) of small partitions. If the number of initial partitions is n there will be n number of possible paths from the leaf to the root. Paths can be defined as $P = \{P_1, P_2, ..., P_n\}$, and each path has a collection of nodes from the lowest leaf towards the root. Every individual path is defined as $P_i = \{nd_1, nd_2, ..., nd_l\}$ where l is the number of nodes along the path from the lowest leaf to the root, l can vary for each path. Evolution of a particular P_i is defined as:

$$\gamma(k) = M(nd_k) \;\; k \in \{1...l\} \tag{3.14}$$

Where $M(nd_k)$ is a model of node nd_k that has a number of feature vectors. For example, in single frame implementation it will only consider the colour and size of the partition, while in the supervoxel BPT, it will consider colour, size, centroid direction and speed. In order to identify the critical merges, a mathematical tool has been proposed in [7]. A modified first and second derivative is employed.

$$\gamma'(k) = \mid \gamma(k) - \gamma(k-1) \mid \;\; k \in \{2...l\} \tag{3.15}$$

$$\gamma''(k) = \gamma(k-1) + \gamma(k+1) - 2\gamma(k) \;\; k \in \{2..l-1\} \tag{3.16}$$

A modified second derivative is employed to identify the peak. Many peaks may be small enough, however, to indicate that a critical merging has occurred and, therefore, a selection rule has to be imposed: for example, by identifying only the highest peak, or identifying the peak above a specific threshold. The simplification algorithm is applied in Chapters 4 and 6 with different features and peak selection rules.

3.4 Video Content Metadata

3.4.1 Metadata Structure

Metadata is designed to be stored in an integrated database structure. The structure is illustrated in Figure 3.12.

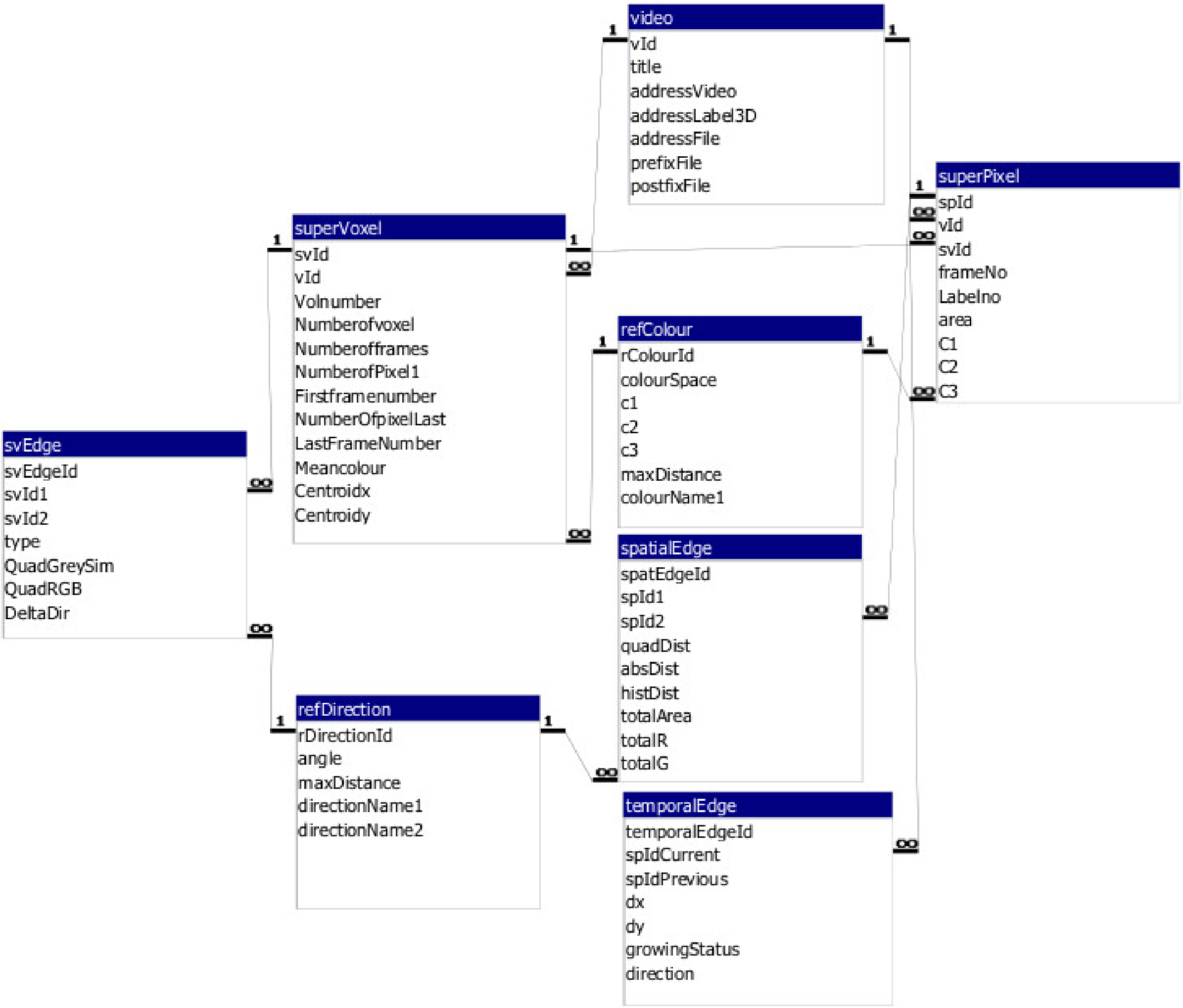


Figure 3.12: Main Database Structure

A $video$ table is designed to record the video identity, address where the video is stored and the initial label map. The $superVoxel$ table accommodates the super voxel in three-dimensional approaches. This will be discussed in Chapter 6. The supervoxel neighbourhood is recorded in the $svEdge$ table. The frames-based approaches that will be discussed in Chapter 4 are recorded in $superPixel$ and $spatialEdge$ tables. The temporal neighbourhood will be extracted in Chapter 5 and will be recorded in the $temporalEdge$ table. In order to allow the system to provide human-like textual data, reference tables for direction ($refDirection$) and colour ($refColour$)are prepared as supporting tables.

3.4.2 Descriptor

The region is the smallest unit that will be managed in the metadata. In order to allow information querying of the content, the descriptor should represent the region while at the same time being understandable to the user. For example, instead of recording the colour as (255, 255, 255), it is more understandable to represent it with the word 'black', since people prefer the colour name rather than an RGB code.

As mentioned before, we use partition as a generic term for region/superpixel and volume/supervoxel. The actual meaningful term for the user in visual space, however, is a region. The recorded descriptor in this thesis is in the region format. Even though in some implementation, a three-dimensional (volume) is obtained, the translation to the region would be projected to the two-dimensional space in a particular frame.

Partition attributes such as size, mean colour and centroid are recorded as main features, while the relationship with its neighbour such as relative position, distance to border, neighbouring colour distance are prepared. In regard to the inter-frame relationship, motion information consists of direction and speed for every region. In addition, region size variation across frames can be prepared to indicate whether a region is growing or shrinking.

3.5 Evaluation

3.5.1 Testing Video and Available Ground Truth

Segmentation results are evaluated by their capability to get near to the human-made ground truth. Even though the quality of ground truth significantly affects the evaluation, the reliable methods still rely on that. The main issues in measuring the segmentation quality are twofold, the evaluation method and the quality of the ground truth. There are some available ground truths such as PASCAL, BSD data set for image segmentation and the ‘xiph.org’ data set for video multiple objects; the SEGTRACK data set for video single object ground truths. In this thesis, the ‘xiph.org’ multiple objects ground truth is selected in order to measure the quality of segmentation. It is selected because the objective of this algorithm is to segment the video into composed objects, no matter how many they are.

The details of the eights tested video clips can be seen in Table 3.5. Throughout the thesis, the name of the video test refers to the list of in the Table 3.5. Due to the public availability of the video tests, a number of researchers in this area, such as [6], [121], have also used the same video test.

Table 3.5: The Description of Testing Sequence

No	Sequence	size	no frame	Description	Camera motion	Object Motion
1	Bus	352 x 288	84	The camera moves to the left with the bus. The cars and fences between the camera and the bus appears to be moving right due to their slower (relative to the bus) movement.	Nearly linear motion.	The bus, car are moving
2	Container	352 x 288	85	The camera remains static while the container and the small boat moves to the right of the scene.	No camera motion	The near linear motion of the container and the small boat from left to right.

Continued on next page

Table 3.5 – *Continued from previous page*

No	**Sequence**	**size**	**no frame**	**Description**	**Camera motion**	**Object Motion**
3	Garden	352 x 240	80	The camera move from right to left while the objects remain static. As a result of the different distance to the camera, the foreground (tree) moves faster than the background (flowers and houses).	Nearly linear camera motion.	No object motion
4	Ice	352 x 288	79	The camera remains static, while the near dozen people in the scene performs articulated motion and complex occlusion.	No camera motion	The skiers perform articulated motion in the scene.

Continued on next page

Table 3.5 – *Continued from previous page*

No	**Sequence**	**size**	**no frame**	**Description**	**Camera motion**	**Object Motion**
5	Paris	352 x 288	79	The camera Remain Static, while the object move in different direction and speed	No Camera Motion	The man and woman perform motion, while the ball move in different speed
6	Salesman	176 x 144	80	The camera remain static, while the object moves	No Camera Motion	The man perform motion

Continued on next page

Table 3.5 – *Continued from previous page*

No	Sequence	size	no frame	Description	Camera motion	Object Motion
7	Soccer	352 x 288	69	The camera moves right with the man kicking the soccer ball.	Nearly linear camera motion.	The three soccer players move with the camera to the right, while the woman and the dog move slowly to the right in the background.
8	Stefan	352 x 240	75	The camera moves to the right with the tennis player, than to the left.	Articulated camera motion	Yes, the tennis player moves to the right, hits the ball, then moves to the left.

There are eight video sequences, and in total 631 frames. The ground truth for each frame is available. Examples of frames and their ground truth can be seen in figure 3.13. The evaluation refers to the test video clips except when stated otherwise.

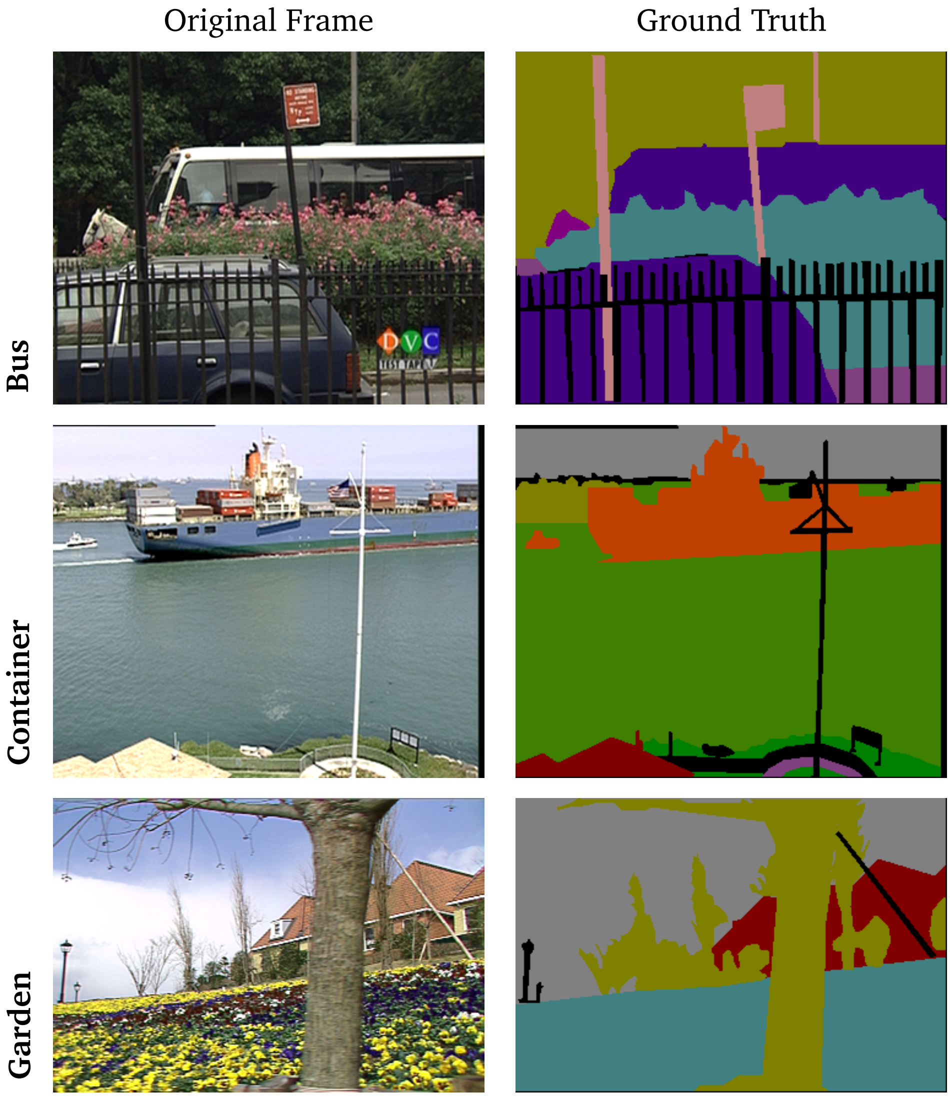


Figure 3.13: Example Frame and Its Corresponding Ground Truth

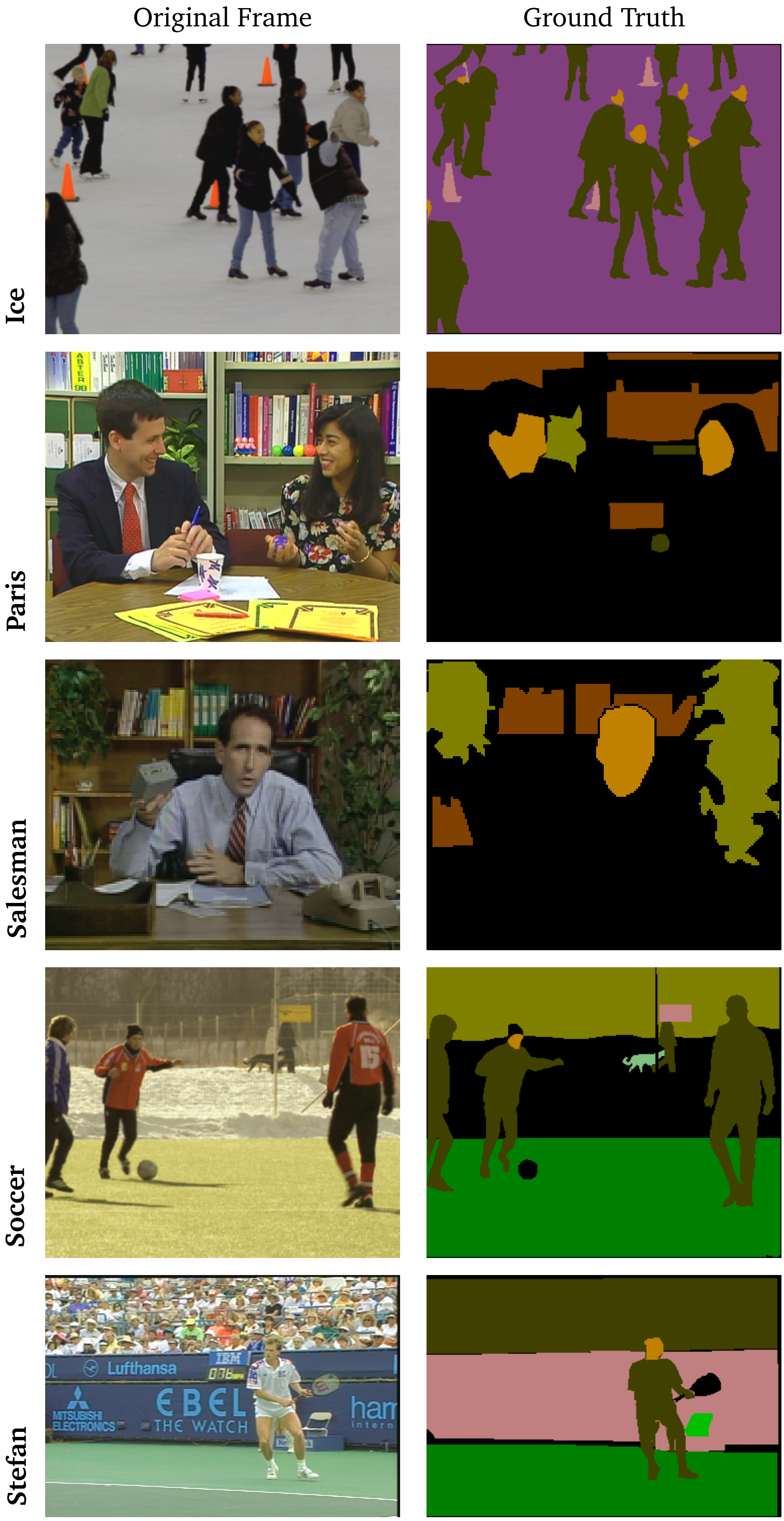


Figure 3.14: Example Frame and Its Corresponding Ground Truth - Continued

3.5.2 Evaluation Method

There are some concerns in evaluating the outcome of segmentation as to the preliminary step of video metadata. In much of the literature, the most common parameters are boundary recall, precision [122] and under-segmentation error [123]. This thesis adopted a simple over-segmentation rate as an additional measurement. This measurement is important in respect to the simplification task. Boundary recall measures how precisely the boundary of the super pixel is directly consistent with the boundary of the ground truth. Under-segmentation, on the other hand, measures how many different objects in the ground truth are melted into one partition. Under-segmentation error is indicated by the portion of boundary that exists in the ground truth, but is unavailable in the segmentation result. The over-segmentation rate, meanwhile, measures how many partitions are needed to form an object in the ground truth. The over-segmentation rate is expected to be one, if one ground truth object is exactly paired with one partition. Smaller over-segmentation rates mean that every partition occupies a small part of the ground truth.

Assuming that there are multiple objects of ground truth $G = \{g_1, g_2, ..., g_n\}$ and the results of segmentation of the particular frames are $S = \{s_1, s_2, ..., s_m\}$ where $m \neq n$. Let p_i be a member of P as all boundary pixels of the ground truth and q_i be a member of P as all boundary pixels of the segmentation results.

The boundary recall is calculated by comparing the boundary of the ground truth against the boundary of partitions. A true positive is when all pixels in a partition border meet the expected ground truth periphery. According to [124] the confusion matrix can be seen in Table 3.6.

Table 3.6: Confusion Matrix

	actual prositive	actual negative
predicted positive	TP	FP
predicted negative	FN	TN

Let us define $Sb = s_1, s_2, ...s_p$ as a set of boundaries in the segmentation result, while $Gb = g_1, g_2...g_q$ is a set of boundaries in the ground truth. A true positive is a number of Sb that meets one of the members of Gb. Most of the time, the machine-segmented data has more partitions compared to the expected ground truth. Therefore, p is greater than q. False Negative (FN) is a quantity of the boundary in the ground truth Gb that does not exist in the segmentation boundary Sb. The precision measure is the amount of accurate boundaries compared to all boundaries in the segmentation result. The precise boundary is a member of true positive, while the remaining segment boundaries are false negative (FN). Recall and precision are calculated from the equations 3.17 and 3.18 respectively:

$$recall = \frac{TP}{TP + FN} \tag{3.17}$$

$$precision = \frac{TP}{TP + FP} \tag{3.18}$$

The ideal value of recall is one, which can be achieved if all the ground truth boundaries are discovered in the segmentation boundaries. Precision aims to quantify the amount of noise in the segmentation result. It is indicated by the portion of the boundaries that exists in the segmentation result that would not be expected from the ground truth.

Under-segmentation error aims to measure the amount of partition floods in the ground truth boundaries. Following the formulation in [123] under-segmentation error is defined in equation 3.19.

$$undersegmentation = \frac{1}{N}\left[\sum_{S \in Gt}\left(\sum_{P:P \cap S \neq \emptyset} \min(P_{in}, P_{out})\right)\right] \tag{3.19}$$

Where Gt is ground truth regions, S = regions as a segmentation result, N is the total number of pixels in the entire image, P_{in} and P_{out} are the parts of segment S inside and outside of the ground truth respectively. Over-segmentation rate is simply calculated by comparing the quantity of the expected ground truth and the available segmentation result as in equation 3.20.

$$oversegmentation = \frac{n}{m} \tag{3.20}$$

Where n and m are quantity of ground truth and segment respectively.

The evaluation is performed in order to assess the reliability of the segmentation and simplification result for the subsequent task. In Chapter 4 where segmentation is performed for a single frame, the evaluation is conducted by comparing the frame ground truth and the label map. While in Chapter 6, the evaluation is performed iteratively across frames and the recall, precision, under-segmentation error and over-segmentation rates are averaged.

The query answer is also evaluated using those four parameters. Since the answer to the query can vary for each request depending on the query condition, the ground truth cannot directly be a measure of the quality of the query results. Therefore, it would set a number of typical queries in which the result is known from the ground truth in order to perform the test.

Chapter 4

Single Frame Segmentation

4.1 Introduction

This chapter is dedicated to discussing various methods of creating hierarchical partition trees for a single image. Firstly, three different pre-segmentation methods will be performed, which are watershed, mean shift and SLIC. Secondly, similarity measures will be implemented in various ways, namely colour, histograms, combined with area.

The input into the algorithm that will be discussed throughout this chapter is a single frame/image in two-dimensional space, meaning that the discussion is confined to the pre-segmentation and the merging tasks in respect to the image. The partition result from the image is defined as a superpixel or region.

A comparison of all the methods will be presented in order to justify the choice of the most suitable option to provide reasonable video metadata. The result at the end of this chapter will be able to drive the choice of method in the next stage of our work.

4.2 Similarity Measure

A single variable will be implemented to control the merging order and decision. In case there is more than one feature, a computation will be carried out to yield a scalar value. Based on its similarity value, the most homogeneous pair of partitions get the highest merging priority. Similarity is considered to be the inverse of the distance between a pair of regions.

Colour images are composed of a number of pixels and are comprised of three colours components. For example, in RGB colour space, every pixel comprises of R, G and B elements while in CIE l*a*b space, it comprises of L (luminance), and a,b colour components. In order to keep neutral to colour space, let us define a colour model c with c_1, c_2 and c_3 as colour components of each pixel $\vec{c} = \{c1, c2, c3\}$. There are two options to calculate the difference, either quadratic (equation 4.1) or absolute distance (equation 4.2).

4.2.1 Mean Colour Euclidean Distance

Consider that R_i and R_j are a pair of neighbouring regions. The feature in a region consists of three mean colour elements, and region size can be expressed as $R_i = (\overline{\vec{c}_{Ri1}}, a_{Ri})$ and $R_j = (\overline{\vec{c}_{Rj1}}, a_{Rj})$. Euclidean colour distances between R_i and R_j is calculated referring to equation of 3.11, yield equation 4.1.

$$\delta_{e\vec{c}}(R_i, R_j) = \left\| (\overline{\vec{c}_{R_i}} - \overline{\vec{c}_{R_j}}) \right\|_2 \tag{4.1}$$

Where $\delta_{e\vec{c}}(R_i, R_j)$ is the colour Euclidean distance between the region i and j. $\overline{\vec{c}_{Ri1}}$ denotes the average of colour components, $\vec{c}$ denotes the colour vector consisting of three components c_1, c_2 and c_3 of region i, $\|\|_2$ denotes Euclidean distance. Considering the mean colour distance without taking into account the area will result in small regions around a big region. In order to minimize that issue, according to [58], the impact of region size a_{R_i}, a_{R_j} on the final similarity is calculated using 4.3. Consider R_l is the new region formed when R_i and R_j are merged.

$$\overline{\vec{c}_{R_l}} = \frac{a_{R_i} * \overline{\vec{c}_{R_i}} + a_{R_j} * \overline{\vec{c}_{R_j}}}{a_{R_i} + a_{R_j}} \tag{4.2}$$

a denotes region size. $\overline{\vec{c}_{R_l}}$ is the accumulation colour mean of regions i and j. By using $\overline{\vec{c}_{R_l}}$ from equation 4.2, the colour distance between the original regions i and j to new region l can be calculated using equation 4.1. The distance between R_i and R_j, considering colour and area can be calculated using 4.3.

$$\delta_{es\vec{c}}(R_i, R_j) = \frac{\Big(a_{R_i} * \delta_{e\vec{c}}(R_i, R_l)\Big) + \Big(a_{R_j} * \delta_{e\vec{c}}(R_j, R_l)\Big)}{(a_{R_i} + a_{R_j})} \tag{4.3}$$

$\delta_{es\vec{c}}(R_i, R_j)$ is proportional Euclidean distance measure considering colour and size.

4.2.2 Mean Colour Absolute Distance

Mean colour absolute distance is an alternative similarity measure if square and square-root operations need to be avoided. $\delta_{a\vec{c}}(R_i, R_j)$ can be calculated by 4.4.

$$\delta_{a\vec{c}}(R_i, R_j) = \left|\overline{\vec{c}_{Ri}} - \overline{\vec{c}_{Rj}}\right| \tag{4.4}$$

Proportional distance measure $\delta_{as\vec{c}}(R_i, R_j)$ considering colour and size can be calculated using equation 4.2 and 4.3 by substituting $\delta a\vec{c}(R_i, R_j)$ from equation 4.4

$$\delta_{as\vec{c}}(R_i, R_j) = \frac{a_{R_i} * \delta_{a\vec{c}}(R_i, R_l) + a_{R_j} * \delta_{a\vec{c}}(R_j, R_l)}{a_{R_i} + a_{R_j}} \tag{4.5}$$

Where R_i and R_j are neighbouring regions, and R_l is a new region formed if R_i and R_j are merged, $\vec{c}_{R_l}$ is calculated using 4.2.

4.2.3 Histogram Distance

Histogram distance is obtained from a colour quantified of the input image. The same method has been implemented in [72], in interactive image segmentation using maximal similarity. The histogram is computed using quantized colour components. Every colour component is quantified into 16 levels (bins), meaning that the total quantified colour is equal to 16 x 16 x 16 = 4096 levels. After the levels are determined, natural images in tricolour image (which can be any colour space such as RGB, cie L*A*B) are transformed into a quantized colour image. A histogram is calculated according to the transformed image.

The Bhattacharyya coefficient is calculated to measure the similarity between two regions. The coefficient value is normalized between zero to one. Zero indicates that the regions are totally dissimilar, while one indicates that they are for identical.

$$\delta_h(R_i, R_j) = \sum_{u=1}^{l} \sqrt{Hist^u_{R_i}.Hist^u_{R_j}} \tag{4.6}$$

Where, $\delta_h(R_i, R_j)$ is the histogram distance between region i and j, l is the number of levels (bins), $Hist^u_{R_i}$, $Hist^u_{R_j}$ is the value of the histogram for level (bin) u of region R_i and R_j respectively.

4.2.4 Merging

The merging decision can be taken according to particular criteria as previously discussed in subsections 4.2.1, 4.2.2, and 4.2.3. The iteration is controlled by the similarity measures. In preparing BPT, merging iterations will be performed until the root node is achieved (i.e when no more valid edges remain in the adjacency table). The root node represents the entire image/frame. The number of iterations to achieve the root is equal to $2n - 1$ where n is the quantity of initial superpixels as a result of the pre-segmentation task. When BPT is available, an evolution analysis can be performed by following the transition of a region's attributes from the lowest leaf node to the root.

4.3 Simplification

Salient regions are defined as regions that stand out compared to their surrounding regions. In this work, saliency is identified in the BPT framework. The BPT structure records a region's evolution from the initial small node to the entire root node. It is therefore possible to track the node transformation in every iteration. The simplification is carried out in accordance with the formulation set out in section 3.3.

Evolutionary analysis tracks the region's transformation in every possible path from the lowest child nodes to the root of the BPT. This means that the number of paths is equal to the initial regions. The evolution function aims to seek the peak of child-parent distance, which may possible exist many times during the region's evolution. Equations 3.14, 3.8 and 3.14 are employed to carry out the evolution analysis. The model of nodes in the evolution consists of the colour and size of the regions. Child-parent distance is calculated using proportional Euclidean 4.3 and absolute 4.5 distance formulation.

Due to the structure of BPT, there is a possibility that a node becomes a member of more than one paths, meaning that a number of nodes are probably selected many times during evolution, and a list of non-unique nodes is created. Further analysis has to be performed in order to ensure that the pruning node candidates meet the requirements. The first task is removing the duplicated nodes. The second task is to ensure that no candidate nodes have direct or indirect child-parent relationships each other. If the relationship exists, the lower level (smaller region) is selected. Finally, the size accumulation of all candidates nodes, have to be equal to the original image size. The simplification algorithm uses the pruning candidate list to cut the branch of the tree under the nodes. The simplification result is demonstrated and assessed against the available ground truth in the evaluation section.

4.4 Evaluation

The evaluation is carried out in regard to the general framework set out in Chapter 3, evaluation section. The evaluation is performed on the original pre-segmentation and the simplification result. Pre-segmentation is performed using three different algorithms. The partitions obtained by the pre-segmentation are employed as initial conditions of the region merging task. The iteration needed to achieve the final root node depends on the quantity of original regions. Table 4.1 shows the boundary recall, over-segmentation rate and under-segmentation error among those three algorithms and is tested using the first frame of the testing videos.

4.4.1 Pre-Segmentation

The initial regions obtained from pre-segmentation task is evaluated against the ground truth. Table 4.1 presents the pre-segmentation results. The boundary recall shows the extent to which the number of boundary pixels in the pre-segmentation result corresponds to the boundary of the ground truth. All algorithms show good results, between 0.88 to 0.92 on average. The result of watershed produces the highest boundary recall, at 0.92 on average, followed by SLIC and mean shift at 0.91 and 0.88 respectively.

The second parameter is under-segmentation error. This error occurs when a partition occupies more than one ground truth area. The values of under-segmentation error are expected to be low (close to zero). According to the experiments, the values of under-segmentation at this stage are around 0.08 to 0.13 whilst still good enough. Watershed produces the best under-segmentation error at 0.08 while the two other algorithms re-

Table 4.1: Pre-Segmentation Boundary Recall, Under-Segmentation Error and Over-Segmentation Rate

No	Test Sequence	Boundary Recall			UnderSeg Error			Over-Segmentation Rate		
		WS	MS	SLIC	WS	MS	SLIC	WS	MS	SLIC
1	bus	0.91	0.98	0.81	0.16	0.20	0.31	0.0082	0.0383	0.0775
2	container	0.89	0.85	0.90	0.09	0.08	0.12	0.0062	0.0790	0.0587
3	garden	0.93	0.78	0.88	0.09	0.15	0.15	0.0044	0.0206	0.0393
4	ice	0.96	0.92	0.97	0.07	0.09	0.09	0.0061	0.1912	0.0526
5	paris	0.88	0.81	0.81	0.05	0.08	0.11	0.0022	0.0116	0.0194
6	salesman	0.93	0.87	0.98	0.08	0.11	0.09	0.0057	0.0290	0.0139
7	soccer	0.94	0.89	0.95	0.06	0.07	0.08	0.0044	0.0830	0.0417
8	stefan	0.93	0.96	0.97	0.06	0.05	0.08	0.0041	0.0197	0.0313
Average		0.92	0.88	0.91	0.08	0.11	0.13	0.0052	0.0591	0.0418

turn an error of 0.11 to 0.13.

The next parameter is over-segmentation rate, which shows a comparison between the expected and the actual number of partitions. In the Bus sequence, for example, the ground truth consists of 39 areas and the number of partition obtained by watershed is 4742. Initially, therefore, every ground truth area is fragmented into 122 partitions. The over-segmentation rate is calculated by dividing the expected ground truth by the actual of the segmentation results. As a result, the value of over-segmentation rate for watershed on bus sequence is very low at 0.0082. In contrast, SLIC only has 503 and mean shift 1019 partitions for the same frame. In terms of over-segmentation rate, mean shift gives best performance with an average of 0.059, followed by SLIC at 0.042.

SLIC produces an almost stable number of partitions around 500 in every test frame. This is independent of the number of pixels and the complexity of the frame. This is due to K setting in this algorithm. SLIC is an extension of K-means, and K is set to 500. The Watershed and mean shift, meanwhile, work without any parameter settings, but purely based on the image input. The number of partitions in those algorithms relies on the complexity of the frame.

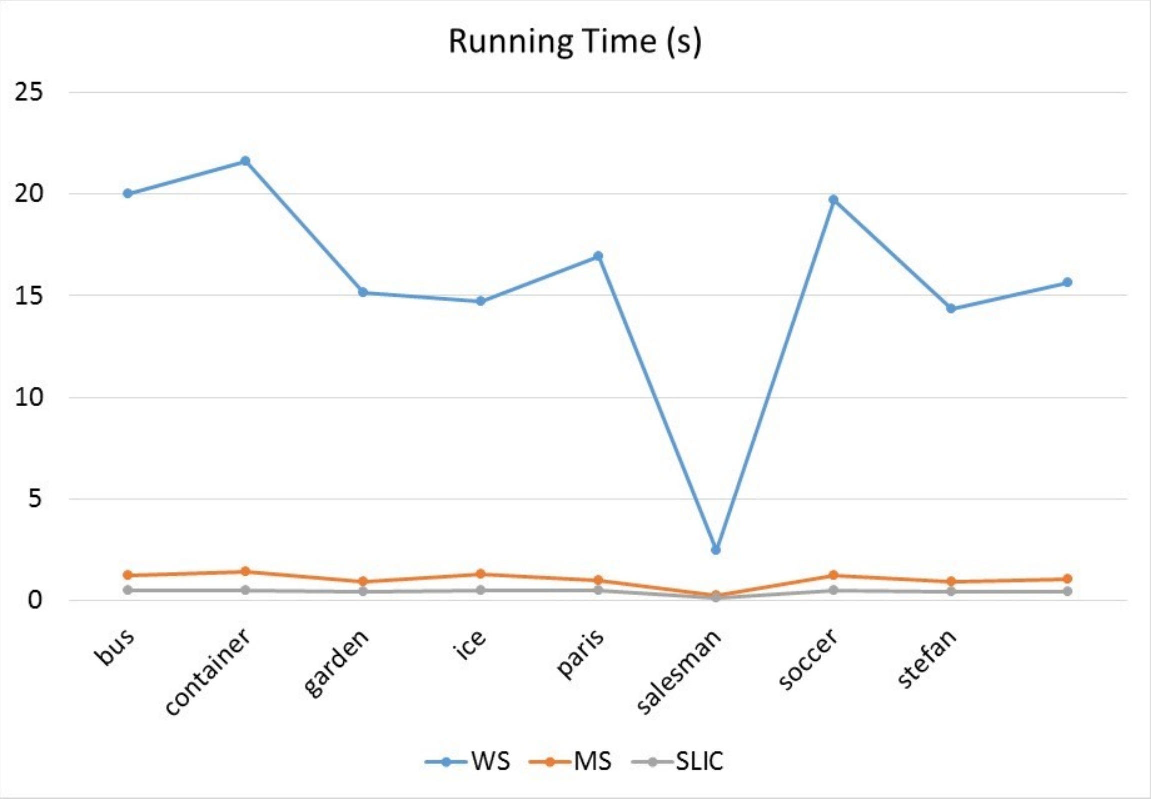


Figure 4.1: Running time comparison of the Watershed, Meanshift and SLIC

Figure 4.1 describes the running time of pre-segmentation algorithms. In pre-segmentation task, watershed gives the worst performance in terms of running time. SLIC consistently outperforms to compare to the other two mean shift and watershed. The watershed take longer because it is including a process to reduce the over-segmentation rate by deleting small regions.

4.4.2 Merging and Simplification

A merging task was carried out based on the pre-segmentation result. The number of partitions affects the BPT creation and the simplification of the multi-scale segmentation. There are nine models, which will be evaluated based on the pre-segmentation algorithm and the similarity criteria. The models are:

- Euclidean Mean Colour

    Watershed

    Mean shift

    SLIC

- Absolute Mean Colour

    Watershed

    Mean shift

    SLIC

- Histogram Distance

    Watershed

    Mean shift

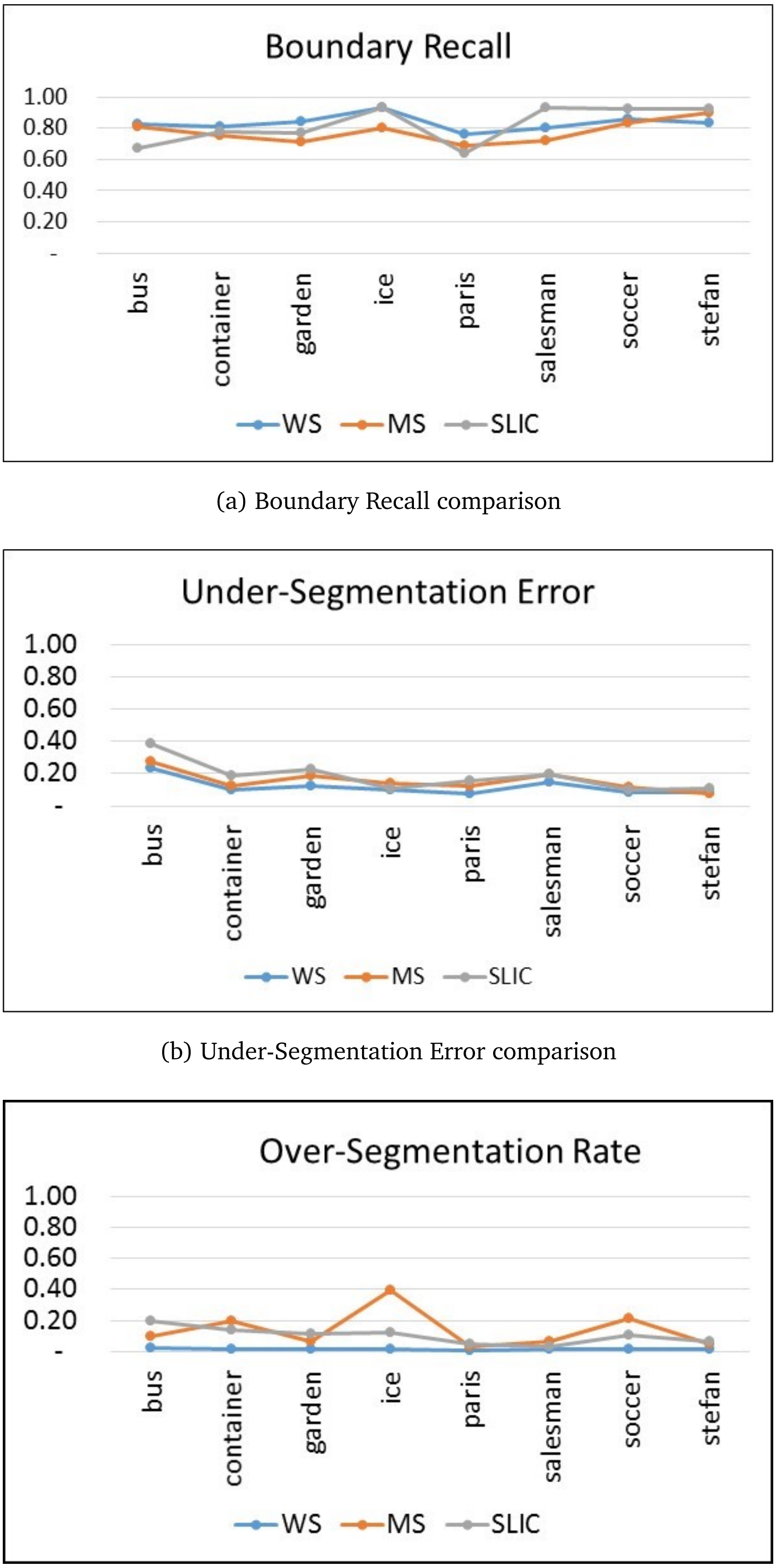
    SLIC

The next sub section describes the evaluation results of the simplification against the available ground truth for all the test video clips.

4.4.2.1 Euclidean Colour Mean

Euclidean colour mean is computed with equation 4.3, The boundary recall of the simplified tree is presented in Figure 4.2(a). This gives an indication that the simplified BPT of watershed slightly outperforms those of SLIC and mean shift. In some video with complex motions such as, the Soccer and the Ice videos, SLIC produces better results. The comparison of over-segmentation rate in Figure 4.2(c) shows that the simplified tree produced by mean shift pre-segmentation gives a better result than the other two. Running time is compared in Table 4.2, and it can be seen that generally BPT preparation and simplification on the tree produced by watershed take much longer compared to mean shift and SLIC. An interesting fact is that SLIC consistently takes almost a uniform time to perform the task in every sample test frame. This can be explained by the fact that SLIC produces a stable number of partitions corresponding to the K value.

In order to get a simpler version of the partition, a simplification task is carried out. Simplification is performed by observing the merging history and look for a critical merging. After simplification, some of the information might be lost because of excessive merging. This occurs because of low contrast neighbouring regions are already merging and critical merging is not identified, even though they belong to different semantic objects. The graph in Figure 4.2(a) shows the boundary recall after simplification. Following simplification the boundary recall was slightly less than in the pre-segmentation results. After simplification, the average watershed gives 0.83, while mean shift and SLIC



(a) Boundary Recall comparison

(b) Under-Segmentation Error comparison

(c) Over-Segmentation Rate comparison

Figure 4.2: Evaluation of Simplification Results on a Euclidean Distance Measure

Table 4.2: The Running Time of BPT Preparation, Simplification of All Algorithms Using Euclidean Similarity

No	Video Test	WS		MEAN SHIFT		SLIC	
		BPT	SIMPLIF	BPT	SIMPLIF	BPT	SIMPLIF
1	bus	506.00	3.36	26.33	0.39	6.95	0.26
2	container	497.18	3.32	3.91	0.13	6.52	0.13
3	garden	381.82	2.78	20.12	0.30	5.69	0.12
4	ice	404.34	2.82	0.74	0.03	6.70	0.14
5	paris	454.41	3.46	24.37	0.40	8.15	0.19
6	salesman	30.06	0.43	2.14	0.09	7.43	0.19
7	soccer	524.55	3.60	2.75	0.09	9.06	0.18
8	stefan	344.23	2.59	21.80	0.39	8.79	0.18
	average	392.82	2.79	12.77	0.23	7.41	0.17

are 0.78 and 0.82, respectively. Compared to the pre-segmentation stage, the values are 0.92, 0.88 and 0.91 for watershed, mean shift and SLIC respectively.

The under-segmentation error for all types shows a slight increase from 0.08-0.13 to around 0.12 - 0.18. Under-segmentation increases as a consequence of the merging task, where some of the regions are merged even though they may belong to different ground truth object.

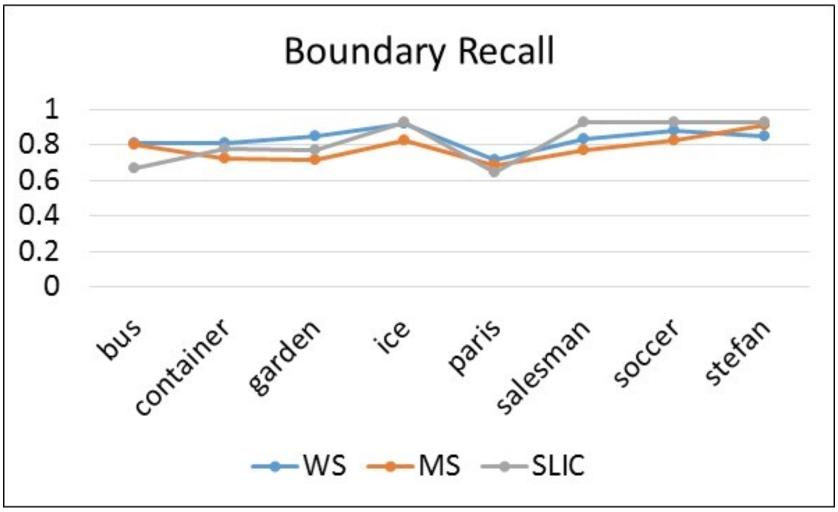
Figure 4.2(c) presents the over-segmentation rate. The best over-segmentation rate is achieved by the simplification result for mean shift; SLIC produces slightly less than mean shift, while watershed produces far lower over-segmentation rate. For the 'ice' sequence, for example, mean shift produces 66 segments compared to 36 expected. This means that every ground truth region is only fragmented into two partitions.

4.4.2.2 Absolute Colour Mean

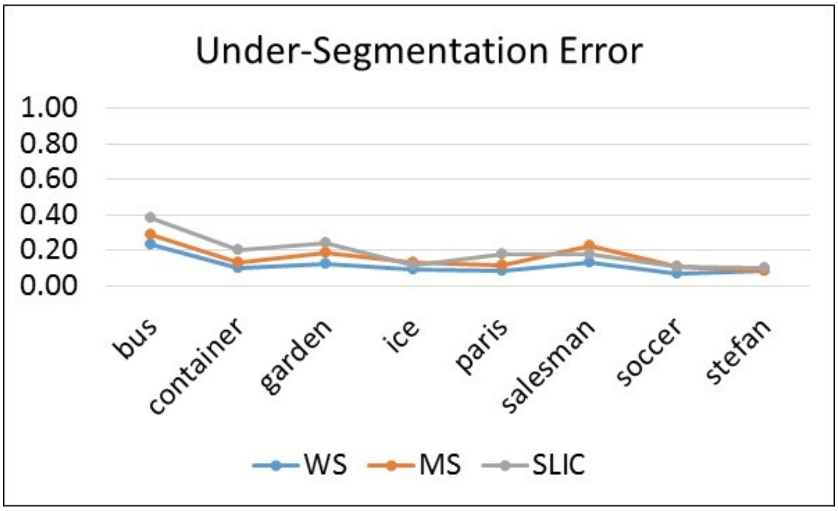
Table 4.3: Running Time of BPT Preparation and Simplification for All Algorithms Using Absolute Colour Mean Similarity

No	Video	WS		MEAN SHIFT		SLIC	
	Test	BPT	SIMPLIF	BPT	SIMPLIF	BPT	SIMPLIF
1	bus	506.02	3.86	35.67	0.54	8.85	0.23
2	container	497.92	3.68	5.69	0.14	8.23	0.17
3	garden	380.81	3.08	28.15	0.43	7.25	0.16
4	ice	404.44	3.07	0.97	0.05	8.53	0.19
5	paris	454.53	3.94	24.72	0.43	9.04	0.19
6	salesman	30.14	0.43	2.17	0.11	7.60	0.18
7	soccer	525.21	3.88	2.94	0.10	9.02	0.18
8	stefan	343.79	2.92	22.26	0.39	8.97	0.19
	average	392.86	3.11	15.32	0.28	8.44	0.19

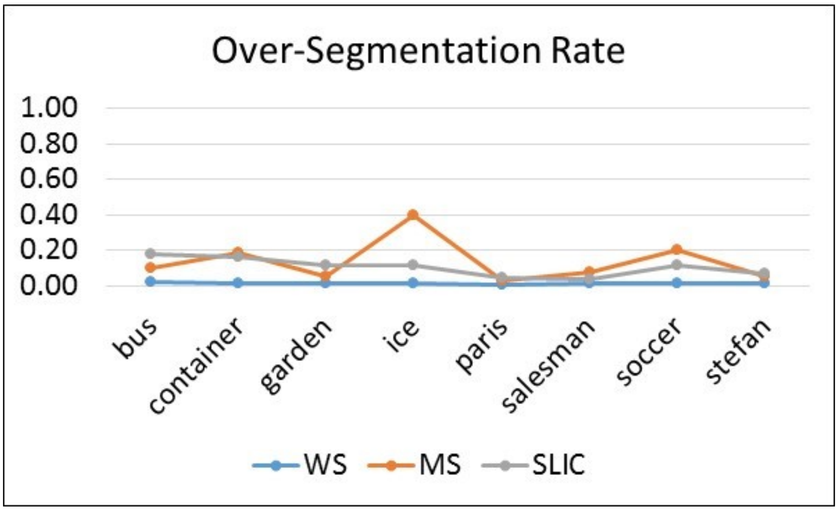
The evaluation results of the simplified tree using the absolute colour mean can be seen in Figure 4.3. It can be seen that the trend is exactly the same as the previous experiment using Euclidean distance. Although on average, the simplification of watershed is slightly better than SLIC, SLIC outperforms in video with complex motion.



(a) Boundary Recall Comparison



(b) Under-Segmentation Error Comparison



(c) Over-Segmentation Rate comparison

Figure 4.3: Evaluation of Simplification Result on Absolute Distance Measure

The best over-segmentation rate is achieved by a simplified tree of BPT with the mean shift algorithm. This parameter indicates how far the ground truth fragmented after simplification. On average, mean shift gives 0.14, meaning that every ground truth object fragmented into seven regions. SLIC gives slightly lower at 0.1 (10 regions per ground truth). In contrast, watershed gives an extremely over-segmented at 0.013 or 76 superpixels per ground truth.

The smallest under-segmentation is achieved by the simplification of BPT on watershed at 0.12 on average. Mean shift and SLIC achieved slightly lower at 0.16 and 0.18 respectivelly.

Running time comparison between evaluated algorithm shows that the BPT creation and simplification of watershed algorithm much slower compare to the remaining two algorithms. In total, it is almost 30 times slower that mean shift, and 60 times slower than SLIC.

4.4.2.3 Histogram Distance

Histogram distance is computed using equation 4.5. The evaluation result of the simplified tree using the histogram colour distance can be seen in Figure 4.4. It can be seen that the trends for under-segmentation error, and over-segmentation rates are similar as those the two previous experiments using Euclidean and absolute distances. Compare to Euclidean and absolute similarity, generally using histogram similarity measure is slower, almost doubled in running time.

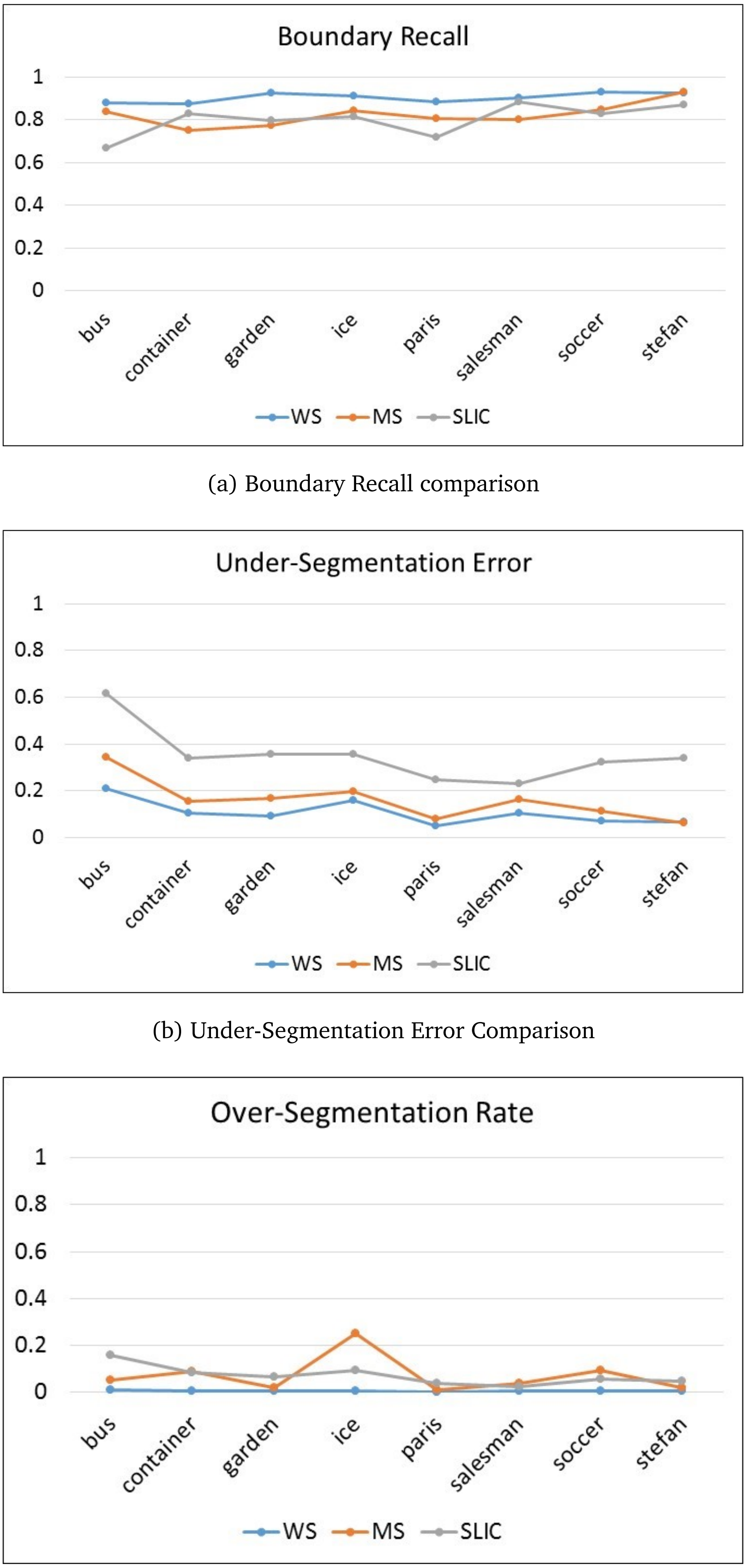
Table 4.4: Running Time for BPT Preparation and Simplification of All Algorithms Using Histogram Distance

No	Video Test	WS		MEAN SHIFT		SLIC	
		BPT	SIMPLIF	BPT	SIMPLIF	BPT	SIMPLIF
1	bus	649.46	43.17	41.96	2.47	9.06	0.26
2	container	881.34	118.70	8.66	0.85	9.80	0.31
3	garden	502.02	92.97	30.55	2.02	7.75	0.21
4	ice	629.46	81.86	2.95	0.10	9.78	0.26
5	paris	587.68	125.24	42.68	4.88	9.48	0.25
6	salesman	39.80	4.04	2.91	0.24	8.32	0.23
7	soccer	1,119.31	134.03	5.45	0.35	12.43	0.43
8	stefan	432.12	88.44	24.99	2.23	9.63	0.25
	average	605.15	86.06	20.02	1.64	9.53	0.27

A visual comparison of the simplification result can be seen in the figure 4.5

4.5 Discussion

The outputs of the pre-segmentation and simplification are assessed according to some parameters: boundary recall, under-segmentation error, over-segmentation rate. The time required to execute pre-segmentation, merging and simplification are recorded and compared. The effect of three different similarity formula are evaluated. At the pre-segmentation phase, the algorithm produces a nearly similar quality as indicated by

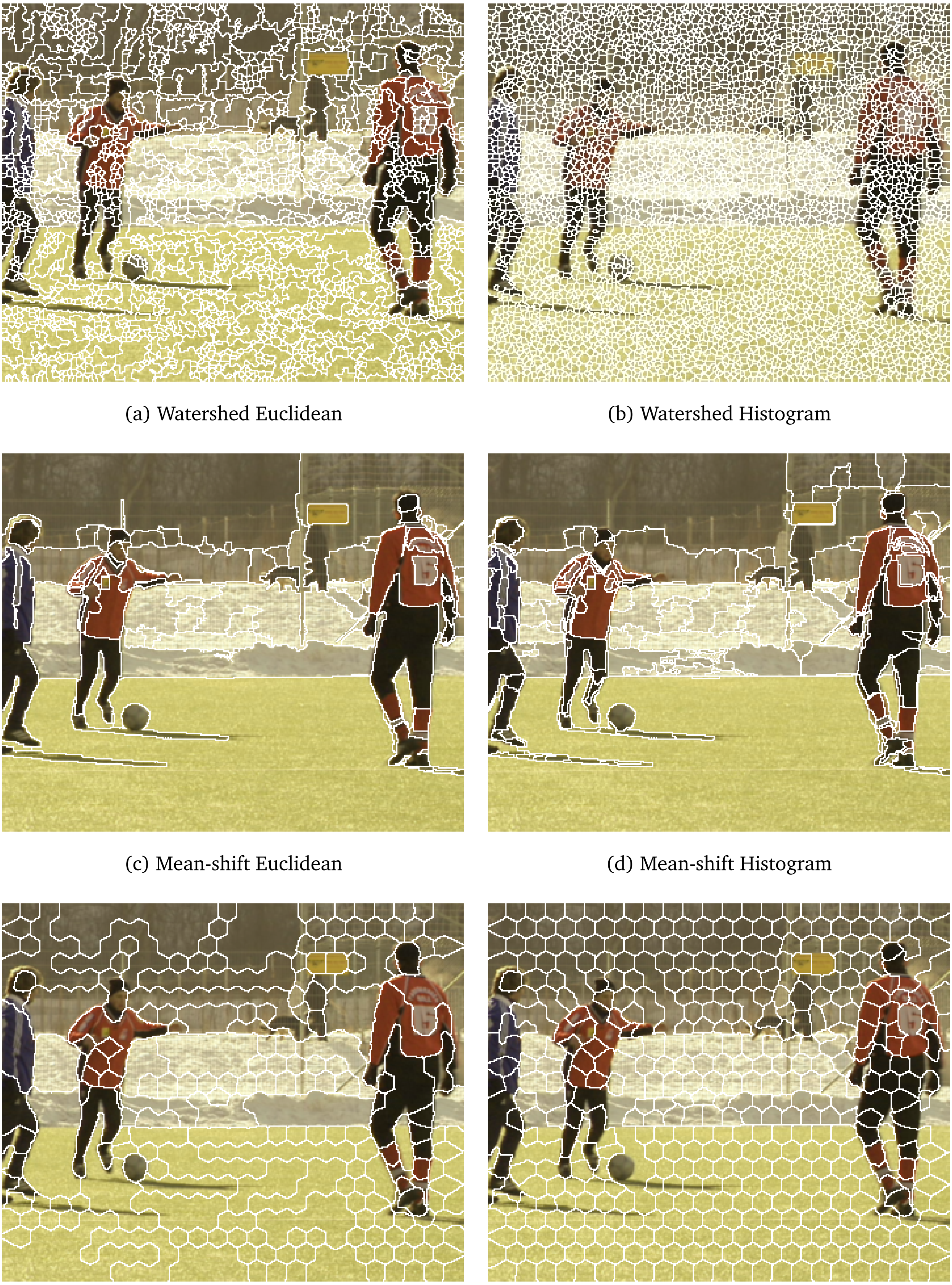


(a) Boundary Recall comparison

(b) Under-Segmentation Error Comparison

(c) Over-Segmentation Rate Comparison

Figure 4.4: Evaluation of Simplification Result by The Histogram Distance Measure



(a) Watershed Euclidean

(b) Watershed Histogram

(c) Mean-shift Euclidean

(d) Mean-shift Histogram

(e) SLIC Euclidean

(f) SLIC Histogram

Figure 4.5: Simplification Result for The First Frame of The 'Soccer' Video Using Euclidean and Histogram Similarity Measure

boundary recall and under-segmentation error. In terms of over-segmentation rate, watershed produces significant over-segmentation, where on average a single ground truth area corresponds to around 200 partitions. SLIC and mean shift, meanwhile, perform far better than watershed, with a single ground truth corresponding to around 20 partitions.

Table 4.5: Total Running Time Comparison

No	Video Test	EUCLIDEAN			ABSOLUTE			HISTOGRAM		
		WS	MS	SLIC	WS	MS	SLIC	WS	MS	SLIC
1	bus	529.4	28.1	7.7	529.9	37.4	9.6	712.7	45.6	9.8
2	container	522.1	5.0	7.1	523.2	7.2	8.9	1,021.6	10.8	10.6
3	garden	399.7	21.0	6.2	399.0	29.5	7.8	610.1	33.4	8.4
4	ice	421.8	1.6	7.3	422.2	2.3	9.2	726.0	4.2	10.5
5	paris	474.7	25.5	8.8	475.5	26.1	9.7	729.8	48.5	10.2
6	salesman	33.0	2.5	7.7	33.0	2.5	7.9	46.3	3.4	8.7
7	soccer	547.8	4.0	9.7	548.8	4.2	9.7	1,273.0	6.9	13.3
8	stefan	361.2	23.1	9.4	361.1	23.5	9.6	535.0	28.1	10.3
	Average	411.2	13.8	8.0	411.6	16.6	9.0	706.8	22.6	10.2

According to Tables 4.2 and 4.3 the largest portion of the running time is consumed in BPT preparations. The running time required by the histogram similarity measure is identified as the highest the among other similarity measures. According to Table 4.5, the best running time is achieved by SLIC in any similarity measure performed, while watershed shows the worst running time, consistently in all similarity measures.

An average performance of all combination techniques is presented in table 4.6 that refers to graph in figures 4.3, 4.3 and 4.4 and table 4.5. The analysis is based on BPT created over original segmentation produced by watershed using gradient image, means shift and SLIC in the CIElab colour image.

An average performance of all combination techniques is presented in table 4.6, that which is averaged from the graphs in Figures 4.3, 4.3 and 4.4 and table 4.5. The analysis is based on a BPT created over an original segmentation produced by watershed using a gradient image, means shift and SLIC in the CIElab colour image.

Table 4.6: Average Boundary Recall, Under-Segmentation Error, Over-Segmentation Rate and Total Running Time

No	Pre-segment methods	Similarity	Average			
			Boundary Recall	Under-segment	over-segment	running time (ms)
1	Watershed	Euclidean	0.8313	0.1190	0.0132	411.21
		Absolute	0.8314	0.1181	0.0130	411.59
		Histogram	0.9054	0.1057	0.0055	706.81
2	Mean shift	Euclidean	0.7775	0.1564	0.1391	13.85
		Absolute	0.7817	0.1592	0.1378	16.60
		Histogram	0.8237	0.1601	0.0720	22.60
3	SLIC	Euclidean	0.8205	0.1842	0.1018	8.00
		Absolute	0.8205	0.1897	0.1038	9.04
		Histogram	0.8020	0.3511	0.0705	10.22

The tree simplification result evaluation shows that the boundary recall has slightly declined compared to the pre-segmentation result. The best boundary recall is achieved by the histogram distance measure, while the Euclidean and absolute give slightly worse outcomes. If we look at the required running time, however, the histogram similarity measure needs almost twice as much compared to other distance measures. The most stable boundary recall is shown by a simplified tree of SLIC at around 0.8 to 0.82.

According to the evaluation, watershed pre-segmentation produces more partitions than other algorithms. The BPT preparation and simplification, therefore, needs heavy computational power. The histogram distance measure doubles the running time, due to the iteration needed to calculate the similarity coefficient. Although it affects the achievement of boundary recall, which shows a slightly better value compared to the result of Euclidean and absolute distances, the small benefit is not comparable to the heavy computing power required.

Due to the relatively constant quantity of the initial regions, SLIC ensures the number of iterations required for the remaining task. This helps the running time for BPT preparation and simplification of SLIC to remain stable and independent of the frame size. The drawback of SLIC is the input requirement of K. It will lead to inaccuracy when the user provides an unreasonably small K value. For that reason, in this experiment, K is set to be large enough, because the result of the pre-segmentation is prepared for the merging steps.

In this experiment, an implementation of mean shift introduced by [68] is adopted. No user input was needed to perform the pre-segmentation; therefore, the previous issue with SLIC does not exist, although the running time is longer than the other pre-segmentation algorithms. Although the boundary recall and under-segmentation results in the simplified tree does not produce the best results compared with other algorithms, mean shift consistently produces better over segmentation rate in all distance measures.

The result of a single frame segmentation produces a number of superpixels or regions. This task is performed in order to provide intra-frame segmentation in multi-scale details. The next step is to find the inter-frame relationship in order to identify the region correlation across frames. The identification of inter-frame BPT node relationship will be discussed in the next chapter. The regions/supervoxels and their adjacency graphs will be recorded in the database in order to support video content metadata. The region features will be recorded in the $superpixel$ table, while the adjacency graph is designed to be recorded in the $spatialEdge$ according to the database design in Figure 3.12.

4.6 Conclusion

In order to seek the most reliable techniques, boundary recall, under-segmentation error, over-segmentation rate and running time are presented in Table 4.6. The optimum performance should maximize the boundary recall and over-segmentation rate, meanwhile minimize the under-segmentation error. If the weights of boundary recall, under-segmentation and over-segmentation are equal, the best value is achieved by watershed with the histogram similarity measure.

In the watershed simplification result with histogram similarity measures, the over-segmentation rate is very small (at 0.055). This is indicated by the fact that the boundary recall is consistently high because the simplification chooses the low level nodes in the tree and only small simplification steps are performed. This is confirmed by the small change in the over-segmentation rate before and after simplification (from 0.057 to 0.055). This means that the initial segmentation does not undergo a significant simplification, as shown in Figure 4.5(b). According to Table 4.6, therefore, the most reliable method is mean shift with the absolute similarity measure.

If speed is considered, however, the SLIC with the Euclidean similarity measure outperforms the others. This gives the best running time, while achieving good under-segmentation error and over-segmentation rate. Furthermore, the boundary recall of this combination is better than mean shift with absolute similarity.

An interesting aspect of the watershed result is that, out of all the similarity measures, watershed gives the worst speed. This is due to the amount of initial segmentation, which causes subsequent merging, and means that the simplification needs many more iterations. The small over-segmentation rates indicate that the number of final segments is still far more than the expected number of segments in the ground truth. All of the watershed based techniques demonstrate this over-segmentation issue, which is confirmed by the visual results in Figure 4.5.

A result of the average of boundary recall and over-segmentation rate is achieved in high value by SLIC with Euclidean distance, and this combination gives the best speed of all the techniques (see Table 4.6). This suggests that, in this study, the SLIC initial segmentation with K set to be 1000 outperforms the alternative techniques. This conclusion is based on the evaluation of the simplification result compared to the ground truth from xiph.org data set.

Chapter 5

Spatial Approach

5.1 Introduction

This chapter is dedicated to discussing the spatial approach to video segmentation on BPT. The preliminary tasks, pre-segmentation, merging, BPT construction and simplification, have been discussed in Chapter 4. In the spatial approach, every frame is individually segmented. The remaining task is to create the region links across successive frames.

The frame pre-segmentation and simplification demonstrated in Chapter 4 produces two sets of regions. They are called pre-segmentation and simplification set. The set can be represented as a graph, in which the vertex/node reflect a region, and the edge represent the neighbouring relationship. The graph representing the intra-frame region relationship is called a spatial Region Adjacency Graph (RAG).

An inter-frame relationship is built as a node matching between two spatial RAGs.

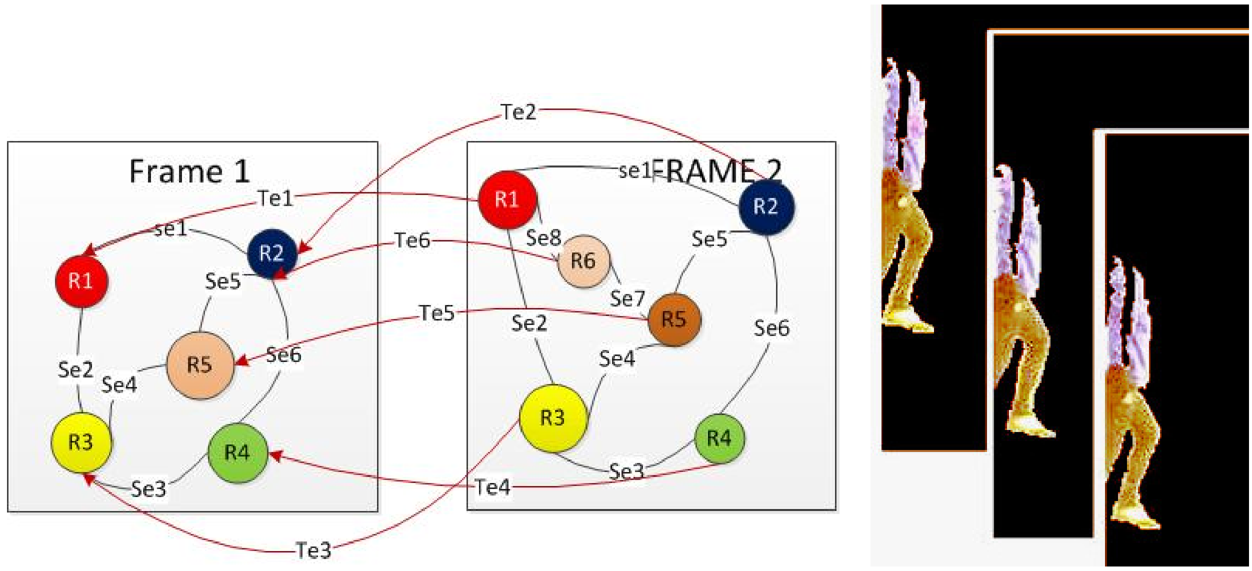
Temporal edge is defined as an edge connecting a pair of regions in different frames. There are three methods to be evaluated in order to build the temporal edge. The first technique is a limited window search. Secondly, a genetic algorithm to be employed to find the best pair of nodes in the binary partition tree at any level. The last experiment develops the temporal relationship between top salient nodes across successive frames.

5.2 Spatio Temporal Region Adjacency Graph (STRAG)

STRAG consists of a spatial and temporal adjacency graph. As discussed in previous chapter, a spatial region adjacency graph is created through a frame segmentation. In a region adjacency graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ are vertices which represent regions; $\mathcal{E}$ are edges which represent distance between neighbouring regions. The edges are spatially connected; therefore, they are called spatial edges. For instance, a video consists of n frames; it has $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, ...\mathcal{G}_n\}$ in every $\mathcal{G}_i$, there exists m vertices $\mathcal{V}_i = \{\mathcal{V}_i^1, \mathcal{V}_i^2, ...\mathcal{V}_i^m\}$. Spatial edges connect every neighbouring pair of vertices, for each $\mathcal{V}_i$ there exist $p < (m \times m)$ edges. On the other hand, temporal edges correlate vertices of the graph across frames $\mathcal{V}_i^r, V_j^s$ where $i, j \in \{1, ..., n\}n$ and $r, s \in \{1, ..., m\}$.

Temporal Edge is established by a matching operation between regions in subsequent frames. In a single video, subsequent frames are made up of similar objects; some of which are motionless, while others move. Most of the regions in the current frame are inherited from the previous frame, except if they belong to different shots of the video. Temporal edge aims to record those relationships. Due to motion, some regions have their counterparts in slightly different locations. Some static regions have their correlated regions at the same location. It can be expected that the region pair shares similar features such as colour but the location moves slightly, and the shape can thus be evolved.

There are some criteria underlying the choice of a region as a temporal counterpart, although it is important to note that some regions have no temporal links caused by disappearance from the scene or occlusion. The possible counterpart of a region in a subsequent frame is determined by position and feature similarity.



(a) Ilustration of Spatio Temporal Edge in Streaming Scenario

(b) Region Temporal Correlation of Region #100 in Frame #1 - #3 'Soccer' Video

Figure 5.1: Spatio Temporal Correlation, Illustration and Example

The temporal edge represents a short relationship. It is illustrated in Figure 5.1. In the illustration, there are five vertices in frame #1 and six in frame #2. In reality, the first frame of soccer, for example, has 103 while the second frame has 120 vertices. A vertex in the current frame can have n pairs in the previous and subsequent frames. An example of this relationship is depicted in Figure 5.1(b) for region label 100 in the first three frames of the soccer video.

5.3 Limited Windows Search Temporal Edge

In this method, a set of spatial segmentations are generated for a particular frame. The current frame (t) is considered as the frame where the region is located. The temporal edge will be made to the previous frame $(t-1)$. A bounding box of a particular region R_i^t. in the current frame is determined, and a twice bigger search window is projected to the preceding frame. This is designed to deal with possible region movements. Assuming there are l regions in the search window $R_j^{t-1} = \{R_1^{t-1}, \ldots R_l^{t-1}\}$, the smallest distance to the current region is selected as the region pair.

$$s = arg\, min_j \left(\left\| (\overline{\vec{c}_{R_i^t}} - \overline{\vec{c}_{R_j^{t-1}}}) \right\|_2 \right) \tag{5.1}$$

Where s is the selected index of projected region, therefore R_s^{t-1} is the pair of R_i^t, $j \in \{1 \ldots l\}$. This formula is executed for all R_i^t where $i \in \{1 \ldots n\}$, n is the quantity of the region in the current frame. The visual illustration of limited windows searching that we implemented in this proposal can be seen in Figure 5.2.

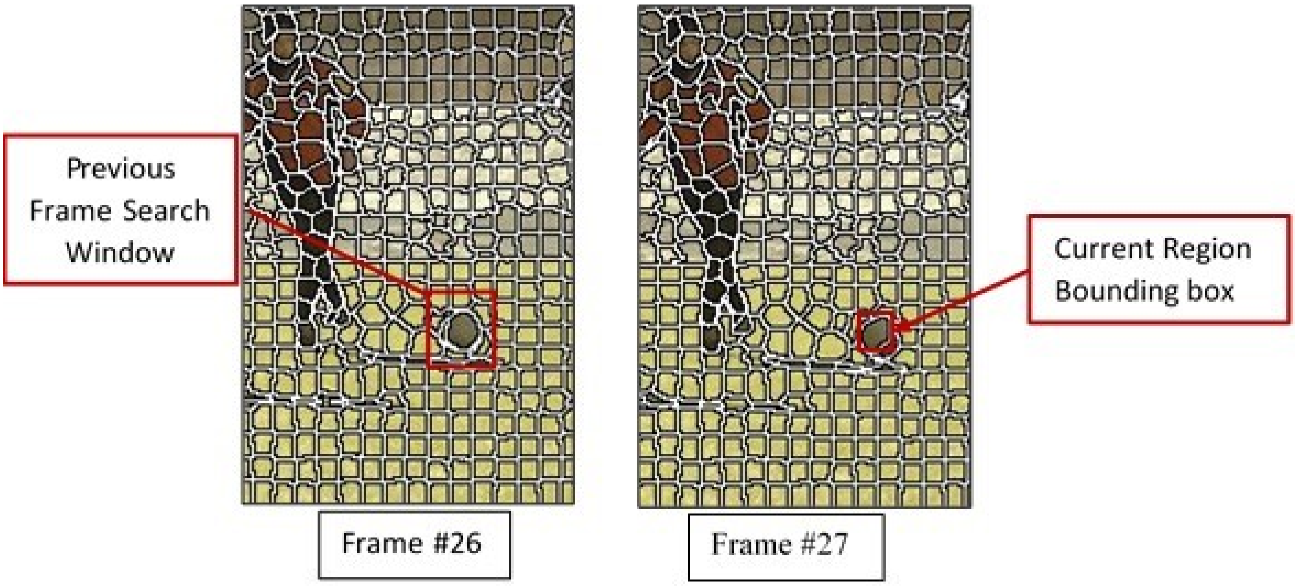


Figure 5.2: Current Region Bounding Box and Search Window in The Previous Frame

As illustrated in Figure 5.2, this is the actual snippet of frame 26 and 27 of Soccer video with the SLIC pre-segmentation ($k = 500$). It shows the pairing process to establish a temporal edge between a region (the ball) in the current frame (frame #27), and the prospective candidates in the search window in the previous frame (frame#26). The search window is twice the size, with a centroid in exactly the same position. The member of prospective candidates Rp consists of all regions touching the search window. There are 11 region candidates and the algorithm will choose the closest colour distance to the current region.

Accumulation of temporal edge throughout a video constructs a long-term relationship. The region trajectory can be observed, and their relationship can be explored with further merging criteria.

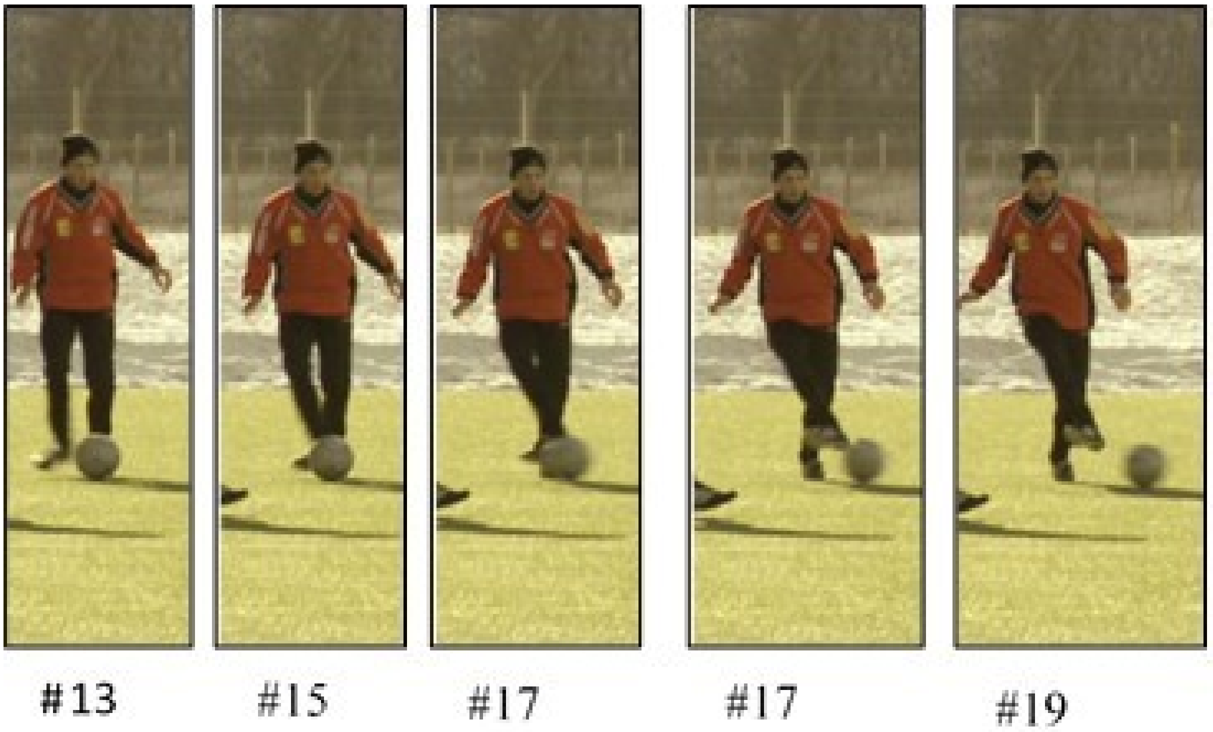


Figure 5.3: Invalid Long Term Neighbourhood Between The Ball and Foot of The Player From Frames #13 to #19

5.3.1 Long Term Temporal Relationship

A pair of regions which are consistently neighbouring each other and move in a similar direction and velocity is a cue that they might belong to a single object. In contrast, if there is a pair of neighbouring regions in some frames but in certain frames they are separated, it is reasonable to argue that they belong to different object. The idea of region long term neighbourhood merging has been proposed in [5]. Long term neighbourhood represents region activity in a group of frames in a certain shot.

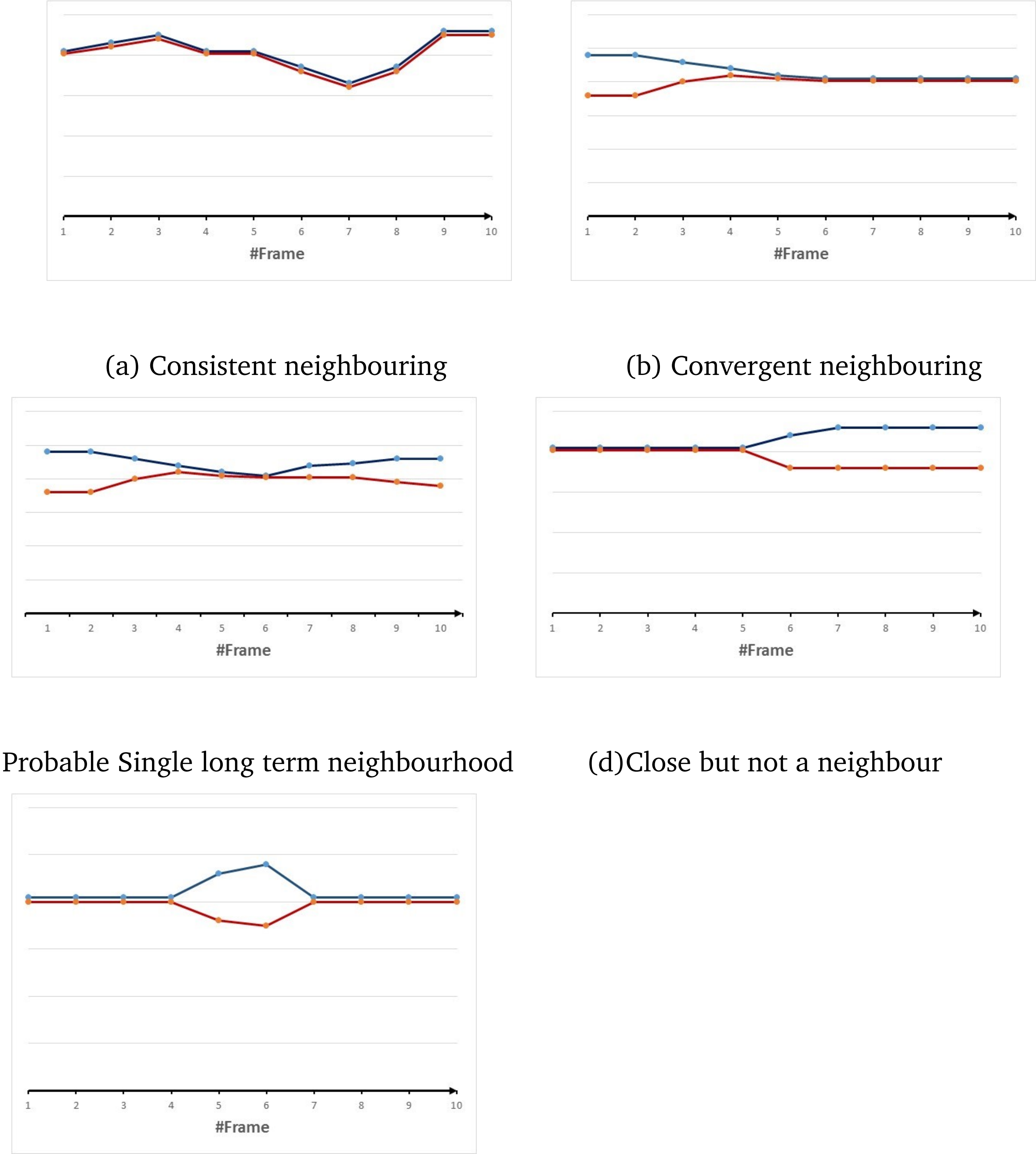
A long term region temporal neighbourhood is defined as a set of consistent neighbouring regions during a number of frames. They have to keep neighbouring each other in every frame they exist in. Their neighbourhood is no longer valid if this requirement is violated. R1 and R2 are considered to have a long term neighbourhood if they are spatially neighbouring in every frame in which they coexist. The long term neighbourhood between R1 and R2 is no longer valid whenever they are disconnected in any particular frames in which they coexist.

For example, the ball and the player foot in the Soccer video as shown in Figure 5.3, are initially in contact but after several frames (frame #19) the ball and the foot become disconnected. It is indicated that the ball, and the foot are different objects. Therefore, merging between the ball and the foot has to be avoided. In contrast, the neighbouring relationships between the red jersey and black trousers are consistently existed in all frames. As a result, they have valid long term temporal neighbourhood and merging between them is recommended.

Several conditions can happen in relation to a pair of long term neighbourhood, as can be seen in Figure 5.4.

5.4 Genetic Algorithm Approach

Genetic algorithms (GA) are search procedures inspired by biology and the workings of natural selection, and were originally proposed by John Holland in the 1960's [125]. Many implementations of this algorithm can be found in the image processing field. GA is particularly useful when an exhaustive search for the solution is expensive in terms of the computational cost. In our implementation, the start and desired final conditions are known, making this problem suited to a GA solution.



(a) Consistent neighbouring

(b) Convergent neighbouring

(c)Probable Single long term neighbourhood

(d)Close but not a neighbour

(e)Probable occlusion

Figure 5.4: Long Term Region Neighbourhood Relationship

The purpose of this experiment is to correlate objects temporally between successive frames in a video. Temporal correlation is important in video analysis for purposes such as: key frame identification, object tracking, activity recognition and cognition. A series of frames in a single scene generally contains the same objects, and therefore an object in the previous frame is likely to appear in the current frame and in future frames. The problem we propose is finding them in the corresponding binary trees. The object may move slightly or be subject to rotation, scaling or warping.

The problem is defined as searching the match nodes in two different binary partition trees. Given a single target branch which contains a number of region nodes of a BPT representing a salient and/or semantic object, where are the corresponding branches on the BPT of the subsequent video frame? This is not an easy problem to solve as the branch in one frame may map to many branches in the next frame or vice versa. This problem is well suited to a GA solution.

The Genetic Algorithm (GA) has been used by many researchers in image and video processing. A GA is used for image retrieval utilizing local similarity in [126]. Image enhancement and segmentation works take advantage of the GA e.g. [127]. The Genetic Algorithm is successfully implemented on a shape based object recognition based on Fourier's descriptors in [128]. An Interactive Genetic Algorithm (IGA) is used on low level image properties such as colour, texture and edge description as a basis for machine classification with a user in the loop in [129]. The GA is employed for emotion based video scene retrieval using an interactive genetic algorithm in [130].

In the approach utilized here a genetic algorithm is applied to identify the temporal correlation between a node in the tree of different frames in a video clip.

5.4.1 Genetic Algorithm on BPT

In order to perform a genetic algorithm, some data structures and rules must be applied to the chromosome coding, fitness function, crossover and mutation. The problem space is performed in a BPT. Therefore, a binary chromosome representation is selected. The first 1 digit represents the root of the binary tree. The following digits represent the left leaf where the value is 0 and the right leaf where the value is equal to 1. The number of genes in a chromosome is based on the level inside the binary tree. Some genes are valued 0 for the most significant bit, however. As illustrated in Figure 5.5, the binary tree consists of three levels, and the chromosome is coded on three binary digits. For a real image, the number of levels in a tree can vary from tens to hundreds of levels.

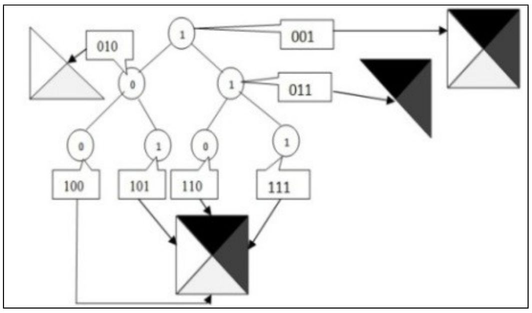


Figure 5.5: Chromosome

The fitness function plays a central role in genetic algorithms. A fitness function provides a measure of similarity between a target and the candidate solution. Each chromosome is considered as a solution candidate and compared to the target. Based on Figure 5.5, each chromosome represents one node in the BPT and a region area in the original image. In this research, provided that each region target and solution candidate has some statistical data (e.g. pixel intensity), it is compared according to the fitness

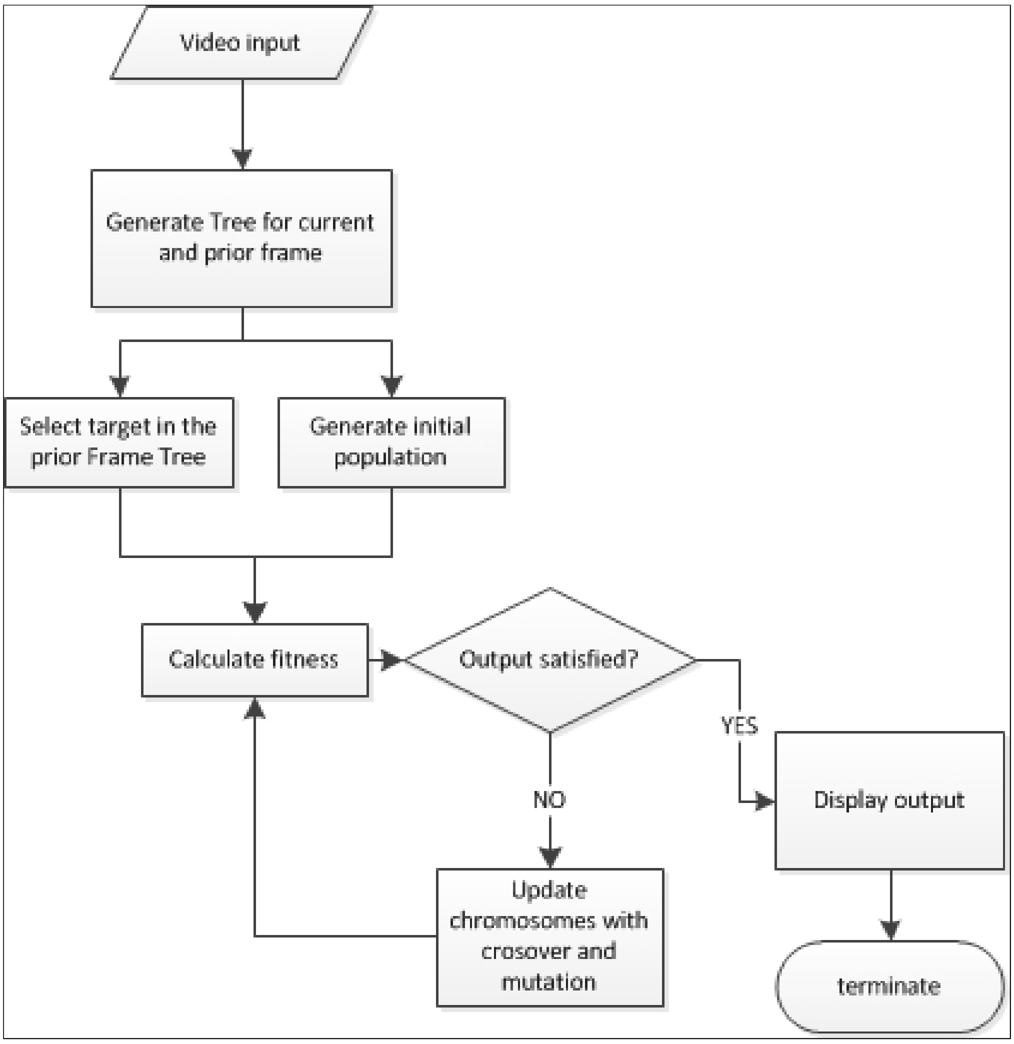

Video input
Generate Tree for current and prior frame
Select target in the prior Frame Tree
Generate initial population
Calculate fitness
Output satisfied?
YES
NO
Display output
Update chromosomes with crosover and mutation
terminate

Figure 5.6: Complete Flowchart of Proposed GA Algorithm

function defined with the following equation:

$$\varphi((R_i^t, R_j^{t+1}) = \begin{cases} 1 & \text{if } \delta_{e\vec{c}}(R_i^t, R_j^{t+1}) = T \\ \frac{0.99}{\delta_{e\vec{c}}(R_i^t, R_j^{t+1})} & \text{if } \delta_{e\vec{c}}(R_i^t, R_j^{t+1}) \neq 0 \end{cases} \tag{5.2}$$

Where φ denotes fitness value of a chromosome compared to the target, t denotes index of the t_{th} frame in a video sequence. $\delta_{e\vec{c}}(R_i^t, R_j^{t+1})$ is calculated using Euclidean distance formulated in 4.1. R_i^t denotes region i_{th} of frame t as a target region, and R_j denotes region j_{th} of frame $t+1$. $\overline{\vec{c}_{R_i^t}}$ denotes the region average colour value of region R_i in frame t, while $\overline{\vec{c}_{R_j^{t+1}}}$ denotes the same of region R_j on frame $t+1$. $\delta_{e\vec{c}}(R_i^t, R_j^{t+1})$ denotes the colour distance between region R_i in frame t and region R_j on frame $t+1$ and T is a constant set to zero when both regions are identical, the fitness value is set to be one.

The initial population is unique and randomly generated within the solution space. The lowest leaf is a very small region and there are a lot of regions at the lowest level. The randomly generated chromosomes are designed to select larger regions at higher levels nearer the root to produce faster solutions.

At each iteration of the GA the quality of the chromosome relative to the target gradually improves. Crossover and mutation are two operators which aid the solution and play an important role in modifying the chromosome in order to search for non local solutions. Crossover is an attempt to improve the fitness of the chromosome genes by combining the genes of two high-ranking chromosomes. A roulette mechanism is chosen in our algorithm. Each time two chromosome parents are chosen, the crossover mechanism is applied until a better fitness value is achieved compared to both of their parents, or a maximum crossover is achieved. In the case of maximum crossover the process is aborted and the original chromosome survives to the next generation.

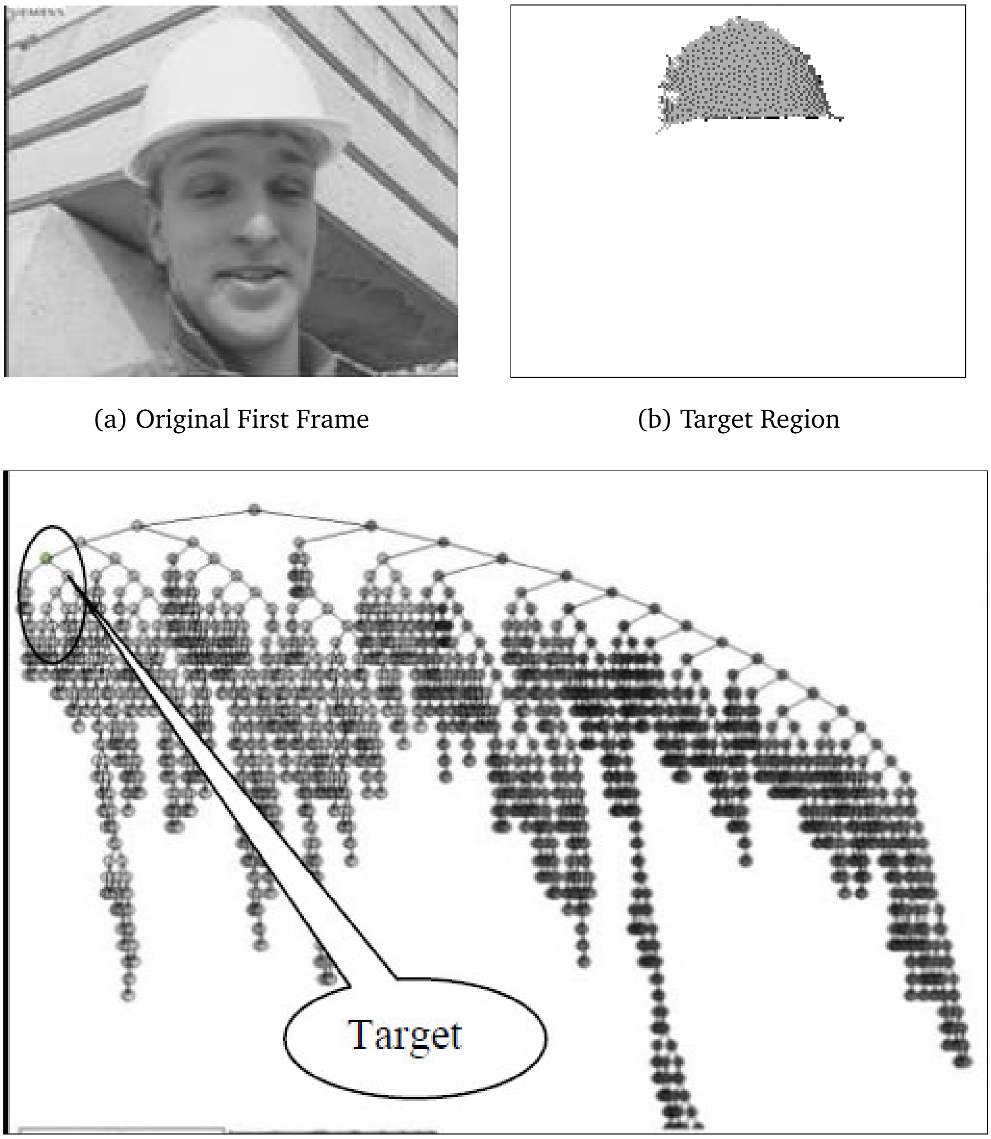
Mutation is another operation, which aims to achieve better chromosomes in the subsequent generations. This operation is done every time a random value is greater than the mutation threshold. Mutation will be executed until a better chromosome is achieved or the maximum number of mutation attempts (m) is reached. The mutation rate parameter is an empirical value in this algorithm. Mutation is aborted when the fitness value falls below that of the original chromosome. If this is the case, the original chromosome will live on in the next generation. Mutation can be achieved either by random selection of a gene to be changed from one to zero or zero to one or a bitwise operation shifting left or right. A flowchart of the proposed algorithm is illustrated in Figure 5.6.

5.4.2 Experiment Result and Discussion

Table 5.1: Genetic Algorithm Parameters

No	Parameter	Value
1	Population Size	100
2	Crossover rate	0.7
3	Mutation rate	0.1
4	Max Generation	40
5	Min fitness	500
6	Max crossover attempt	10
7	Max Mutation attempt	10
8	Max solution branch	100

In the experiment, the genetic algorithm is configured according to Table 5.1. The results of the first experiment are shown in Figure 5.8. A branch is selected as the target and represents the helmet object of the foreman on the BPT as illustrated in Figures 5.7(b) and 5.7(c). The solution is found in the next frame as illustrated in Figure 5.8. The solution is found on the third iteration of the GA. The solution is a single branch



(a) Original First Frame

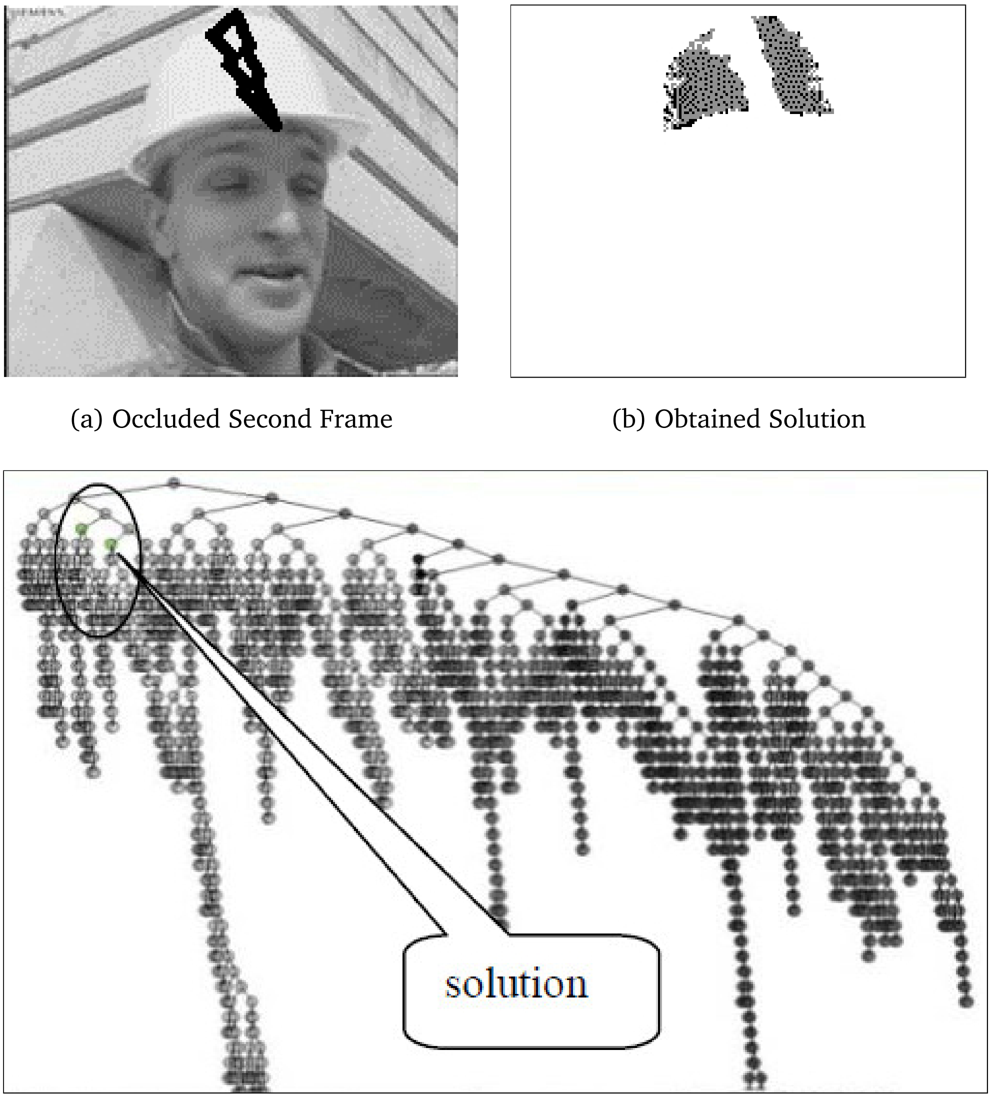
(b) Target Region

(c) BPT of Fisrt Frame and The Selected Target Node

Figure 5.7: First Frame of 'Foreman' Video and Selected Target

which starts from the circled node in Figure 5.8(c).

In our experiment, a scenario of occlusion in the region target can be seen in the Figure 5.9, the region and nodes target still the same as illustrated in Figure 5.7.



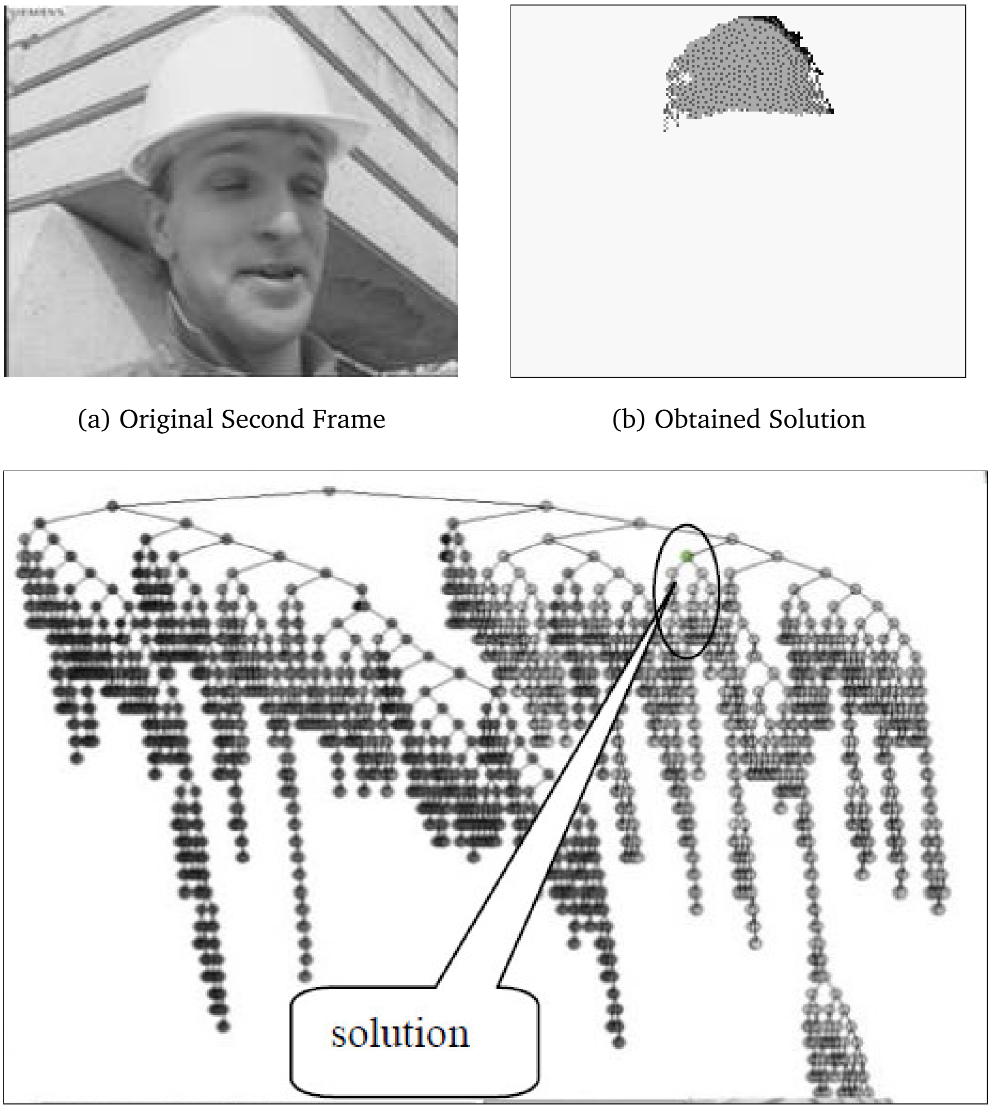
(a) Occluded Second Frame

(b) Obtained Solution

(c) BPT of Second Frame and The Obtained Solution Nodes

Figure 5.9: The Second Frame With Occlusion and Obtained Nodes Solution

As can be seen in Figure 5.9, because the target region is completely occluded (fore-



(a) Original Second Frame

(b) Obtained Solution

(c) BPT of Second Frame 2 and The Obtained Solution Node

Figure 5.8: The Second Frame Without Occlusion and Obtained Solution

man hat in the second frame), the corresponding region is obtained from two different branches of the tree.

The first experiment is conducted with one to one searching, while the second experiment is conducted to test the ability of the proposed algorithm to deal with scattered solutions across several branches. In the real world, this happens for example when an object is covered by another object or an object is broken into smaller pieces.

The results show that the algorithm can precisely find the target. This shows that the algorithm works in both occluded and non-occluded target. In our experiment, the genetic algorithm has successfully solved the problem of temporal correlation of objects inside a binary tree representation. The algorithm can successfully identify regions inside a binary partition tree and find their best match according to a cost function in subsequent frames. The algorithm is robust and can cope with objects either within a single branch or separated into a number of branches. Furthermore, the measure of the fitness function provides a measure of the quality of the object match. This algorithm can also find a target object in the presence of translation, rotation and occlusion.

5.5 Inter-frame Salient Region Matching

5.5.1 Region Based Saliency in a Single Frame

BPT provides multi resolution segmentation representations. The root on the top level represents an entire image, and the lowest leaves represent the initial pre-segmentation results whilst being small and (most likely) meaningless. There must be a salient node in between the top and lowest level. By definition, saliency is a region which has popped out around their neighbours.

In this experiment, the binary partition tree (BPT) is generated for every frame from the video sequence. Each node in the BPT (except the root), is evaluated according to a distance measure to its parent. A sharp change in the mean colour between child and parent indicates that the child node is a prominent region. In order to identify the noticeable regions in a frame according to this measurement, a rank-ordered region is presented. A higher rank indicates the region as being more conspicuous in a frame. Salient nodes can be found anywhere in the BPT but are obviously larger toward the top of the tree.

5.5.1.1 Region Node Saliency Calculation

According to [7], each node in the BPT experiences an evolution from a small initial child region node to the whole image represented by the root. During the evolution from low levels of the BPT towards the root, each node is merged with its neighbour to form a new parent node. As can be seen in Figure 5.10, every node in the binary partition tree forms a triangle consisting of left child and right child with one parent. Every parent region contains the accumulation of area, mean colour and centroid from both child nodes. If those child nodes are homogeneous, the colour distances between them are insignificant. Consequently, the parents mean colour distance to its child will be relatively small. In contrast, a pair of heterogeneous child nodes results in a higher distance between parent and child nodes. When the difference is large, it indicates a critical merging occurs, and therefore, the region may be distinct (i.e. salient).

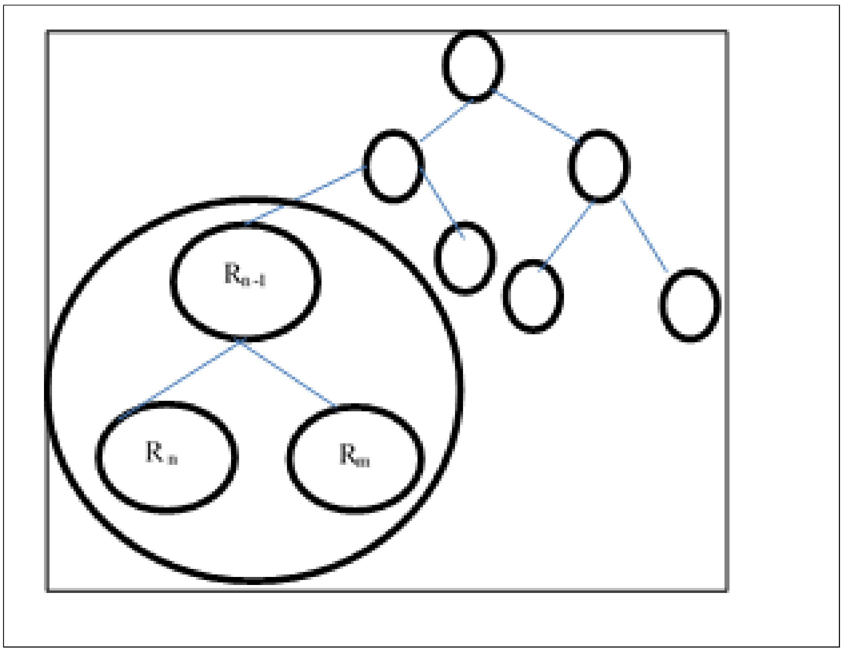


Figure 5.10: Child and Parent Nodes Triangle

Each node, (except the root) is examined for saliency by calculating the distance from its parent. Rather than defining a path from the lowest child node to the root, as suggested in [120], in this work, the distance to its parent is calculated for every child node in the BPT. The distance $\delta_{e\vec{c}}(R_n, R_{n-1})$ and $\delta_{e\vec{c}}(R_m, R_{n-1})$ are calculated based on Euclidean colour distance in 4.1. Where R_n and R_m are left and right child and R_{n-1} is the parent node. $\delta\vec{c}(R_n, R_{n-1})$, $\delta\vec{c}(R_m, R_{n-1})$ denote the colour distance of node n, and node m to its parent (node $n-1$).

Each region/node is ordered by its distance to the parent, meaning that the highest distance is considered as the most salient. The more salient regions are more likely to be found in the future frames of the video.

5.5.1.2 Region Node Feature selection

In the experiment, we use location, colour and area as the distance metrics utilizing Euclidean norm. Two regions at the current and future frames are paired as a match if the distances between them are small enough. The contribution from each variable in the Euclidean estimation needs to be scaled according to the individual importance.

Weighting factors can be varied, according to the video content. In this experiments, with the weighting factor was chosen manually, although we did means for automatic selection, but it is beyond the scope of this chapter. We can model the feature of a region n as $R_n = (c_1, c_2, c_3, x_c, y_c, a)$. c_1, c_2 and c_3 denote colour features of a region (R_n). The colour feature depends on its colour space, in RGB, for example; c_1 = Red, c_2=Green and c_3 = Blue colour components. x_c and y_c denote the centroid in horizontal and vertical axes respectively. a denotes the size of the region (R_n).

5.5.2 Region Temporal Matching

This experiment is designed based on the reasonable premise that a salient region in the current frame will also be present in a future frame. In the processing of a single frame, the more salient regions will be paired before the less salient regions. It is sensible to give priority to the more salient regions, because they are more distinct and should be easier to find in future frames.

5.5.2.1 Inter-Frame Region Similarity

In order to determine the region link across frames the similarity between them has to be calculated. A region has a number of features, which are location, intensity, colour, and area. A matching rate between regions in two consecutive frames is determined according to the similarity in colour, position and size. Let us consider R_i is a region i in current frame t, and R_j is a region in the upcoming frame $t+1$. Colour similarity between R_i^t and R_j^{t+1}, $\delta_{e\vec{c}}(R_i^t, R_j^{t+1})$ is calculated using Euclidean colour distance referred to equation 4.1. The similarity between them can be calculated using the equation 5.5.

$$\delta\vec{ct}(R_i^t, R_j^{t+1}) = \sqrt{(x_{cR_i^t} - x_{cR_j^{t+1}})^2 + (y_{cR_i^t} - y_{cR_j^{t+1}})^2} \tag{5.3}$$

$$\delta s(R_i^t, R_j^{t+1}) = 1 - \left(\frac{\min(a_{R_i^t}, a_{R_j^{t+1}})}{\max(a_{R_i^t}, a_{R_j^{t+1}})}\right) \tag{5.4}$$

$$\delta_{cts}(R_i^t, R_j^{t+1}) = \beta_1\delta_{e\vec{c}}(R_i^t, R_j^{t+1}) + \beta_2\delta_{\vec{ct}}(R_i^t, R_j^{t+1}) + \beta_3\delta_s(R_i^t, R_j^{t+1}) \tag{5.5}$$

Where β_1, β_2 and β_3 are the weighting coefficients for colour, centroid and size respectively. $\delta_{cts}(R_i^t, R_j^{t+1})$ is and inter-frame region distance, considering colour, centroid, $\delta_{\vec{ct}}(R_i^t, R_j^{t+1})$ is inter-frame centroid distance, and $\delta_s(R_i^t, R_j^{t+1})$ is the inter-frame size distance.

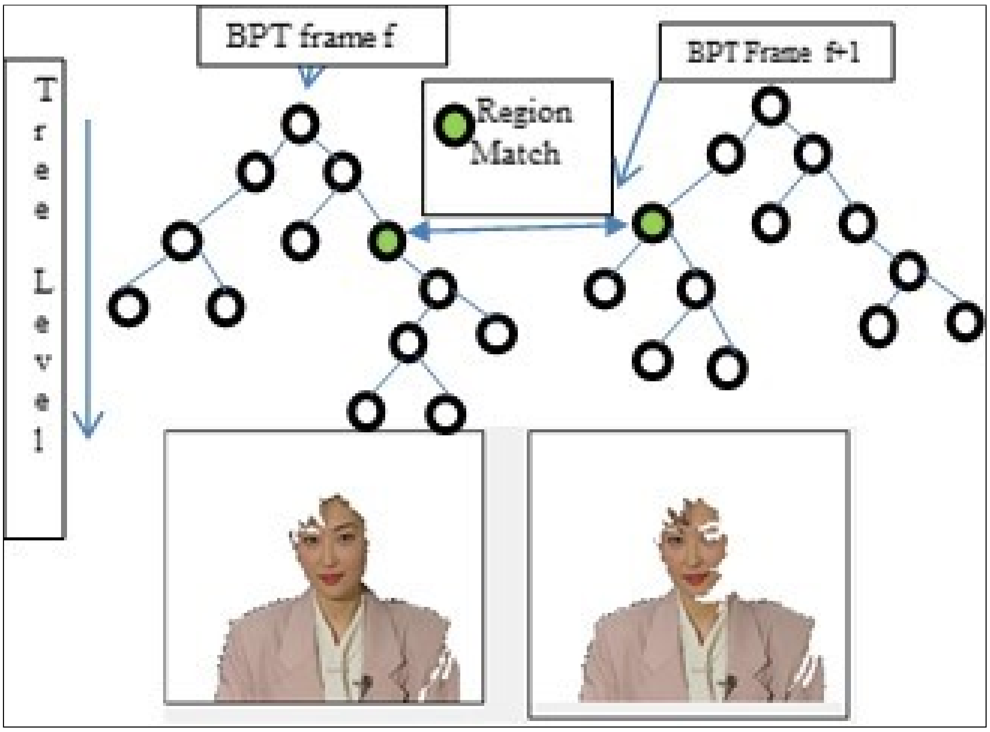


Figure 5.11: Inter-Frame Region Pair

5.5.2.2 Regions Temporal Saliency

A salient region in the current frame will be matched with a region with the highest similarity in the upcoming frame. The process can be repeated until the entire area of the image is covered. It is not necessary to examine all the nodes in the BPT since there is significant redundancy within the tree. A number of node pairs can be identified across the temporal domain. An example of nodes' pair across frames is shown in Figure 5.11.

In order to match region pairs in successive frames, the algorithm below is applied:

1. Calculate spatial saliency for all nodes in the BPT of the current and upcoming frame by measuring the colour distance between the leaf node and the parent node according to equation 4.1.

2. Order the list of regions according to the distance measure and produce an ordered list for both frames.

3. Enter the maximum rank of salient regions which will be matched against salient regions in upcoming frames e.g. the first 20.

4. Find the closest match for every salient region in the current frame against regions in the upcoming frame according to equation 5.5.

5. Order the list of the region pairs according to their distance.

Although there are thousands of region nodes, the constraints such as colour and location make the number of calculations relatively small. Due to the ordering used in both region lists, high ranks in the first frame are likely to match with high ranks in the upcoming frame.

5.5.3 Salient rank ordered

Every salient region candidate is rank ordered according to its saliency level. This is calculated according to its difference compared to surrounding regions, area and position. The salient region list is paired to the salient rank ordered list. In order to minimize iteration, it is expected for the top salient list to get its matching pair from the top salient list in the subsequent frame.

5.5.4 Experiment Result

In the experiments, we examined the 20 most salient regions of the first three frames in the test video 'Akiyo'. In order to illustrate this, a number of selected regions are shown in Table 5.2. Table 5.3 shows the scores for the first three frames of 'Akiyo'. The greatest similarity is observed at the top of the table with the differences increasing downwards.

In the next experiment, the temporal region match is examined for two consecutive frames. The results are ordered according to the inter frame distance. The lower the distance, the greater the similarity; the distance could be zero representing an exact match for position, colour and area. A lower rank indicates a higher matching error and a decreased likelihood it is the same object. A region can have high spatial saliency but this does not guarantee a high temporal match.

The results are repeatable across most video sequences but we present the results for frames 1 and 2 of 'Akiyo'. The algorithm maps the 20 topmost salient regions in consecutive frames. Some of the matches are very good and have minimal error whilst matching errors can be identified particularly for small non saturated regions. The result below is between first and second frames, threshold = 28, centroid coefficient $\beta_1 = 0.75$, mean Colour coefficient $\beta_2 = 1$, area coefficient $\beta_3 = 1.5$, colour space: CIE l*a*b. In The result for the top saliency rank in the frames 1 to 3 is shown in Figure 5.2.

The first 20 regions cover approximately 47 percent of the total image area. The increase in distance for the first 20 most salient regions is shown in Figure 5.12. An abrupt increase in the distance can be clearly observed at the rank of 11 and the rest of the graph. This indicates when matching errors start to happen.

Table 5.2: Spatial Saliency Rank Ordered for Frames 1, 2 and 3

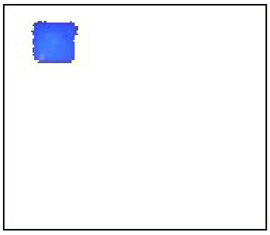
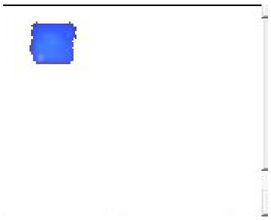
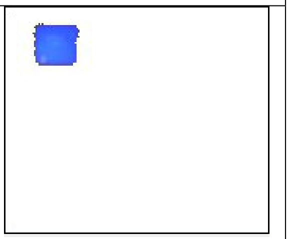
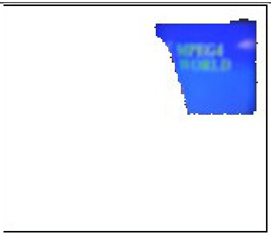
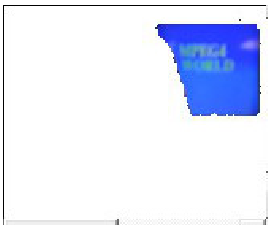
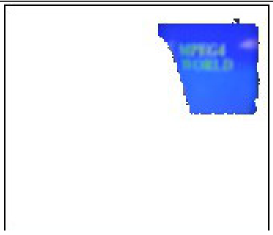
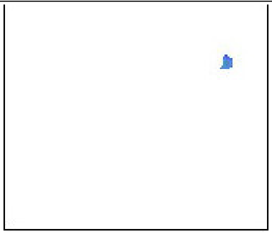
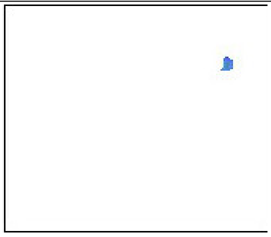
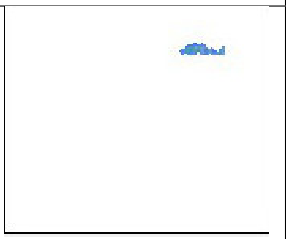
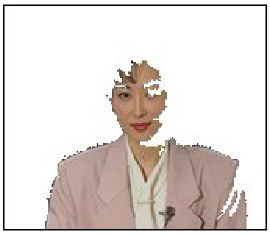
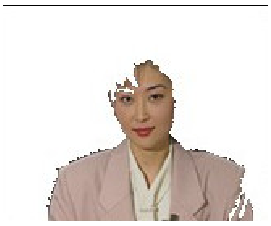
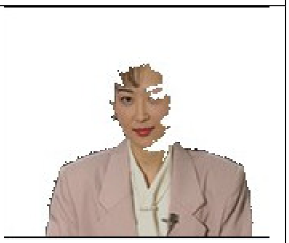
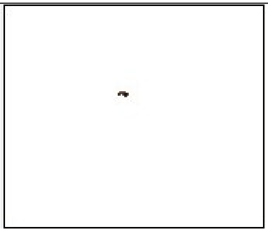
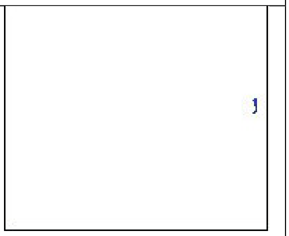
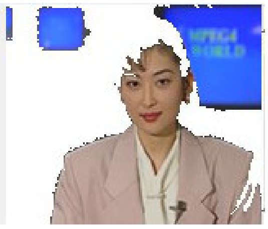
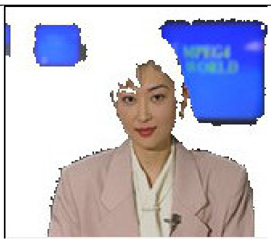
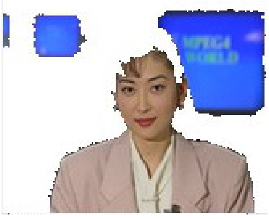
Rank	Frame 1	Frame 2	Frame 3
1			
3			
8			
9			
10			
All			

Table 5.3: Spatial Saliency Ordered List of Frame 1,2, and 3

Rank	Frame 1	Frame 2	Frame 3
1	81,548	80,809	81,562
2	73,315	74,638	74,214
3	69,916	69,667	70,007
4	42,874	43,979	59,336
5	39,993	40,256	50,255
6	38,003	39,271	43,764
7	36,135	37,518	41,638
8	35,369	35,725	37,375
9	31,501	30,948	30,901
10	29,954	29,474	30,348
11	29,436	28,018	30,250
12	27,653	27,094	28,711
13	27,394	26,329	27,466
14	25,749	25,889	26,164
15	23,935	24,373	25,222
16	22,462	22,592	23,441
17	22,422	21,791	23,069
18	22,390	20,502	22,635
19	22,355	20,428	20,075
20	21,853	20,247	19,176

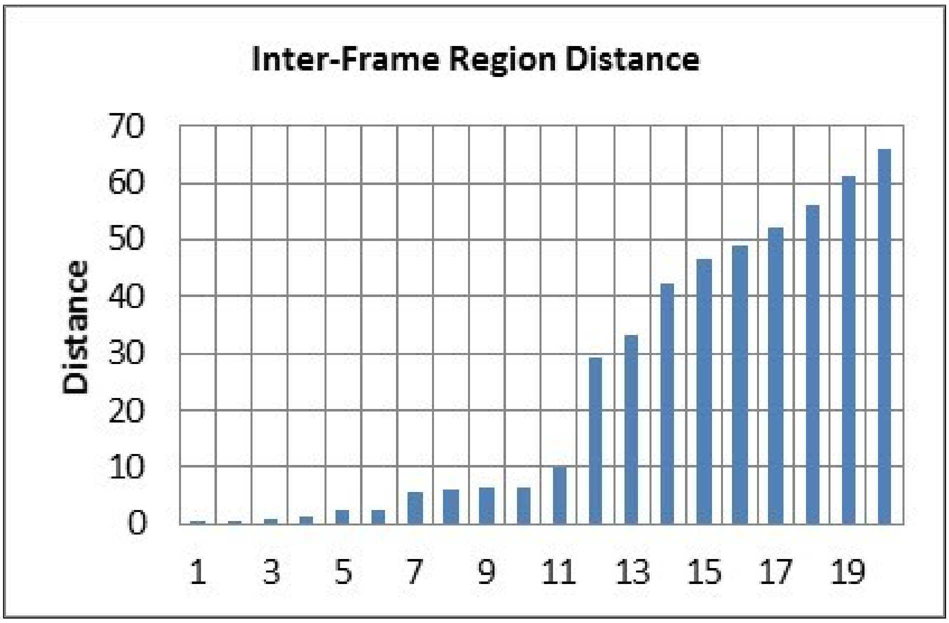


Figure 5.12: Ordered Region Pair Distance Frame 1 and 2

5.6 Conclusion

A spatial approach is demonstrated where the segmentation is carried out individually for each frame. Region correlation across frames has to be calculated accordingly. In this chapter, three methods to establish a correlation between regions across frames are demonstrated.

Limited window searching is designed with an assumption that regions in the current frame have its pair in the previous frame but the position is not always static. The correlated region is therefore expected to be found over a twice bigger area than the current region size. Thereby, if the region moving as far as their size can be anticipated. This technique is dedicated to build all regions in consecutive frames. This technique is an iterative method. Consequently, the computation is highly depended on the number of segments in each frame. Let's assume that the twice bigger window contains nine prospective regions, and we have 1000 regions in every frame. The iteration would be nine thousands for every pair of consecutive frames. In fact, watershed, for example, gives around 5000 regions per frame, mean shift and SLIC around 1000 and 500 respectively.

In order to avoid that heavy computation, an inter frame salient region matching is demonstrated. The region in every frame is ranked on its saliency. Saliency is defined as the child-parent distance in the tree structure. The mating operation performs for short list salient regions. Because the calculation of high saliency in each frame is carried out independent of each other, it is possible that some salient regions in the particular frame does not exist in the list of another frame. This issue causes some low score match region pairs to be incorrect pair, in Figure 5.12. The inter-frame region distance sharply increased at rank 11. It is indicated that they are incorrectly matching to the region in another frame.

A genetic algorithm approach is designed to conduct searching in the binary tree structure. The target region is picked from a node on the tree of a particular frame. The searching is conducted in the tree of any other frame. The expected result is the closest regions in the target frame. It is demonstrated even though the object has been covered by an occlusion, the algorithm still be able to find the target. This approach is not designed to established an inter-frame region link.

According to the discussion above, region link calculation in general suffer from high computational loads. Alternatives need to be explored, and a volumetric approach is one of the possible ways, as will be discussed in the next chapter.

Chapter 6

Volumetric Approach

6.1 Introduction

The purpose of this chapter is discussing a volumetric approach whereby the three-dimensional matrices employed to represent a video. Every picture element in a video (voxel) is considered as a point in a three-dimensional space of x (horizontal) y(vertical) and t (temporal) axis. The segmentation is carried out to group the voxel in regard to a particular homogeneity criterion so as to produce a number of non-overlapping supervoxels/volumes. Pre-segmentation is prepared using the watershed and SLIC algorithms in 26 voxel neighbourhoods. Initially, the partition is highly over-segmented, so further merging step and simplification need to be performed.

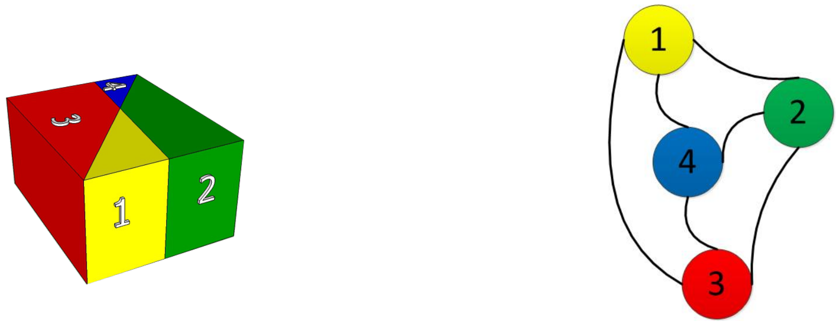
In order to obtain a simpler version of the segmentation, a merging task is performed, and the supervoxels, merging history is recorded in the BPT structure. As previously discussed in the general framework, the identification of critical merging is carried out in order to discover the pruning nodes on the tree. By cutting the tree at the pruning point, a simpler tree is obtained and, as the result, a simple version of segmentation is produced.

6.2 Supervoxel Segmentation

In order to provide a set of homogeneous partitions, a pre-segmentation task is conducted. A video can be considered to be a matrix of voxels in the spatio temporal (x, y, t) space. Our work is based on two supervoxel algorithms. Watershed algorithm can be applied to partition the voxels into three-dimensional matrix using 6, 18 or 26 pixel neighbourhood (allowed in Matlab). Watershed work on gradient magnitude of the video input. Secondly, [69] uses a fast implementation of the improved K-means (SLIC) algorithm in local search area. As a result, it is faster without necessarily needing to get the whole object in a global search.

The main benefit of supervoxels is the consistent label inter frame, meaning that it can avoid matching tasks among regions in the subsequent frames. Matching regions across frames has been discussed in Chapter 5, but is both error prone and requires a heavy computational load due to many-to-many inter-frame region relationships. Supervoxels carry spatial and temporal information (motion) in a single representation.

Figure 6.1(a) illustrates four supervoxels as a result of segmentation and the volume adjacency graph in Figure 6.1(b). Neighbourhoods are divided into two categories: temporal and spatio-temporal neighbourhoods. As can be seen in Figure 6.1(a), supervoxels 2 and 4 are temporal neighbourhoods wherein supervoxel 2 is before 4. Spatio-temporal



(a) Volumetric Segment

(b) Corresponding VAG

Figure 6.1: Illustration of Volumetric Segmentation and a corresponding Volume Adjacency Graph

neighbourhoods are illustrated between supervoxel 1 and 2, where they are coexist in the same time instant. Spatio temporal neighbourhoods occur only if a pair of supervoxels are spatially neighbouring in every frame in which they coexist. In other words, if two supervoxels only share borders in a number of frames but are separated in the remaining frames, they are not considered as valid neighbours. They therefore have no chance to merge in order to form a greater supervoxel; this avoids merging between two supervoxels belonging to different object.

6.2.1 Three Dimensional Watershed

Watershed algorithms can be expanded into three-dimensional space by utilizing 6, 18 or 26 pixel neighbourhoods. The fast implementation is available in a standard Matlab function which is based on the flooding approach [67]. Table 6.1 shows pre-segmentation results for some test video sequences.

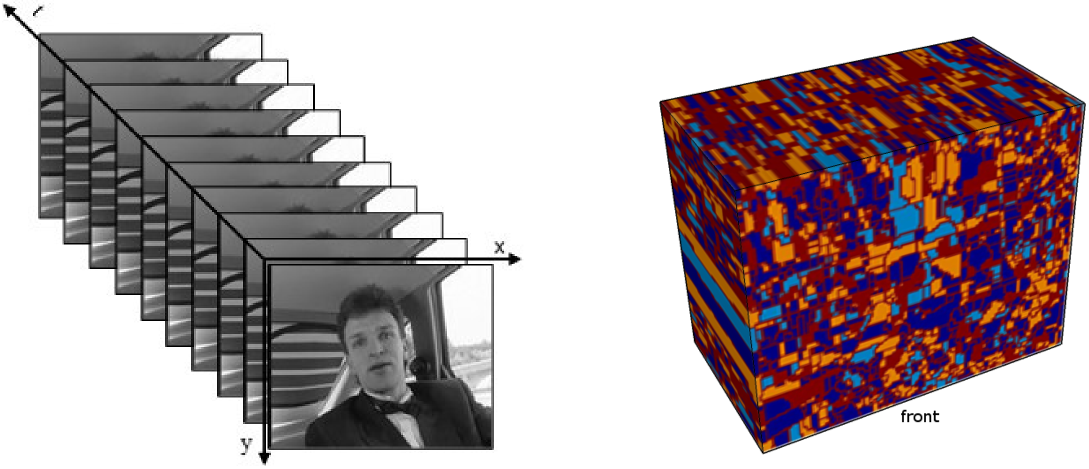
Table 6.1: Pre-Segmentation Result and BPT Before Simplification

Video	Size	Initial Supervoxels			Nodes	Level	Edges
		Quantity	Size	frames			
Carphone	144x176x20	2675	106.25	3.556	5349	47	10563
Soccer	288x352x21	16946	62.611	3.419	33891	62	67158
Stefan	240x352x21	18938	41.854	3.797	37873	76	75747

The visual result of the watershed algorithm is shown in 6.2(b). The 3D plot is made for frames 1 to 10 of the 'Carphone' video which is illustrated in Figure 6.2(a). As can be visually observed, the size of the supervoxels is varied but most of them are small. The duration of a supervoxel is around 3 frames. If we compare to the ground truth of this video, which has 16 isolated object candidates in the soccer video, in the whole 10 frame duration, the pre-segmentation produces almost 17,000 supervoxels. Therefore, a single object candidate in ground truth would consist of around 1,000 supervoxels.

6.2.2 SLIC Supervoxel

[69] introduce a fast implementation of a well known K-means clustering algorithm, by limiting search area into a particular window. As a derivation of K-means, SLIC needs K values either direct or indirect. The value of K determines how many segments are expected. One may argue that this is a disadvantage due to the dependency of the algorithm on the value of K. In this proposal, however, supervoxels are not intended to be the final result, therefore, setting a high value of k leads to an over-segmented result.



(a) Original 10 Frames ‘Carphone’ Video

(b) Over-segmented in 3D Plot

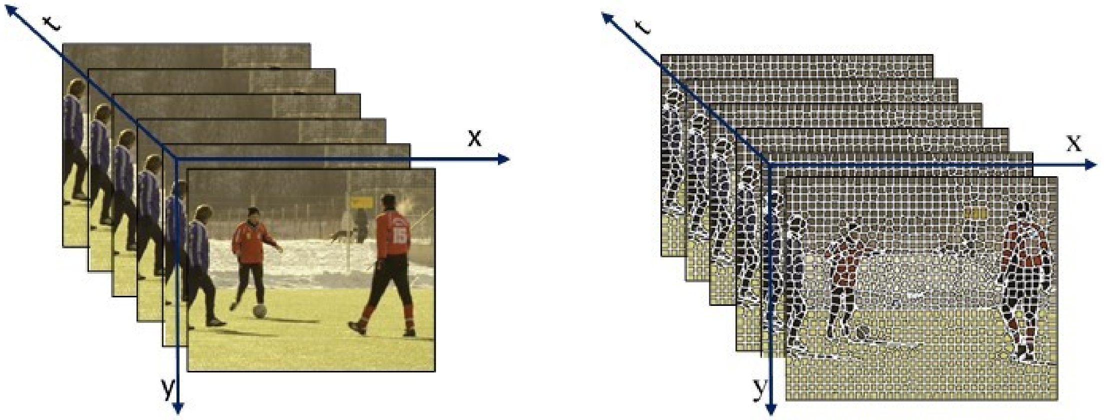
Figure 6.2: Watershed Pre-Segmentation of ‘Carphone’ Video

Later, the small partitions are subject to the next merging process. The publicly available implementation of SLIC is provided by [69]; in this experiment the initial volume is set to 1000. The initial supervoxels before K-means is executed in 10 x 10 x 10.

The result of the initial segmentation of the test video can be seen in table 6.2.

Table 6.2: Pre-Segmentation Result of SLIC supervoxels

No	Video		Supervoxel			Time(s)/Frame
	Name	Size	Quantity	Size	Frames	
1	Carphone	144x176x21	716	743	8.8	0.13
2	Soccer	288x352x21	2865	743	8.65	0.73
3	Stefan	240x352x21	2420	698	9.25	0.47



(a) The First 6 Original Frames (b)Volume Segmentation Using SLIC

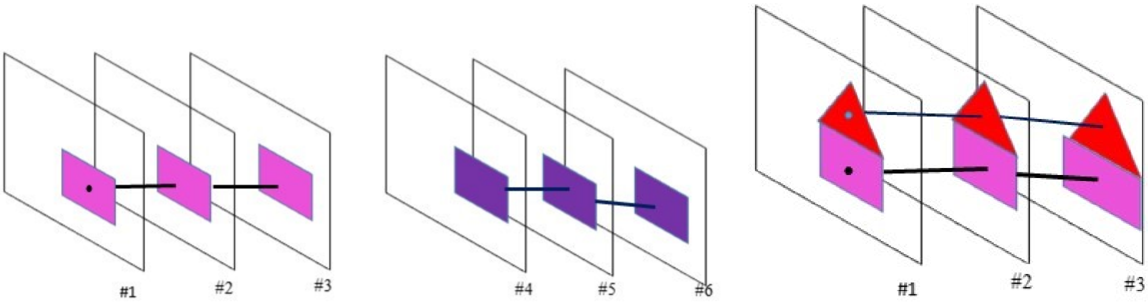
Figure 6.3: SLIC Supervoxels Pre-Segmentation of 'Soccer' Video

An example of a pre-segmentation result using SLIC on the 'Soccer' video is illustrated in Figure 6.3(b). In the experiment, 21 frames are segmented in a single execution, and the pre-segmentation speed is quite fast, as can be seen in the running time column in Table 6.2.

6.3 Supervoxel Merging

6.3.1 Supervoxel Neighbourhood

Neighbouring relationships between supervoxels are defined as temporal and spatio-temporal. A temporal neighbourhood occurs when a supervoxel exists before another as illustrated in Figure 6.4(a). They share a spatial location, and coexist on a maximum of two frames where they meet each other. The merging between them extends the duration of the merged supervoxel. On the other hand, a spatio-temporal neighbourhood is defined as two supervoxels coexist and touching each other in a number of frames, as illustrated in Figure 6.4(b).



(a) Temporal Neigbourhood (b) Spatio Temporal Neighborhood

Figure 6.4: Supervoxel Neighbourhood

Supervoxels and neighbouring relationships among them in an entire video create a volume adjacency graph as illustrated in Figure 6.1 (b). Each supervoxel is represented as a vertex $\mathcal{V}$, while the neighbouring relationship between them is represented as an edge $\mathcal{E}$. The similarity between a pair of neighbouring supervoxels is recorded in the edge data structure.

6.3.2 Supervoxel Similarity

Similarity is an inverse of merging cost, and this cost is equal to distance between a pair of supervoxels. Some properties such as mean colour, number of frames, start and end frames, and centroid displacement are recorded as supervoxel features. The distance between a pair of neighbouring supervoxels is mainly measured based on their colour difference and centroid displacement. Centroid displacement is the distance between the spatial centroid of the supervoxel in the start and the end frame. It represents the general motion direction and speed of the region during a period of time. The merging cost is calculated when a pair of partitions are going to merge, how far the average colour changes and how far the direction and speed of a region's movement is modified. Priority to merge is assigned to pair of supervoxels with higher similarity (e.g. lower merging cost). This allows homogeneous supervoxels to merge at an early iteration. Let consider a supervoxels $V_i = (\overline{\vec{c}}, cx1, cy1, cx2, cy2, a, nf)$ where $\overline{\vec{c}}$ is the colour average of supervoxel i; $(cx1, cy1)$ and $(cx2, cy2)$ is the centroid of the first and last frame of the supervoxel; nf is the number of frames of the supervoxel. The merging cost is calculated according to its colour and centroid distance in respect to the formula 6.1.

$$\delta_v(V_i, V_j) = \alpha.\delta_{es\vec{c}}(V_i, V_j) + \beta.\delta\vec{d}(V_i, V_j) \tag{6.1}$$

$\delta_v(V_i, V_j)$ denotes the distance between supervoxel V_i and V_j , considering the colour, size and direction of the supervoxel. Let us assume V_k is a merging result between V_i and V_j and the colour average of V_k can be calculated using equation 4.2 and substituting region/superpixel R_i and R_j with supervoxels V_i, V_j respectively.

$$\overline{\vec{c}_{V_k}} = \frac{a_{V_i} * \overline{\vec{c}_{V_i}} + a_{V_j} * \overline{\vec{c}_{V_j}}}{a_{V_i} + a_{V_j}} \tag{6.2}$$

Where a_{V_i} and a_{V_j} denotes the size of supervoxels i and j, adding those two volume size is equal to the size of the merging result between them. The $\overline{\vec{c}_{V_k}}$ denotes the new colour average if V_i and V_j are merged.

$$\delta_{es\vec{c}}(V_i, V_j) = \frac{\left(a_{V_i} * \left\|(\overline{\vec{c}_{V_i}} - \overline{\vec{c}_{V_k}})\right\|_2\right) + \left(a_{V_j} * \left\|(\overline{\vec{c}_{V_j}} - \overline{\vec{c}_{V_k}})\right\|_2\right)}{(a_{V_i} + a_{V_j})} \tag{6.3}$$

The $\delta_{es\vec{c}}(V_i, V_j)$ denotes colour Euclidean distance between V_i and V_j, considering the size and colour of the supervoxels.

$$vel_{V_i} = \frac{1}{nf_{V_i}}\sqrt{(cx2_{V_i} - cx1_{V_i})^2 + (cy2_{V_i} - cy1_{V_i})^2} \tag{6.4}$$

$$dir_{V_i} = \arctan\frac{(cy2_{V_i} - cy1_{V_i})}{cx2_{V_i} - cx1_{V_i}} \tag{6.5}$$

$$\delta\vec{d}(V_i, V_j)) = dir_{V_i}.vel_{V_i} - dir_{V_j}.vel_{V_j} \tag{6.6}$$

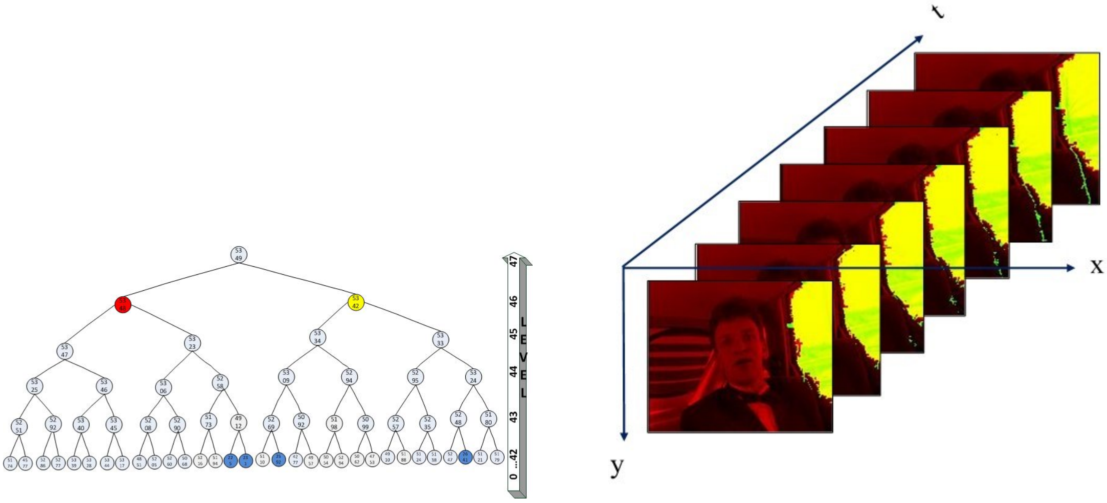
$\delta_v(V_i, V_j)$ represents the merging cost if supervoxel i and j are merged. The merging cost consists of two factors: the combinations of colour and size factor $\delta_{es\vec{c}}(V_i, V_j)$ and the movement factor $\delta(\vec{d}(V_i, V_j))$ while α and β are coefficients of colour and movement respectively. The colour and size factors are computed based on the colour model $vecc(V_i)$ difference multiplied by the size of the supervoxels $N(V_i)$. The colour difference is computed using L-2 norm. The movement speed and direction of a supervoxel is computed by the movement of the centroid of the region of a particular supervoxel at the starting frame and the intersection of the supervoxels at the terminated frame. $vel(V_i)$ denotes the movement speed of the centroid, it is calculated by using a quadratic formula of horizontal and vertical displacements of the centroid, divided by the number of frames of the supervoxels. The direction $dir(V_i)$ is computed in polar coordinates. The distance between a pair of supervoxels $\delta(\vec{d}(V_i, V_j))$ is calculated by the difference in their direction speeds.

6.3.3 Merging

The merging procedure conforms to that discussed in the general framework (Chapter 3). It is performed sequentially in regard to equation 6.1. The merging history is recorded in the binary tree structure. In every iteration, the edge's list is updated, and the upcoming selected pair is decided. The tree appearance shows no difference to the BPT of a single frame. The main difference between of the supervoxel BPT is that the node in the tree represents a series of regions in subsequent frames instead of a region in a particular frame. In this volumetric approach, every single node represents a supervoxel which is the correlated region across frames with a consistent label.

The merging operation is carried out on the Volume Adjacency Graph (VAG), which is constructed from all edges of every node. An illustration of a VAG can be seen in Figure 6.1(b). The merging order depends on the edge weight in the graph. A lower merging cost will result in a higher place in the merge order. When partitions V_i and V_j are selected for merging, the corresponding edge is discarded from the VAG, and the affected edges' weights are updated. The merging is iterated until no more edges are left in the updated volume adjacency graph. When the merging is stopped, the root node is achieved at the top of the tree, and represents the entire video. The number of iterations needed to achieve the final root node is $n-1$, where n is the number of initial partitions. The total number of nodes in the tree after merging will be $2n-1$.

The BPT represents a multilevel view of video segmentation. The top node is the root which represents an aggregate of the entire video. The lowest child nodes are the original pre-segmentation. The other nodes are a result of the selective merging process. The higher levels give a smaller number of big supervoxels (under-segmentation) while



(a) Top 6 Levels of BPT (b) Plot of 2nd Level at frame 1

Figure 6.5: Top Level BPT of Carphone Video

the lower levels give a large number of small and often meaningless over-segmented supervoxels.

Figure 6.5 shows pieces from the top-level of the BPT for the 'Carphone' video. The number of original regions from the watershed pre-segmentation result before merging was 2675 partitions. After the merging task was carried out the multi-scale segmentation is presented in a BPT consisting of 5349 nodes. An illustration of the BPT and a plot of the corresponding volumes can be seen in Figure 6.2. At the second level from the top for example, it consists of two partitions only. The node in that level is illustrated in Figure 6.5(b); It shows two spatio-temporal neighbouring partitions. As can be seen, every single node represents the correlated regions across frames (in the actual experiment, it consists of 20 frames consistent partitions, although for illustration purposes, only seven frames are drawn).

Figure 6.5(b) shows a plot of the volumes at level 2 and shows some simple semantic content in the right node(yellow), which represents the car window, while the left node (red) is not semantically meaningful as it is an under-segmented partition. Relying on the level of the tree cannot ensure that the nodes in a particular level are salient enough to be object candidates. As can be seen in Figure 6.5(b), the red partitions experienced an excessive merge, because they are under-segmented and contain more than one object. Tree representation allows detailed browsing to the lower level of the tree, whenever a node is under-segmented. The question of to what extent detailed browsing is needed and where the salient node can be discovered in a cluttered tree with thousands of nodes is not easy to answer, however.

6.4 Evolutionary analysis for Tree Simplification

Evolution analysis was introduced in [120] to identify the salient nodes in the BPT of a still image. This chapter proposes to extend the idea to a BPT of volumetric video. In the previous work, the evolution involved growing spatial regions while, in this approach, the supervoxel grows in the spatio-temporal axis. Supervoxel BPT is a historical merging archive that created over initial supervoxels refers to the merging rule set out in section 6.3.3, which respect to the merging cost formulated in equation 6.1. The merging cost considers colour and movement of supervoxel. Evolutions need to take into account both factors in order to examine a merging event considered as normal or critical. The speed and direction of a partition is simplified by calculating the angle of the supervoxel. For example, if there is a pair of supervoxels that have similar colour but move in distinct directions, they are not supposed to be merged because they probably belong to different object candidates.

The basic idea of evolution analysis is categorizing the merging task as normal or critical. Normal merging happens between two partitions with small differences, with the outcome being a new composite partition with small changes. Critical merging happens between a pair of neighbouring partitions with highly different features. Because of that the merging result will be very different compared to the original partitions.

The initial segmentation produces vast numbers (around thousands) of small supervoxels. If the number of initial partitions is n there will be n number of possible paths from the leaf to the root. Paths can be defined as $P = \{P_1, P_2, P_n\}$, and each path has a collection of nodes from the lowest leaf towards the root. Every individual path is defined as $P_i = \{nd_1, nd_2, ..., nd_l\}$ while l is the number of nodes along the path from the lowest leaf to the root, l can vary for each path. The evolution for every path P_i is defined as:

$$\gamma(k) = \alpha.\overline{\vec{c}_{nd_k}} + \beta.\vec{d}_{nd_k} \;\; k \in \{1...l\} \tag{6.7}$$

Where $\gamma(k)$ is the evolution function at level k of a certain path, $\overline{\vec{c}_{nd_k}}$ and $\vec{d}_{nd_k}$ are average colour and centroid movement of node k, α and β are constants to control the proportion of mean colour and centroid movement.

Figure 6.6 plots the evolution function against the node number (k) for the 'Carphone' video. It begins at layer 1, (the lowest child node) towards layer 23 at the top of the tree. It is observed that for the first six nodes, the colour and centroid movement remain steady at around 80. This means that for the first six layers the partitions are homogeneous, indicating the same salient object. A discontinuity is observed at $k = 6$, this is a cue that merging has occurred between heterogeneous supervoxels. A prominent discontinuity is at k = 21, indicating that it is likely that at this point the merge has occurred between dissimilar objects, as can be seen in the plot of the first frame of that particular node, and the result when they merge (see Figure 6.6).

In order to identify the critical merges, a mathematical tool is adapted from [120]. A modified first and second derivative is employed.

$$\gamma'(k) = \mid \gamma(k) - \gamma(k-1) \mid \tag{6.8}$$

$$\gamma'(k) = \mid (\alpha.\overline{\vec{c}_{nd_k}} + \beta.\vec{d}_{nd_k}) - (\alpha.\overline{\vec{c}_{nd_{k-1}}} + \beta.\vec{d}_{nd_{k-1}}) \mid k \in \{2...l\} \tag{6.9}$$

$$\gamma''(k) = \gamma(k-1) + \gamma(k+1) - 2\gamma(k) \tag{6.10}$$

$$\gamma''(k) = (\alpha.\overline{\vec{c}_{nd_{k-1}}} + \beta.\vec{d}_{nd_{k-1}}) + (\alpha.\overline{\vec{c}_{nd_{k+1}}} + \beta.\vec{d}_{nd_{k+1}}) - 2(\alpha.\overline{\vec{c}_{nd_k}} + \beta.\vec{d}_{nd_k}) \;\; k \in \{2...l\} \tag{6.11}$$

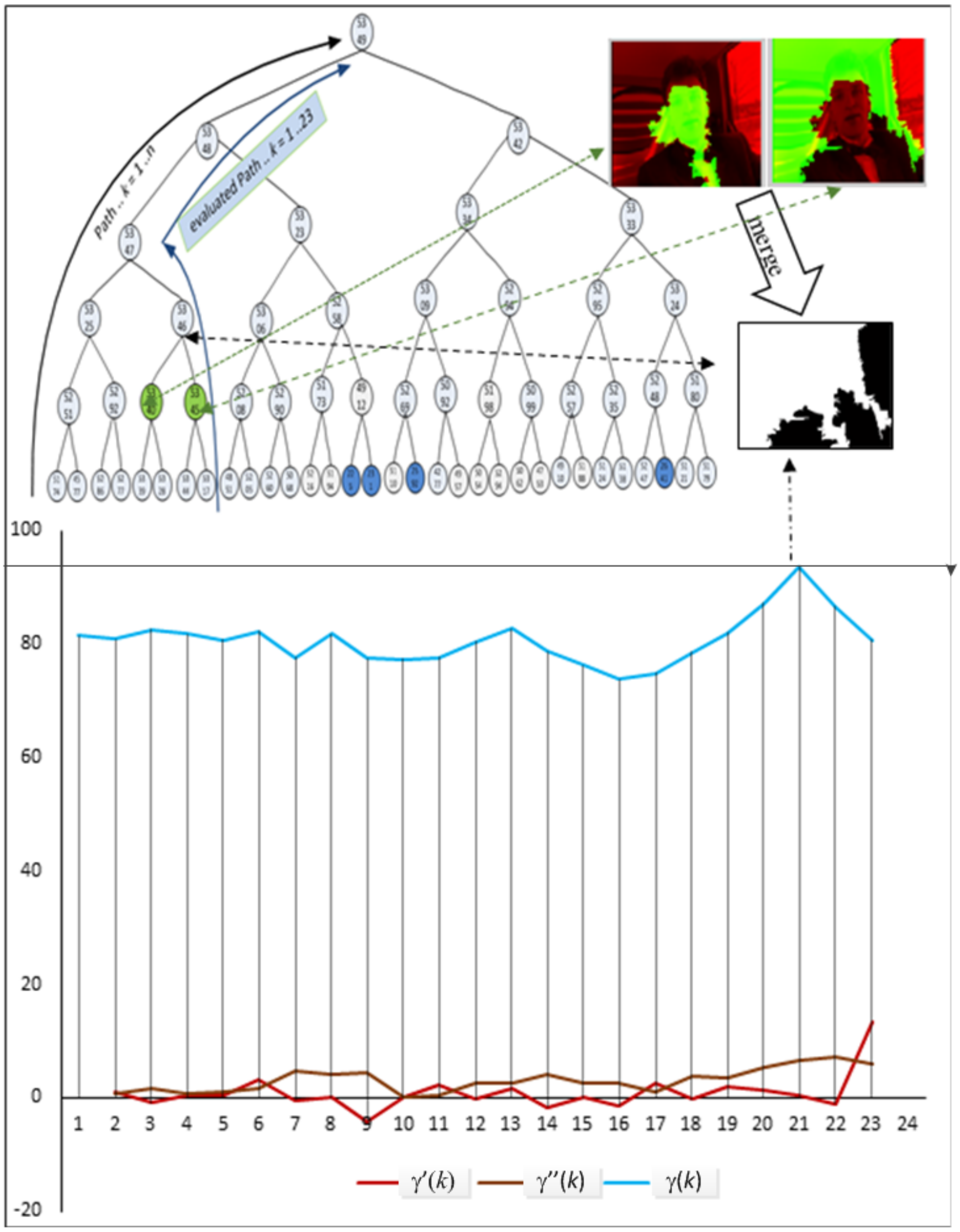
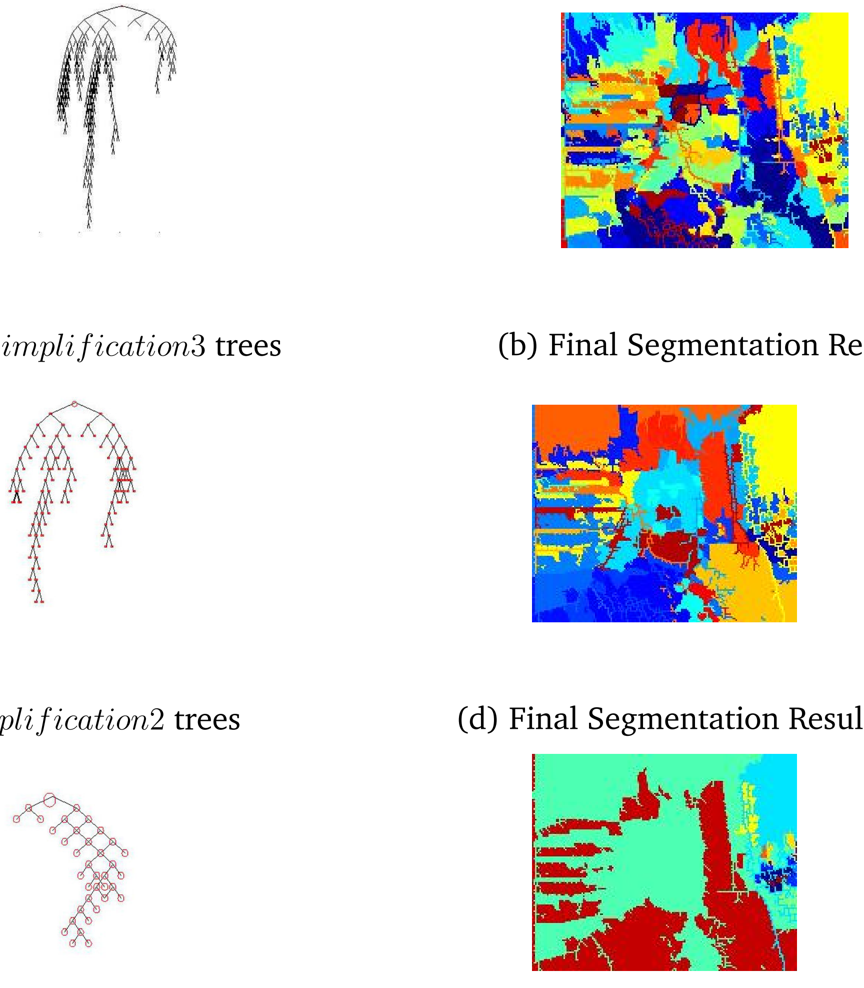


Figure 6.6: Plot of The Peaks, k = 21, Node in The BPT and The Projection in The First Frame

By substituting equation 6.7 to equation 6.11 the evolution of volume in the tree can be tracked and plotted in a graph, as shown in Figure 6.6. As can be seen in Figure 6.6, for the $\gamma(k)$ graph (the red line), it is observed that for the first 12 nodes, the colour and centroid movement remains steady at around 80. This means that for the first 12 layers the partitions are homogeneous, indicating they belong to a single object. A discontinuity is observed is at k = 13, this is a cue that merging has occurred between heterogeneous volumes. It is likely at this point that the merge has occurred between two dissimilar objects.

The difference between the parent node ($k+1$) and the current node (k), $\gamma(k+1)-\gamma(k)$ is considered to be the first derivative of $\gamma(k)$. A critical merge between left child and right child (sibling) can be identified by the value of the first derivative of $\gamma'(k)$. Figure 6.6(b) shows the evolution of the volume along a path, a value close to zero means a homogeneous merging has taken place. According to equation 6.11, the peak of the evolution is observed where the second derivative $\gamma''(k)$ crosses the zero line. Equation 6.9 calculates the magnitude of change along each path. An example of the highest peak in Figure 6.6 at $k = 21$ shows a critical merging when the face of the people starts to merge to the background.

A modified second derivative is employed to identify the maxima (peak). The maxima/peak is marked by negative value of the the second derivative function. Many peaks could be small enough to indicate that a critical merging has occurred, however. In this experiment, therefore, the algorithm will select the three highest peaks detected in each path. The selected peaks will be ordered with regard to its value. The first peak (highest) in each path is considered as the most critical merging event, where a very heterogeneous supervoxels merged. It is indicated that the pair of supervoxels most likely belonging to different objects. The set of nodes that have the highest peak during is the evolution forms a set of pruning node candidates called $simplification1$. The second and the third peak forms $simplification2$ and $simplification3$ respectively. The BPT is pruned in order to get simpler version as illustrated in Figure 6.7. With regard to the original video, an over-segmented result will occur, as can be seen in Figure 6.7(a). Conversely, if the highest peak in every path is selected, the result will be under-segmented, as seen in Figure 6.7(c).



(a) $simplification3$ trees

(b) Final Segmentation Result

(c) $simplification2$ trees

(d) Final Segmentation Result

(e) $simplification1$ trees

(f) Final Segmentation Result

Figure 6.7: Simplified BPT and its Corresponding Segmentation at Frame 1 of 'Carphone' Video

As can be seen in the result of the proposed algorithm in Figure 6.7, three levels of simplification are provided. The algorithm does not define a final single partition. It is designed to identify sets of important nodes on the tree. Each set of nodes can be scattered in any level of the original BPT. $Simplification1$ comes from the highest peak in every path, $simplification2$ and $simplification3$ from the second and third highest. These are the two requirements that have to be fulfilled by every member of simplification sets. Firstly, none of the node members of the simplification set have parent-child relationships, either directly or indirectly. Secondly, the size accumulation of all nodes in the simplification set has to be equal to the size of the video. The proposed algorithm is summarized as follows:

1. Create a BPT of a video
    (a) Pre-segment using the pre-segmentation algorithm.
    (b) Calculate the mean colour and centroid motion.
    (c) Form a Volume Adjacency Graph (VAG).
    (d) Iteratively merge the partition until the root is obtained, update VAG in every step.
2. Evolution analysis to identify simplified BPT
    (a) Prepare a variable (P) to records peaks data consisting of (path number, {node number, peak value})
    (b) Calculate the peaks for each path and record in the variable P.
    (c) For all identified peaks, select the first, second and third highest peak in every path and record the peak value and node number in the $simplification1$, $simplification2$ and $simplification3$ sets.
    (d) Select only unique nodes in these sets.

   (e) Check all the member of simplification set, if a sibling of a node do not included in simplification sets add the sibling node.

   (f) Check if there is a direct or indirect child-parent relationship, and choose the child as a member of the set and remove the parent.

3. Plot the salient video segmentation according to the peak node

   (a) Prune the BPT at the nodes in the simplification sets.

   (b) Plot the supervoxels correspond to the simplification sets.

6.5 Results and Evaluation

In order to assess the segmentation results before and after simplification, boundary recall, under-segmentation error and over-segmentation rate are measured. The ideal value of recall is one, which is achieved when all boundaries in the ground truth are completely aligned with the boundaries of the segmentation result. Under-segmentation error happens when two partitions belonging to different objects in the ground truth start to merge. A boundary of the ground truth therefore exists but cannot be found in any segmentation result. A higher under-segmentation value leads to many objects being missing from the segmentation result because they were excessively merged. Under-segmentation shows the error rate caused by excessive merging.

The simplification task begins with an over-segmentation condition. Simplification aims to detect where the tree must be pruned in order to get a simpler version of the tree and to discover salient nodes close to the expected object. In the real experiment, however, the result hardly achieves a hundred percent correct segment conforming to the ground truth. This is due to two reasons: firstly, because of the ground truth itself. The ground truth of the test video has been produced by human subjects who manually craft the boundary of each region object in every single frame. The available ground truth does not therefore reflect the absolute truth in that the segment may be interpreted differently. The second factor is the simplification algorithm. Without any prior knowledge of the object within the scene, simplification algorithms rely on the intrinsic features only. Nevertheless, comparing with ground truth is one of the most feasible ways to measure the quality of segmentation objectively. The comparison of the boundary recall of the initial pre-segmentation and simplification results is presented in Table 6.5. The comparison of under-segmentation error can be seen in Table 6.7.

In this chapter, the test video sequence from the xiph.org data set is used. Accumulated boundary recall, under-segmentation error and over-segmentation rate are evaluated. Moreover, the execution time is also compared.

6.5.1 Pre-Segmentation

Pre-segmentation aims to obtain an initial partition where the merging process would begin. Pre-segmentation produces a number of small partitions, and it is desired to avoid loss of detailed information. Watershed and SLIC are selected to prepare initial supervoxels. Figure 6.9 shows the visual pre-segmentation result of the first frame of the test video sequences using both algorithms.

The watershed is executed in a three-dimensional using 26 pixels neighbourhood. The implementation of watershed is available in the MATLAB standard. Watershed al-

Table 6.3: Watershed Pre-Segmentation Result

No	Video Name	Quantity	Size	Duration	Recall	Precision
1	bus	61048	9.723	2.336	0.939	0.364
2	container	51912	13.741	2.568	0.942	0.172
3	garden	47548	11.670	2.493	0.944	0.182
4	ice	22373	51.095	2.859	0.945	0.219
5	paris	48016	14.705	2.521	0.942	0.086
6	salesman	13197	12.809	2.424	0.940	0.180
7	Soccer	54320	11.810	2.641	0.945	0.153
8	Stefan	39999	14.748	2.436	0.943	0.143

gorithms work on gradient magnitude video as converted from its colour version. Watershed works without needing any parameter setting.

SLIC is a modified K-means introduced by [69] The implementations of SLIC supervoxels are publicly available from their website. It works with the original colour video. The SLIC algorithm inherits K-means properties, it needs a certain number of expected quantity of partition K. In the experiment, K is supplied indirectly by defining the expected size of partition. In this experiment the expected size is set to 1000. This means that the expected K is equal to the number of voxels divided by 1000. For example if we have a video with frame resolution of 200 x 100 and it has 20 frames, it consists of 400,000 voxels, if the expected size of partition is 1000, the expected K is 400,000/1000 and is equal to 400.

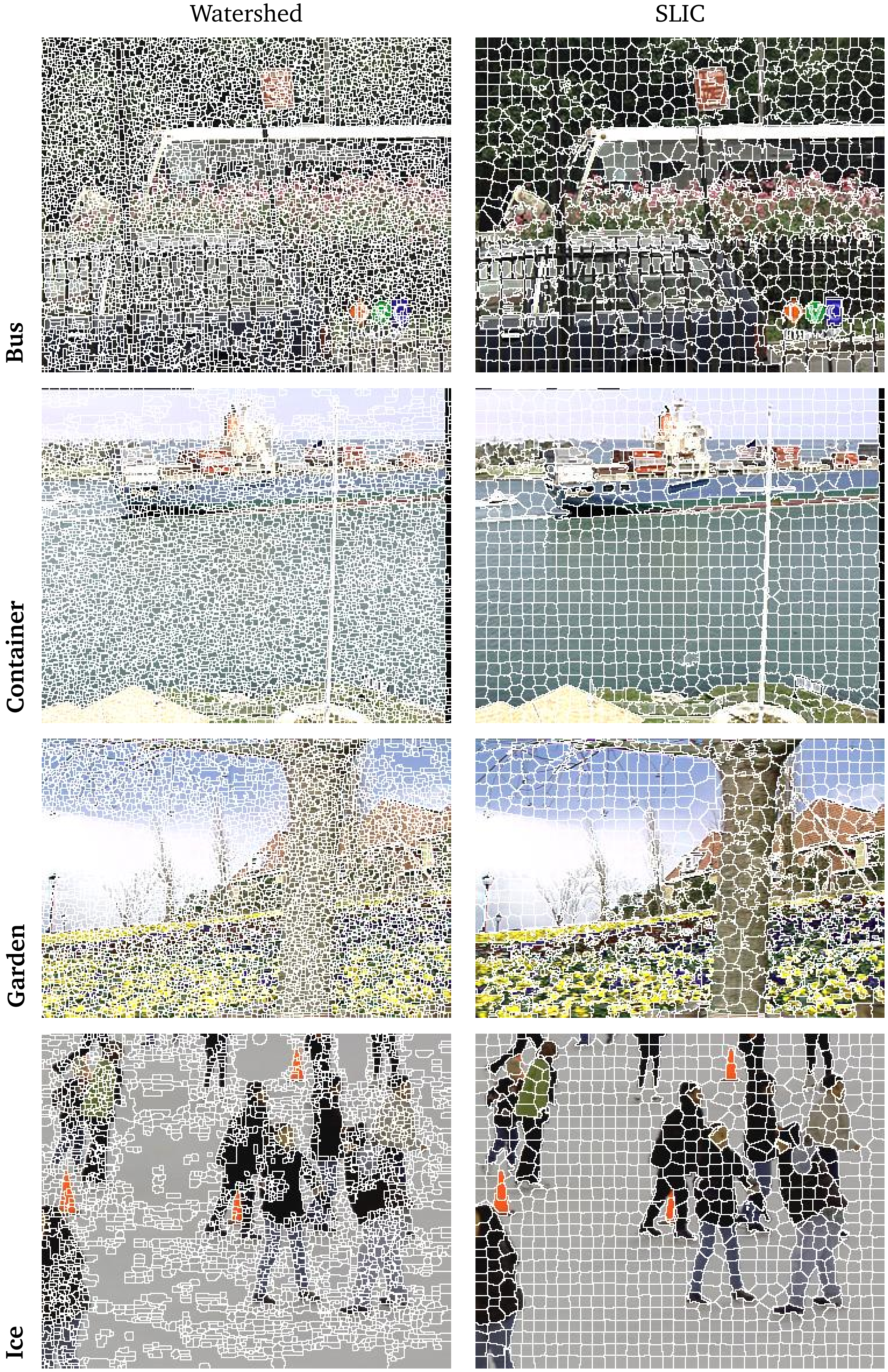


Figure 6.8: Visual Result of Pre-segmentation of The First Frame of The Video Tests

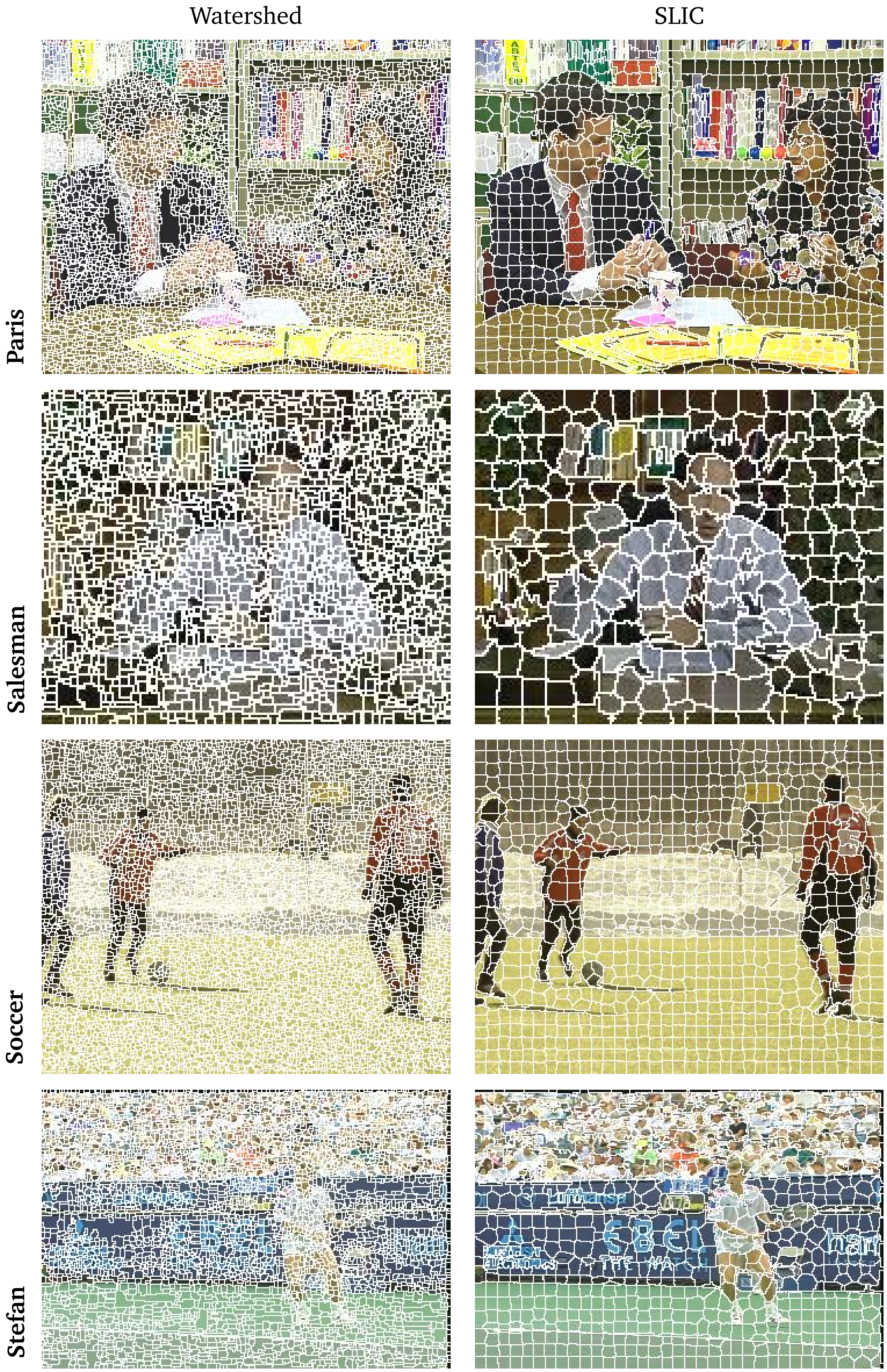


Figure 6.9: Visual Result of Pre-segmentation of The First Frame of The Video Tests

Table 6.4: SLIC Pre-Segmentation Result

No	Video Name	quantity	size	duration	recal	precision
1	bus	2642	767.416	9.388	0.929	0.422
2	container	2465	822.523	10.068	0.965	0.245
3	garden	2227	758.681	10.273	0.960	0.199
4	ice	2395	846.563	9.477	0.985	0.299
5	paris	2735	741.320	10.455	0.900	0.104
6	salesman	608	833.664	10.016	0.894	0.207
7	Soccer	2584	784.644	8.957	0.926	0.231
8	Stefan	2419	698.470	9.079	0.980	0.179

The detailed partition produced in the pre-segmentation task is evaluated in order to see the condition before the simplification task is performed. Tables 6.3 and 6.4 show the evaluation results for watershed and SLIC respectively. The quantity describes the number of partitions obtained by the pre-segmentation task. The size column shows the average size of supervoxels in the entire video.The duration column is taken from the average supervoxel life time during temporal axis. Recall and precision describe how well the boundaries of the partitions aligned to the boundaries of the ground truth. Recall describes the average recall value from all frames in the entire video. Average precision also shows in the precision column.

As can be seen in the column quantity of Table 6.3 and Table 6.4, watershed produces around twenty times more partitions compared to that of SLIC. It is directly affected by the average size and duration. The boundary recall and precision of both algorithms are not much different. SLIC gives 0.9423 on average for boundary recall values, while watershed yields slightly better at 0.9425. The value indicated that around 94 percent of ground truth boundaries are correctly aligned to the result of segmentation and six percent of the ground truth boundary were not well matched to the boundary of the segmentation results. In regard to the quality evaluation, therefore, both algorithms yield nearly the same in terms of the quality boundary recall and precision.

6.5.2 Simplification Results

Pre-segmentation produces quite high boundary recall results, but it still comprises too many partitions compared to the expected number of segments. Further tasks therefore need to be done in order to reduce the final number of segments. According to Tables 6.4 and 6.3, the quantity of partitions for the first twenty frames of video in the experiments is around forty thousand for watershed and approximately two thousand for SLIC. It is still far from the expectations in the ground truth, however, which consists of twenty partitions on average. Simplification aims to reduce the number of partitions to be as close as possible to the expected number of regions while still maintaining a good value of boundary recall and a small under-segmentation error. In other words, the objective of simplification is to maximize the boundary recall, minimize the under-segmentation error and maximize the over-segmentation rate. The over-segmentation rate is calculated as a fraction between the expected quantity of ground truth partitions and the actual number of partitions in the segmentation or simplification result. The ideal value is one, which is achieved when the number of segment equal to the quantity of expected ground truths. In our proposal, the simplification is performed in three levels and gives a different set of results.

Table 6.5: Boundary Recall and Precision After Simplification for Watershed Pre-segmentation

No	Video Name	$simplification1$		$simplification2$		$simplification3$	
		Recall	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**
1	bus	0.87589	0.3890	0.83962	0.4154	0.85878	0.3887
2	container	0.80775	0.3838	0.86931	0.3220	0.90438	0.3478
3	garden	0.86945	0.2117	0.87233	0.2066	0.91650	0.1926
4	ice	0.85432	0.3417	0.87788	0.3437	0.88279	0.3485
5	paris	0.81590	0.1005	0.87992	0.1019	0.86842	0.1029
6	salesman	0.87662	0.1771	0.89127	0.1755	0.87717	0.1742
7	Soccer	0.86293	0.2397	0.86106	0.2016	0.85610	0.2063
8	Stefan	0.91892	0.1499	0.91717	0.1546	0.91976	0.1534

Comparing Table 6.5 and Table 6.6, the watershed algorithm generally produces a better rate of boundary recall compared to that of SLIC. At the same time watershed produce a worse over-segmentation rate, as presented in Table 6.7. Figure 6.10 compares the result of the $simplification3$ with the boundary recall produced by BPT on watershed and SLIC. The watershed in general produces better boundary recall but gives more partitions, while SLIC tends to slightly lost the boundary recall but produces a smaller number of partitions. SLIC, therefore, has a better over-segmentation rate, which means that every partition in the SLIC occupies a wider area of the ground truth.

Table 6.6: Boundary Recall and Precision After Simplification for SLIC Pre-segmentation

No	Video Name	$simplification1$		$simplification2$		$simplification3$	
		Recall	Precision	Recall	Precision	Recall	Precision
1	bus	0.66464	0.44638	0.75042	0.47900	0.71234	0.46083
2	container	0.63684	0.43098	0.63273	0.44946	0.71324	0.34534
3	garden	0.78030	0.23310	0.83584	0.23010	0.86257	0.21876
4	ice	0.81037	0.71260	0.82524	0.68157	0.92614	0.47109
5	paris	0.70803	0.14093	0.72510	0.14351	0.75995	0.12477
6	salesman	0.60898	0.19501	0.70264	0.20793	0.68151	0.20148
7	Soccer	0.67323	0.44592	0.68289	0.46326	0.68611	0.45379
8	Stefan	0.80817	0.27324	0.80990	0.26985	0.87564	0.26502

Table 6.8: Under-segmentation Error Before and After Simplification for Watershed and SLIC Pre-Segmentation

No	Video Name	**Watershed**				**SLIC**			
		Preseg	Simp 1	Simp 2	Simp3	Preseg	Simp 1	Simp 2	Simp3
1	bus	0.709	0.832	0.825	0.750	0.310	0.695	0.798	0.803
2	container	0.513	0.380	0.432	0.414	0.084	0.562	0.554	0.497
3	garden	0.661	0.599	0.603	0.592	0.121	0.744	0.709	0.558
4	ice	0.335	0.390	0.389	0.373	0.069	0.164	0.161	0.104
5	paris	0.240	0.303	0.286	0.306	0.066	0.269	0.228	0.221
6	salesman	0.387	0.484	0.442	0.438	0.153	0.411	0.371	0.446
7	Soccer	0.669	0.754	0.750	0.701	0.077	0.443	0.437	0.439
8	Stefan	0.640	0.547	0.522	0.517	0.075	0.364	0.362	0.276
average		0.519	0.536	0.531	0.511	0.119	0.456	0.452	0.418

Table 6.7: Over-segmentation Rates Before and After Simplification for Watershed and SLIC Pre-Segmentation

No	Video	Watershed				SLIC			
		Pre	Simp1	Simp2	Simp3	Pre	Simp1	Simp2	Simp3
1	bus	0.0010	0.0231	0.1407	0.0512	0.0062	0.065	0.098	0.090
2	container	0.0012	0.4246	0.0786	0.0608	0.0066	0.242	0.327	0.119
3	garden	0.0008	0.1401	0.0817	0.0243	0.0045	0.197	0.067	0.032
4	ice	0.0015	0.0406	0.0537	0.0419	0.0045	0.164	0.114	0.030
5	paris	0.0008	0.0795	0.0169	0.0241	0.0034	0.097	0.070	0.025
6	salesman	0.0024	0.1223	0.0389	0.0359	0.0135	0.185	0.082	0.052
7	Soccer	0.0009	0.0757	0.0367	0.0511	0.0059	0.193	0.147	0.119
8	Stefan	0.0011	0.0418	0.0633	0.0339	0.0054	0.499	0.138	0.084

The conditions before and after simplification can be observed in Table 6.3, Table 6.3, Table 6.5 and Table 6.6. Simplification is performed by merging a number of pairwise partitions, with the associated risk of excessive merging. Excessive merging between two partitions causes the boundary separated two objects in the ground truth to dissolve. It affects the declining value of boundary recall. Although during simplification, excessive merging was avoided by identifying the critical merging, there is still the potential for it to occur. Particularly when a pair of partitions have low contrast, although they belong to different objects, the feature distances between them are small. In such circumstances, the boundary recall of the simplification result tends to drop compared to the pre-segmentation result. Moreover, at the simplification of lower peaks (i.e. $simplification2$

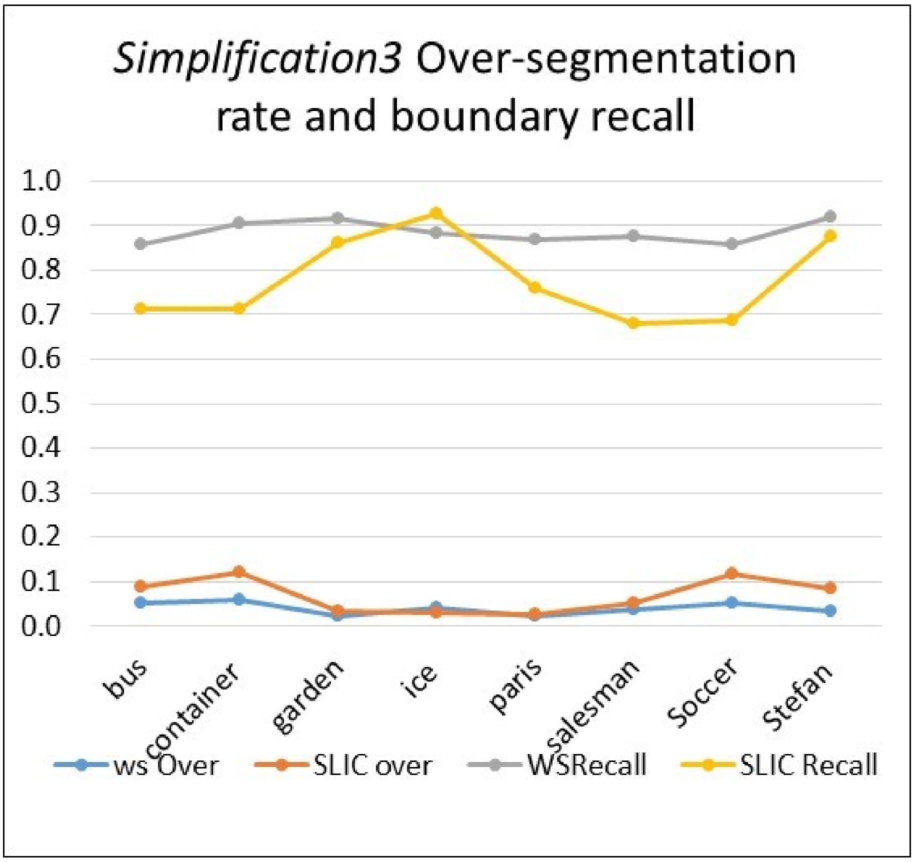


Figure 6.10: Comparison of The Result of $simplification3$ Between Boundary Recall and Over- segmentation Rate

and $simplification3$) produces a smaller rate of under-segmentation, but at the same time it shows more partitions involved and therefore worse over-segmentation rate. In Figure 6.11, the results of simplification can be visually observed. Because of space limitation, only the first frame of three video clips are figured out while the complete results for all frames are available in Appendix B for the 'Soccer' video. The remaining results are available in the digital format.

In Table 6.8 the under-segmentation errors are presented. In general, the under-segmentation error before simplification is smaller than the condition after simplification.

A small value is expected in this parameter. SLIC gives better results in terms of under-segmentation error.

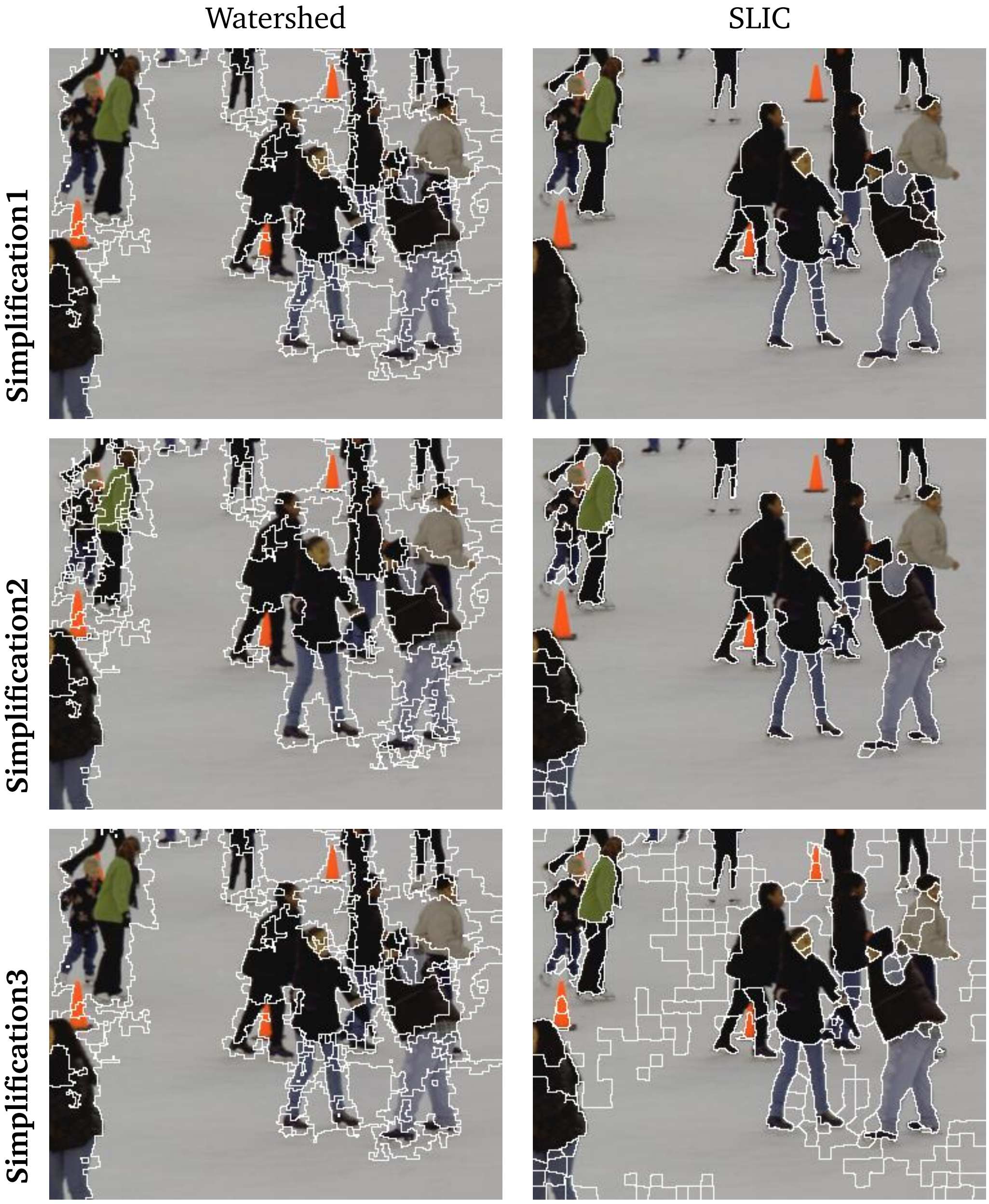


Figure 6.11: Visual Comparison of Simplification 1,2 and 3 of The First Frame 'Ice' Video Test Using Watershed Versus SLIC Pre-segmentation

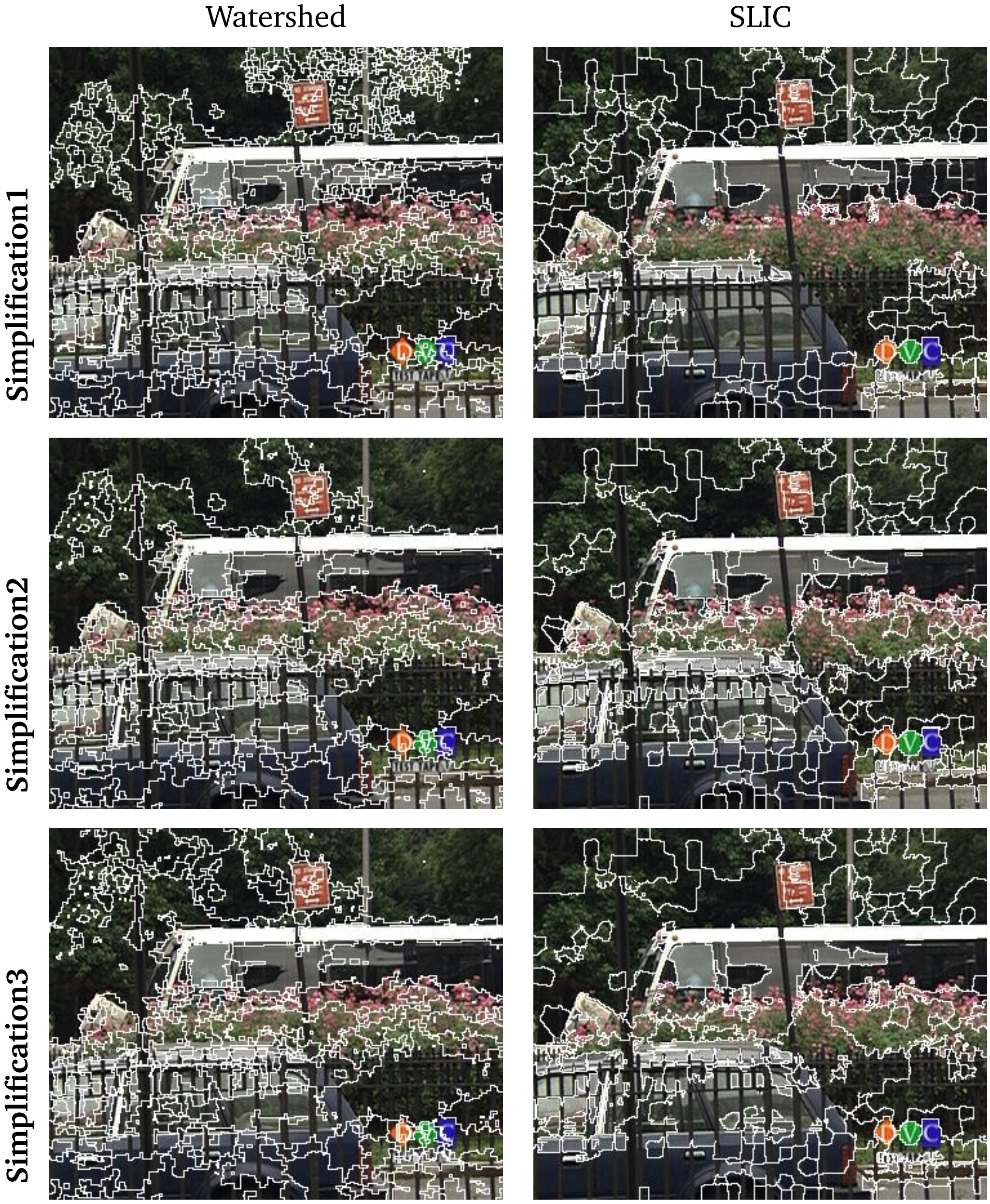


Figure 6.12: Visual Comparison of Simplification 1,2 and 3 of The First Frame of ‘Bus’ Video Test Using Watershed Versus SLIC Pre-segmentation

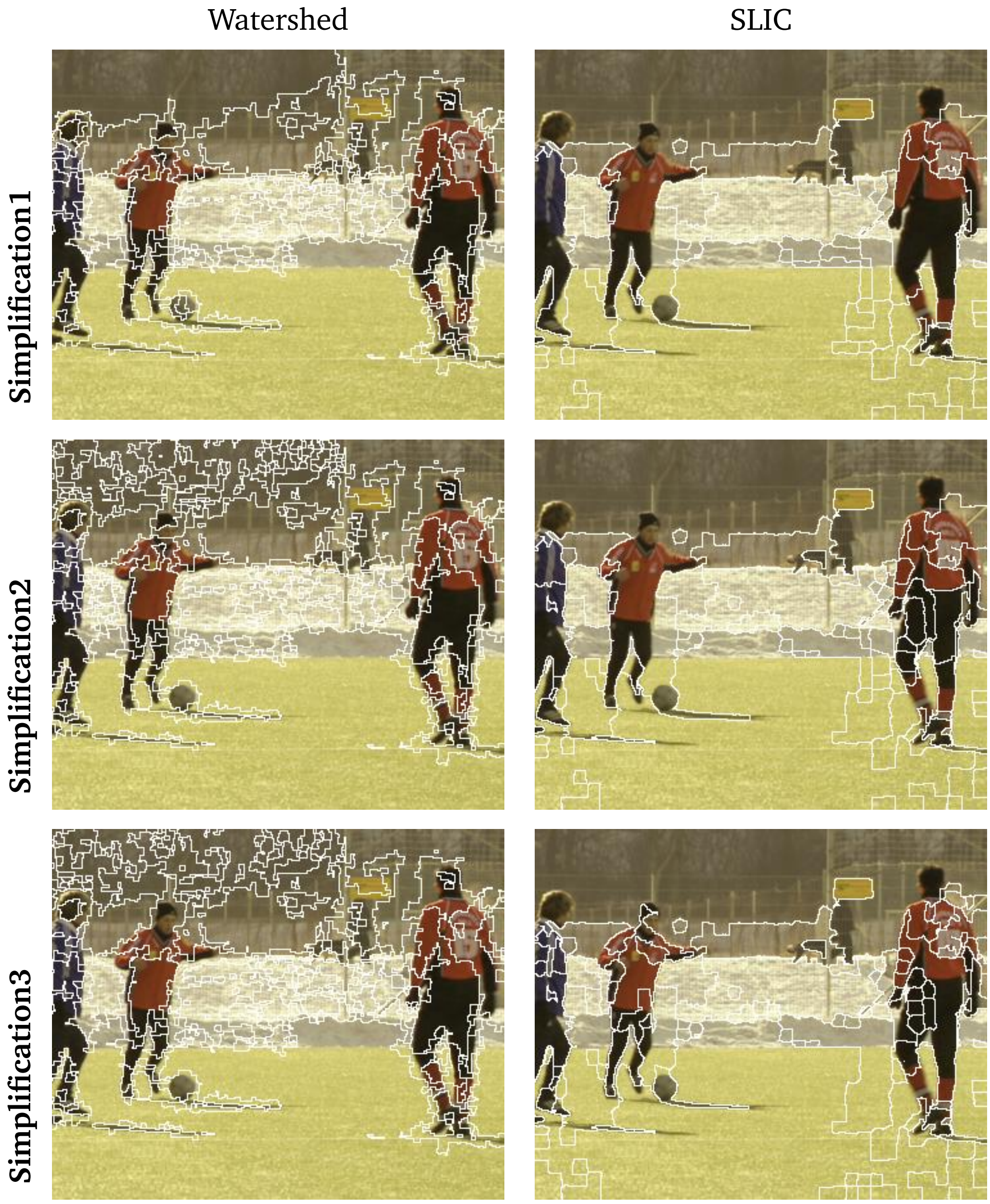


Figure 6.13: Visual Comparison of Simplification 1,2 and 3 of The First Frame of 'Soccer' Video Test Using Watershed Versus SLIC Pre-segmentation

As can be seen in Figures 6.11, 6.12 and 6.13 the $simpification2$ and $simplification3$ shows gives a lot more details segmentation compared to the $simpification1$. The tendency for over-segmentation error can be clearly seen in the $simpification2$ and $simpification1$ result. For example, in Figure 6.13 on BPT of SLIC, the ball on the soccer video has undergone excessive merging and was merged to background in $simplification2$, compared to $simplification3$ where it still perfectly segmented. Consequently, the score for the under-segmentation error gets higher and the boundary recall becomes lower in the $simplification2$. Loss of detailed information is an undesirable property of segmentation; therefore, it must be avoided.

6.5.3 Running Time

Figure 6.14 shows the total execution time for all tasks needed to segment and simplifies the tree for every frame of the test video sequence. The detailed execution time for each task can be seen in Table 6.9. The running time for segmentation is insignificant compared to the entire running time. It can be understood that the BPT preparation and simplification in the watershed take a much longer time, due to the number of iterations. The number of iterations is directly proportional to the number of the pre-segmentation results. Consequently, watershed algorithms take longer to complete all the tasks.

Watershed takes far longer to prepare a complete binary partition tree and simplification compared to SLIC. It is directly affected by the number of iterations needed to achieve the final root node. The number of iterations in preparing BPT was twice the quantity of partitions in the pre-segmentation result. The iteration in preparing BPT for the partitions yielded by watershed is much higher than that of SLIC, therefore. Although the pre-segmentation needs longer in SLIC, the rest of the tasks needs a smaller amount of time. In total from the pre-segmentation to the simplification for all three peaks shows that SLIC is computationally lighter than watershed.

The running time presented in this thesis was executed using a PC with specification: Intel Core i7 2.10 GHz processor, 8GB memory with Windows 7 Home Basic operating systems.

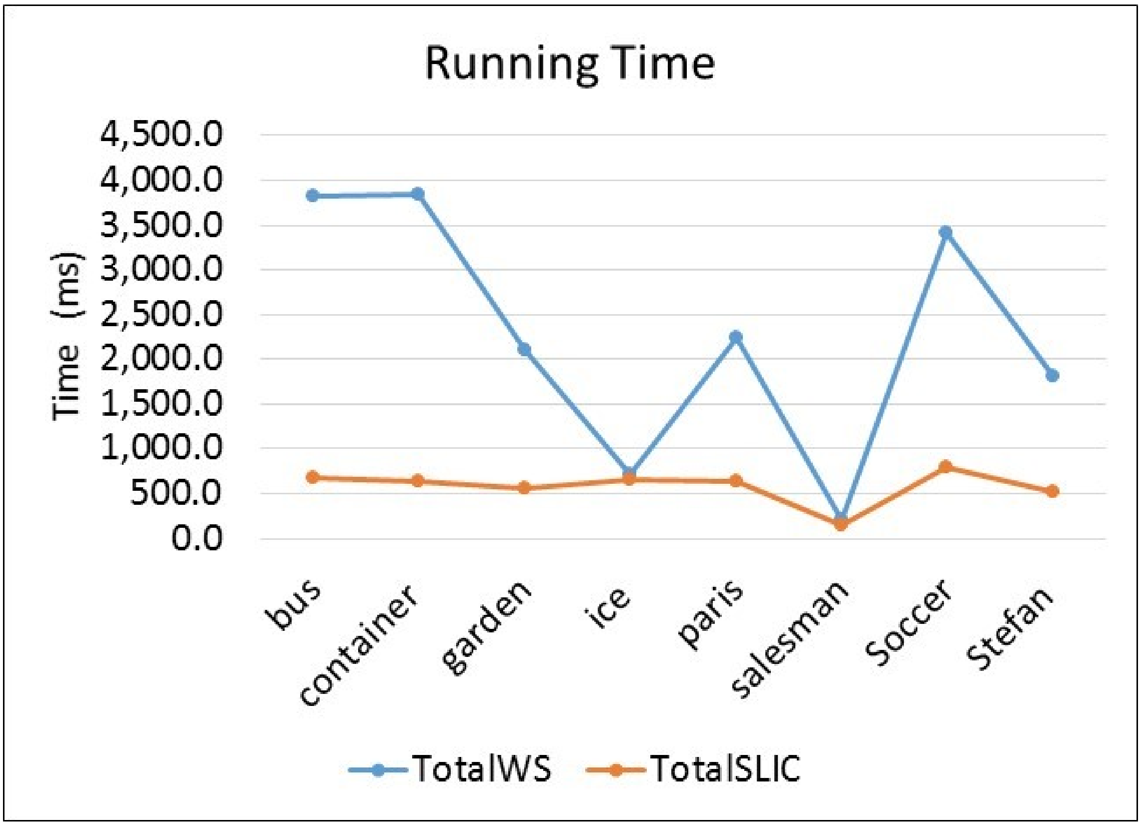


Figure 6.14: Total Running Time Comparison in Milliseconds

The total running time shows that SLIC generally needs smaller amounts of time to complete all tasks. In terms of pre-segmentation, however, watershed outperforms SLIC. Watershed needs around 2.5 milliseconds for each frame to produce a partition, while SLIC takes around 500 milliseconds (0.5 second) to perform pre-segmentation. Conversely, preparations for the BPT and the simplification need a much shorter duration on

Table 6.9: Running Time in Milliseconds (for 20 Frames)

No	Video	Watershed				SLIC			
		Pre	BPT	Simp	Total	Pre	BPT	Simp	Total
1	bus	3.4	2,668.7	1144.6	3,816.7	571	53.3	61.5	685.7
2	container	2.9	2,606.3	1242.9	3,852.0	566.3	26	36.4	628.6
3	garden	2.4	1,372.6	727.1	2,102.1	497.1	27.7	26.4	551.2
4	ice	2.6	523.8	190.2	716.6	575.7	29.5	45.6	650.8
5	paris	3	1,499.1	738.3	2,240.5	568.9	23.7	45.3	637.9
6	salesman	0.6	139.6	58.7	198.9	132.1	3.1	3.8	139.1
7	Soccer	3	2,129.6	1278.0	3,410.6	732.7	19.7	45.2	797.6
8	Stefan	2.5	1,271.5	527.7	1,801.7	401.3	49.1	64.7	515.0

SLIC. BPT preparation and simplification depends on how many iterations are needed. Because SLIC generates a fewer number of initial supervoxels, the number of iterations in forming the BPT and simplification is fewer, meaning that, the speeds of the two remaining tasks are faster.

6.6 Record supervoxel BPT and Simplification

The pre-segmentation and simplification results that have been discussed are designed to support the metadata. The result is recorded into the database.The volumetric approach produces a number of volumes/supervoxels. A supervoxel is recorded in the $supervoxel$ table, while the neighbourhood relationship is recorded in the $svEdge$ table.

The neighbourhood relationship is classified into spatio-temporal and temporal edges and will be recorded in $type$ column in the $svEdge$ table as metadata designed in figure 3.12.

6.7 Conclusion

As suggested in Chapter 4, the proportional Euclidean similarity measure is light enough in terms of computation and maintains the quality of the result, as indicated by the high boundary recall rate. For that reason, it is implemented in this chapter. The simplification algorithm prepares three levels of simplification namely $simplification1$, $simplification2$ and $simplification3$. In order to take into account the motion, the weighted speed and direction of the supervoxels are added to the final similarity measure. Two pre-segmentation algorithms are evaluated in relation to their production of the initial supervoxels. The K parameter of SLIC is indirectly set by setting size of the window search at 1000. The experiments try to reveal the preferred technique to provide reliable output. The expected output is a simpler version of the tree, and thus a simpler segmentation of the video that still keeps the boundary of the objects. It is measured by comparing the segmentation result to the ground truth across frames.

Pre-segmentation is carried out by watershed using 26 pixel neighbourhoods in order to work with three-dimensional space time matrices. The result is highly over-segmented with 42,000 partitions yielded in 20 frames of video tests (see Table 6.3). The duration of the supervoxels relatively short in less than three frames out of a total of twenty one available. The total number of nodes in a BPT will doubled of the number of the initial supervoxels (n). The simplification process will look for the critical merging events on the forty two thousand paths along the tree.

On the other hand, BPT constructed on SLIC initial pre-segmentation is simpler. The initial supervoxels is around 2000 with an average of nine frames out of 20 available (see Table 6.4). Compare to the BPT constructed from initial supervoxels yielded by SLIC, which started with around 2000 partitions with around nine frame duration on average (See Table 6.2) with similar video tests. The total number of nodes in entire BPT before simplification consists of around 4000. The simplification process will look for the critical merging events on the 2000 paths along the tree. It is obvious why BPT creation and simplification through SLIC is much faster than through watershed. According to Figure 6.14 for almost all the video tests the entire process is faster in SLIC than in watershed.

Quality of output is expected to be closer to the available ground truth. This is measured by boundary recall, under-segmentation error and over-segmentation rate. According to Tables 6.6 and 6.7 the boundary recall of $simplification1$, $simplification$ and $simplification3$ in watershed are superior to that in SLIC. The precision of SLIC in $simplification1$, $simplification$ and $simplification3$ is generally better than that in watershed. This suggests that in watershed, the simplification results keep the unnecessary boundary that doest not exist in the ground truth. This conforms with Table 6.7 in respect to the over-segmentation rates. On average, SLIC gives a better over-segmentation rate. This confirms that the simplification through SLIC provides a close number of segments as those in the ground truth. For example, $simplification1$ of 'Ice' video in Table 6.7 shows SLIC at 0.164 while in watershed it is 0.091. This means that in $simplification1$

in SLIC, each segment in the output occupies 0.164 of the ground truth while in the watershed it take up only 0.091 part. Considering the boundary region recall, precision, over-segmentation rate and speed, the simplification of supervoxels BPT in SLIC performs better than that in watershed.

Chapter 7

Region Based Metadata

7.1 Introduction

This chapter is dedicated to discussing our proposal for regional based metadata and spatio temporal queries. The metadata is generated from the simplification result discussed in previous chapters. The metadata is designed to convey visual content in more human-like textual data rather than recording meaningless intrinsic data such as RGB code, pixel position in numeric x, y and z or motion vector. The metadata is provided in order to allow the proposed prototype to answer information enquiries regarding the video content.

The term metadata has been widely used in relation to videos and images to refer to some device embedded technical data such as device identity, date time and location(geo-tagging). MPEG-7 also enables object annotation in a video, although it does not have any standard for the automatic extraction of audiovisual (AV) descriptions/features, and or for its search mechanism [9]. We intend to record textual information regarding the video content in a relational database (RDBMS) in order to allow an adaptation of the standard query language to interact with the metadata.

Currently, there are many approaches to content querying. For example, [109] use query by example (QBE) and query by feature to express the information requests to the multimedia database system. Another approach [101] uses object SQL to reveal the data from a video database. In our example, a standard query language is adapted with a number of extensions in order to deal with spatio temporal data. Some additional keywords are therefore introduced, and special functions are prepared to empower the RDBMS machine to be able to carry out the spatio-temporal logic.

Preceding chapters have proposed many ways to harvest the object candidates from the video content. Some of the candidates may be close to the real object as represented in the available ground truth, however, the rest may still be meaningless. The object candidates in the various levels prepared in the binary partition tree would be turned into metadata.

The designed metadata conveys region statistics for multiple frames in a video. Region features include colour, area, location in a frame, and its presence in the frames, together with child-parent distance their relative colour compared to their neighbours. The content of the metadata is listed below:

1. Volume/supervoxel identity

2. Node level in the binary partition tree

3. Node statistics(colour average, size, centroid)

4. Nodes distance to its parent and child in the tree
5. Nodes distance to its neighbours in spatio-temporal space
6. Supervoxels in temporal domain (how long it is alive, how far it is moving)

The rest of this chapter will discuss a number of important issues such as: the database design, the function to convert nodes in the BPT into video metadata, the design of the spatio-temporal query and the function to deal with it, the prototype of the multimedia database management system to manage the video content based on a binary partition tree and the evaluation of the result for a number of spatio temporal operations.

7.2 Metadata Modelling

Metadata is designed to record nodes and the neighbouring relationships of the BPT structures. In addition to metadata, the original video and label map of the video are recorded. A pre-segmentation task prepares the label map which reflects the initial supervoxels. Metadata permits a content information request to be processed without accessing the original video. BPT structures allow multi-scale information processing, for example, if an information request fails to get the answer in the high level tree, the searching mechanism can track down to the lower level (i.e. access more detail). The searching is considered to have failed whenever it reaches the lowest level of the tree and the desired information has not been found. Figure 7.1 illustrates the relationship among query, metadata and the binary partition tree. Searching is performed using the metadata which follows the BPT structure, the video and the original label map need to be accessed in order to display the result visually.

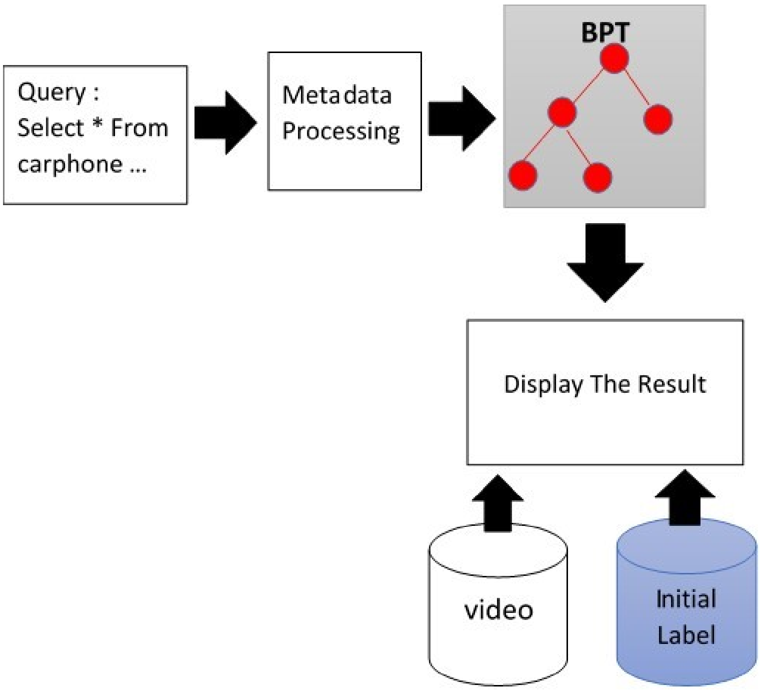


Figure 7.1: System Architecture

The BPT generator and simplification tasks produce a number of nodes in any level from the most detailed and small up to the entire global content. The information of how far a node stands out from its neighbours is also provided as a result of critical merging identifications. Although some salient nodes were therefore identified, the information obtained is not user friendly. For example, the colour of the region is recorded in numeric RGB values instead of its colour name, and the displacement of the centroid which is considered as the region motion speed and direction is also numerically recorded. In order to prepare for a term in a human language, an interface needs to be provided with the information necessary to convert that numeric value into human terminology. The original value is still kept in the metadata in order to serve a more accurate information request from another machine processing task such as robot vision.

The main structure of video metadata consists of the reference table, original video identity, node and neighbourhood. Reference tables allow conversion from numerical values into a textual term of colour and direction of movement. Video tables are provided in order to record the identity of the video. The processing results are documented in some entities such as supervoxel, superpixel and the edges. The structure of the database can be seen in Chapter 3, Figure 3.12 and the detailed context of each database table is described in Table 7.1.

Table 7.1: Explanation of The Database Table Usage

No	TableName	Usage	Data Producer	Data Usage
1	video	Record the identity of the video, location of the original video and the original label map produced by pre segmentation algorithm	spatial approach, volumetric approach	answer user query
2	superPixel	Record the frame/single image pre-segmentation and simplification	spatial approach, projection of volumetric approach	answer spatial user query
3	spatialEdge	Record the region/supervoxel neighbourhood, control the merging order for spatial approaches	spatial segmentation, spatial merging	answer spatial neighbouring query

Continued on next page

Table 7.1 – *Continued from previous page*

No	TableName	Usage	Data Producer	Data Usage
4	temporal Edge	Record the inter frame region neighbourhood	temporal relationship establishment task	answer user temporal query
5	supervoxel	Record the volumetric multiframe pre-segmentation and its simplification	supervoxel segmentation and simplification	answer spatio temporal query
6	svEdge	Record the spatiotemporal neighbourhood relationship	supervoxel segmentation, super voxel merging and supervoxel simplification	answer spatio temporal query
7	refColour	Record colour reference and the textual term of colour	input	answer textual colour term query
8	refDirection	Record the direction reference and textual term of direction	input	answer textual direction term query

In the implementation, this database is stored in MySQL. Standard query processing of the data is handled by this database management system machine.

7.3 Conversion of Binary Partition Tree into Metadata

Metadata includes all the data processing results in the pre-segmentation and simplification tasks discussed in the previous chapters. Some conversions need to be done in order to provide human terminology for some features. Colour conversion is carried out according to the colour value and colour space referring to the reference table. Colour conversion needs to be done when the conversion of BPT to metadata is performed. Direction, either in motion or neighbourhood (spatial edge, does not need to be converted into textual value: the original value will be kept. Conversion only needs to be performed when a query request is accepted with a direction specific keyword. In order to perform a direction query, the algorithm will accept the desired direction from the query and convert it to a numerical value according to a reference table. The answer to the direction query will be found in the data according to the numerical value calculated.

7.3.1 Direction Mapping

In order to provide the metadata in a readable form for humans, a query related to direction does not need to provide the exact angle and speed of the direction. The human language of direction is accepted by a query processor. The direction query processor will translate the request into appropriate numeric data in the actual metadata. The references for direction can be seen in Figure 7.2, while the conversion from the angle to textual direction can be seen in Table 7.2.

The direction conversion table allows different references to be applied if desired. The direction would be recorded as the original numerical value. The direction reference is

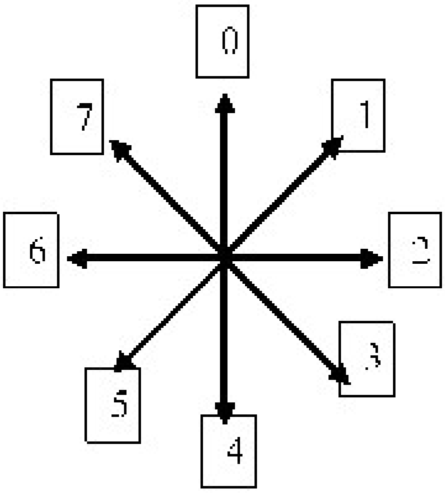


Figure 7.2: Reference Direction

Table 7.2: Conversion table from angle to semantic

code	Semantic	Semantic	Radian angle	Degree	min	Max
0	upper	North	1.57	90	67.5	112.5
1	upper right	north east	0.785	45	22.5	67.5
2	right	east	0	360	337.5	22.5
3	under right	south east	-0.785	315	292.5	337.5
4	under	south	-1.570	270	247.5	292.5
5	under left	south west	-2.356	225	202.5	247.5
6	left	west	3.14	180	157.5	202.5
7	upper left	north west	2.356	135	112.5	157.5

employed in order to answer the query. For instance, when the query state 'motion = left', the algorithm refers to the table $refDirection$ and reads all the nodes adequate the direction value. The maximum and minimum value of each direction is allowed to deviate for each angle. For example, the 'left' direction is 180 degrees, but it is allowed between 157.5 to 202.5.

Algorithm translateDirection Assign Textual Colour for all Nodes

```
procedure TRANSDIRECTION($text_direction$)            ▷ Translate direction query
    $Direction \leftarrow refDirection(directionName = text_direction)$
    $nodesResult \leftarrow Nodes(nodes.directionbetween(Direction.min)to(Direction.max))$
    return $i$
end procedure
```

7.3.2 Colour Textual Mapping

In order to simplify the colour semantic, a simple 16 colour range is mapped to the RGB colour code. This is usually used in web save colour code (source HTML 4.01 Specification section 6.5 "Colors" from W3.org). This can be extended to other colour sets for a more complete semantic such as a 256 colour set. The reference 16-colour table can be seen in table 7.3. Colour codes for each node are assigned when writing the particular node to the table. The colour code is assigned by calculating the nearest distance of every node colour to the particular colour in the $refColour$ table.

Table 7.3: Conversion table for numerical colour to semantic

Name	Red	Green	Blue	Semantic
White	100%	100%	100%	15 (white)
Silver	75%	75%	75%	7 (light gray)
Gray	50%	50%	50%	8 (dark gray)
Black	0%	0%	0%	0 (black)
Red	100%	0%	0%	12 (high red)
Maroon	50%	0%	0%	4 (low red)
Yellow	100%	100%	0%	14 (yellow)
Olive	50%	50%	0%	6 (brown)
Lime	0%	100%	0%	10 (high green); green
Green	0%	50%	0%	2 (low green)
Aqua	0%	100%	100%	11 (high cyan); cyan
Teal	0%	50%	50%	3 (low cyan)
Blue	0%	0%	100%	9 (high blue)
Navy	0%	0%	50%	1 (low blue)
Fuchsia	100%	0%	100%	13 (high magenta); magenta
Purple	50%	0%	50%	5 (low magenta)

Algorithm assigncolourDB Assign Textual Colour for all Nodes

```
procedure AssignColour($nodes$,$colourlist$,$destination$)    ▷ Assign Textual colour
    while $i \neq endofnodes$ do    ▷ Read all nodes
        $c_1 \leftarrow nodes(i).c_1$
        $c_2 \leftarrow nodes(i).c_2$
        $c_3 \leftarrow nodes(i).c_3$
        $colourCode \leftarrow getNearestColour(c_1, c_2, c_3, colourlist)$
        $insertinto\ destination\ values\ (nodes(i), colourCode)$
        $i++$
    end while
    return $i$
end procedure
```

7.4 Multimedia Database Architecture

7.4.1 Extended Spatio Temporal Query Keywords

Standard query language has no ability to deal with the spatio temporal information in the tree representation. Extended SQL is designed to reveal data from a binary partition tree of a video sequence. The structure of the metadata has been discussed in Chapter 3. The operations to generate metadata has been discussed in Chapters 4,5 and 6. An extended standard query language is proposed in order to deal with hierarchical spatio-temporal segmented data. The standard query language (SQL) does not support spatio temporal requests, and therefore, several special keywords are introduced. In order to handle an extra keyword, queries are processed in two separate stages: in the first stage, an additional function is embedded in front of the normal MySQL machine in order to decode some extended keywords. After it has been decoded into standard query, the request is submitted to the database machine. The proposed extended keywords are listed below:

- motion

    Sleft

    Sright

    up

    down

    Static

    dynamic

- colour

    single colour

    multiple colour with 'OR' operator

- temporal

    before

    after

- Saliency

    TopSaliency

7.4.2 Spatio Temporal Query Pre-Processing

7.4.2.1 Syntax Checking

A standard query processing machine could not carry out the spatio temporal request. A query supplied by the user has to be pre-processed before being passed to the standard DBMS machine. The first task of query pre-processing is ensuring that the query is valid. Due to the desired retrieval function, in the prototype, the primary keyword adapted is limited to SELECT statements related keywords. The second task is to identify the extended keyword, decode it and assign the processing into a specific function for each keyword. The functions perform decoding tasks for every extended keyword, from the particular keyword into normal SQL syntax. The third task divides the query into sub queries in SQL in order for them to be executed by MySQL machine. The results of query decoding are a number of sub queries that can be executed independently. The result of each sub query is operated to obtain the final result. The row sets as a result of individual sub queries are operated by a logical operator. The result can be a list of partitions which meet the criteria. In order to display the result, a list of nodes will be displayed according to the original video and original label map, as illustrated in Figure 7.1. The pseudo code of the execution strategy and syntax checking is defined in the algorithm SQL Syntax Checking.

Because the option of the keyword and the object has already been defined in the metadata structure, the probable query syntax combination is limited. The algorithm to check syntax and grammar is defined in pseudo code SQL Syntax Checking below.

Algorithm SQL Syntax Checking

```
procedure CheckSQL($query$)                                   ▷ Check the Query
    $keywords \leftarrow breakdown(query)$
    if $keyword[1] ==' SELECT'$ then
        $indexFROMKey \leftarrow find(keywords,' FROM')$
        $column \leftarrow keywords[2 : indexFROMKey]$
        $datasource \leftarrow keywords[indexFROMKey + 1]$
        $Logicalop \leftarrow find(keywords[indexFromKey + 2 : end], logicalop)$
        $ExtQuery \leftarrow find(keywords[indexFromKey + 2 : end], ExtKeywords)$
        $i \leftarrow 1$
        $StandarQuery \leftarrow (keyword[1] + column +' FROM' + datasource$
        while $i < Length(ExtQuery)$ do
            $StandarQuery \leftarrow StandarQuery + decodeExtQuery(ExtQuery(i))$
            $increment(i)$
        end while
        $Rows \leftarrow execute(StandardQuery)$
    else
        need ‘SELECT’ at the beginning
    end if
    return $Rowset$
end procedure
```

7.4.2.2 Pre-Processing Function

There are a number of extension keywords and related logical operators. Each keyword is mapped to a particular function in order to be dealt with. The functions are in charge of decoding the keywords into a sentence in a standard query language in order to be executed by RDBMS using the available metadata. In order to receive the query, after checking the syntax operator and the arguments are identified. Each operator and argument are passed to the special function in order to decode the syntax into a standard query language syntax in order to be sent to the RDBMS machine for execution. There are three functions prepared to deal with each syntax which are:

- motion decoding function
- direction decoding function
- neighbourhood decoding function

7.4.3 Spatio Temporal Query Processing

The metadata is a direct reflection of the binary tree structure. By default, searching is started from the root and iteratively progressed to the lower level. Every time the level is lowered, the result will be evaluated. Once the result meets the requirement, searching is stopped. When the result is turned away from the requirement, the searching in the particular branch is stopped, but is continued in other branches. Whenever all branches are stopped, either because the lowest child nodes have been detected, or due to getting away from the target, the searching produces no result.

7.4.4 Spatio Temporal Query Post Processing

The execution of each sub query results in a row of sets which meet the criteria supplied in the query request. The row sets need to be operated to another subquery to get the final result. The final set means nothing, if it is given to the user as the original tabular result. The results need to be converted into the desired format. Our proposal provides a possible format, such as a list of frames, where the expected object exists, the specific supervoxel only, or a projection of the region in the particular frames. In order to display the result, we need to access the initial label map and the original video.

7.5 Multimedia Database User Interface

The prototype of the multimedia database interfaces are divided into two categories. First, the interface is designed for the server side, which has functions such as to open the video, perform segmentation, generate a binary partition tree, perform simplification, and record the metadata. Then, there is a user side which provides an interface to write the query and display the query result.

The user interface for the video database prototype is provided in Figure 7.3. The user interface allows users to type a query and the result is displayed in the display video results as can be seen in Figure 7.4.

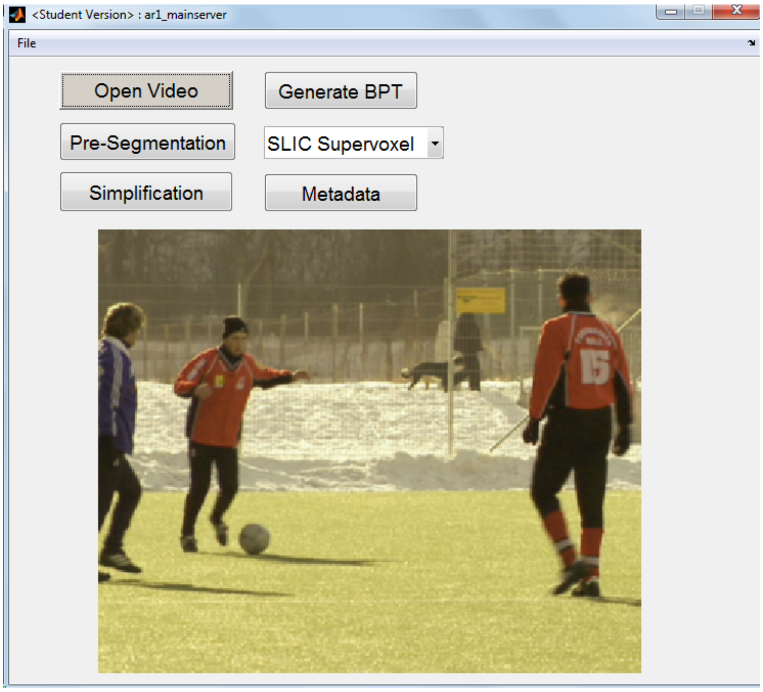
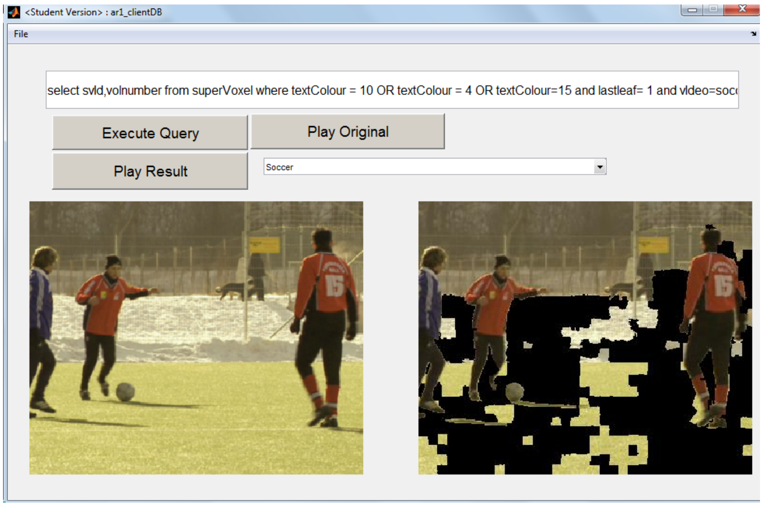


Figure 7.3: User Interface for Generate Metadata

7.6 Result and Evaluation

Although each extended keyword is tested and produces a result, evaluating how far the results meet the expected quality is not straightforward. This is different from the evaluation in the previous chapters. In that case, a comparison of the simplification to the available ground truth was an objective measure of the quality. To measure whether or not a query produces an appropriate result can be very subjective, however.

The evaluation in this section aims to answer the question ‘is the query working?’, therefore. We still use the same dataset in order to evaluate every single spatio temporal operation proposed in the previous section. In our experiment, the result from the pre-segmentation would be visually compared to the result from the simplified sets.



```
select svld,volnumber from superVoxel where textColour = 10 OR textColour = 4 OR textColour=15 and lastleaf= 1 and video=socc
```

Figure 7.4: User Interface for Execute Spatio Temporal Query

7.6.1 Spatial

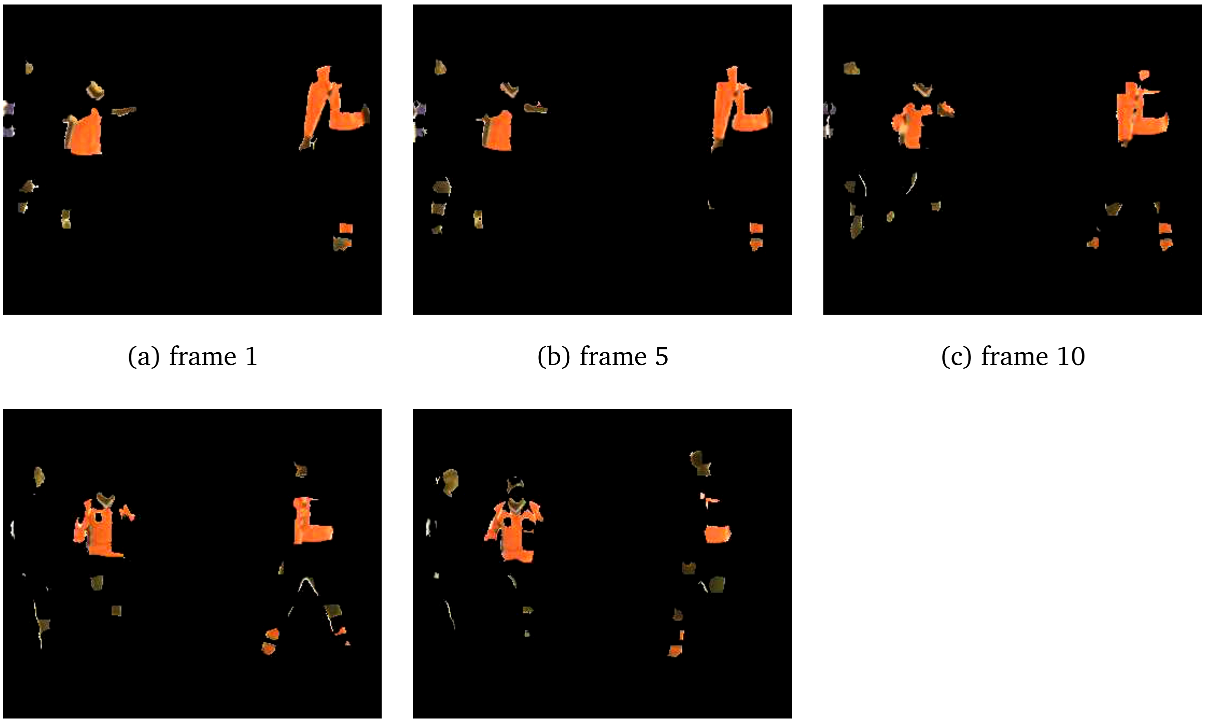
Spatial features such as colour can be the requirement of content queries. Colour, for example, can be passed to the system as a query specification. It can be a single colour or combination of more than one colour. Particularly for colour, only the OR operator can be answered because in every node a single colour is assigned, therefore, an AND logical will not work.

7.6.1.1 Single Colour in pre-segmentation set

This experiment tries to retrieve the content in the soccer video that has a maroon colour. This would be retrieved from pre-segmentation set before simplification. The complete query syntax is:

select svId,volnumber from superVoxel,video where textColour = ‘maroon’ and leftleaf=0 and video.title = ‘soccer’

The result can be seen in Figure 7.5.



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

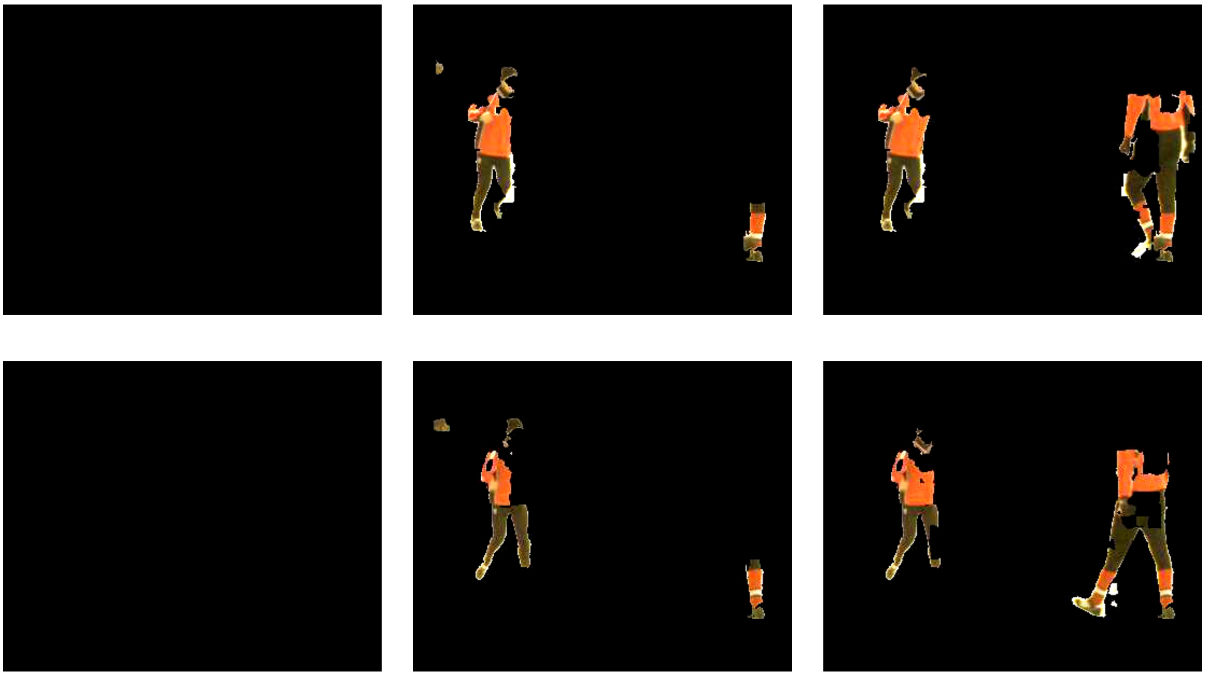
Figure 7.5: Frame 1,5,10,15,20 of Single Colour Query Result for Pre-segmentation of ‘Soccer’ Video

7.6.1.2 Single Colour in A Simplified Segmentation Sets

This experiment tries to retrieve content in the soccer video that has a maroon colour . It would be retrieved from simplified set. The complete query syntax:

select svId,volnumber from superVoxelAll where colourName1 = ‘Maroon’ and simplified=$simplevel$ **and title = ‘soccer’**

The variable $simplevel$ can be set 1, 2 or 3 to specify which simplification level to be retrieved. The result can be seen in the Figure 7.6.



(a) frame #1 and #10 Simplification 1

(b) frame #1 and #10 Simplification 2

(c) frame #1 and #10 Simplification 3

Figure 7.6: Frames #1 and #10 of Single Colour Query Result for Simplification 1,2 and 3 of The 'Soccer' video

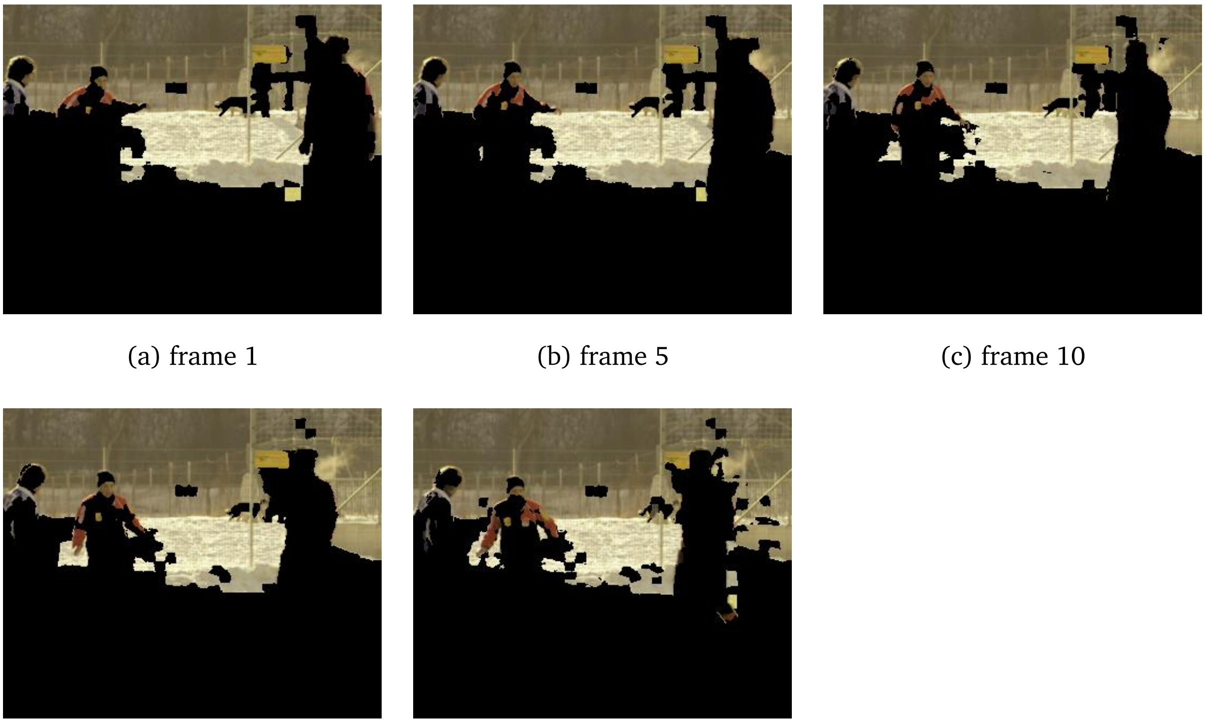
As seen in Figure 7.6, the maroon object cannot be retrieved from $simplification1$ due to the effects of merging. Merging among supervoxels causes the colour average among them do not to reflect the original colour. In contrast, $simplification3$ still maintains the original colour information in the partition, and therefore the result shown not only the pure maroon colour but also the regions around them that has been merged but the total average still in maroon colour range.

7.6.1.3 Multiple Colour in simplified segmentation set

This experiment tries to retrieve content in the soccer video which has silver or grey colour. This is retrieved from $simplification2$ set. The complete query syntax is:

```
select svId,volnumber from superVoxelAll where (colourName1 = ’Silver’ OR colourName1 = ‘Gray’ ) and simplified=2 and title = ‘soccer’
```

The result can be seen in Figure 7.7.



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

Figure 7.7: Frame 1,5,10,15,20 of Multiple Colour Query Result for Simplification 2 of the ‘Soccer’ video

7.6.2 Spatio Temporal

Generally, a spatio temporal related operation is usually needed by an object tracking task.

7.6.2.1 Static Background Filtering from Pre-Segmentation Set

Video content consists of a foreground and background. Background parts can be either static or experience camera motion (ego motion) while the foreground is usually moving. Filtering part of the scene with no motion can be performed with the textual query below.

SELECT region FROM supervoxel where motion = 'static' and Leftleaf = 0 and video = 'ice'

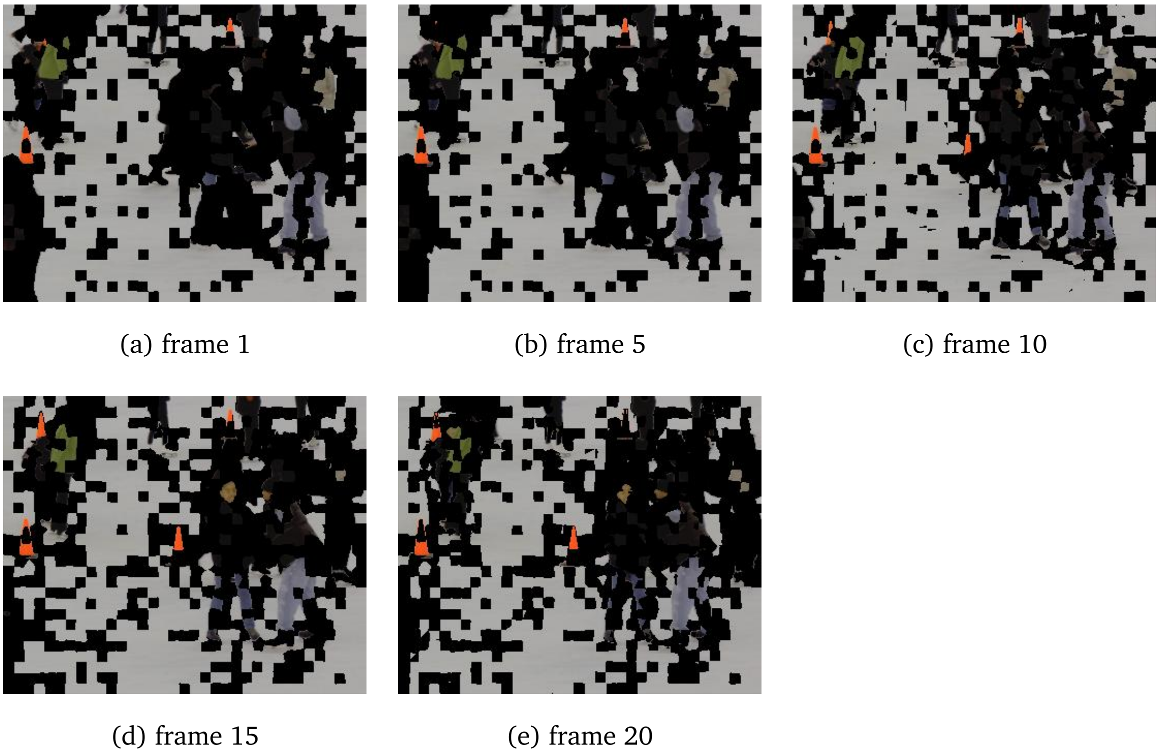
The output can be seen in Figure 7.8.

7.6.2.2 Static Background Filtering From A Simplified Set

This experiment tries to retrieve content in the 'Ice' video which has no motion (static). This is retrieved from $simplification1$ set. The complete query syntax is:

SELECT region FROM supervoxel where motion = 'static' and Lastleaf = 1 and video = 'ice'

The output can be seen in Figure 7.9.



(a) frame 1

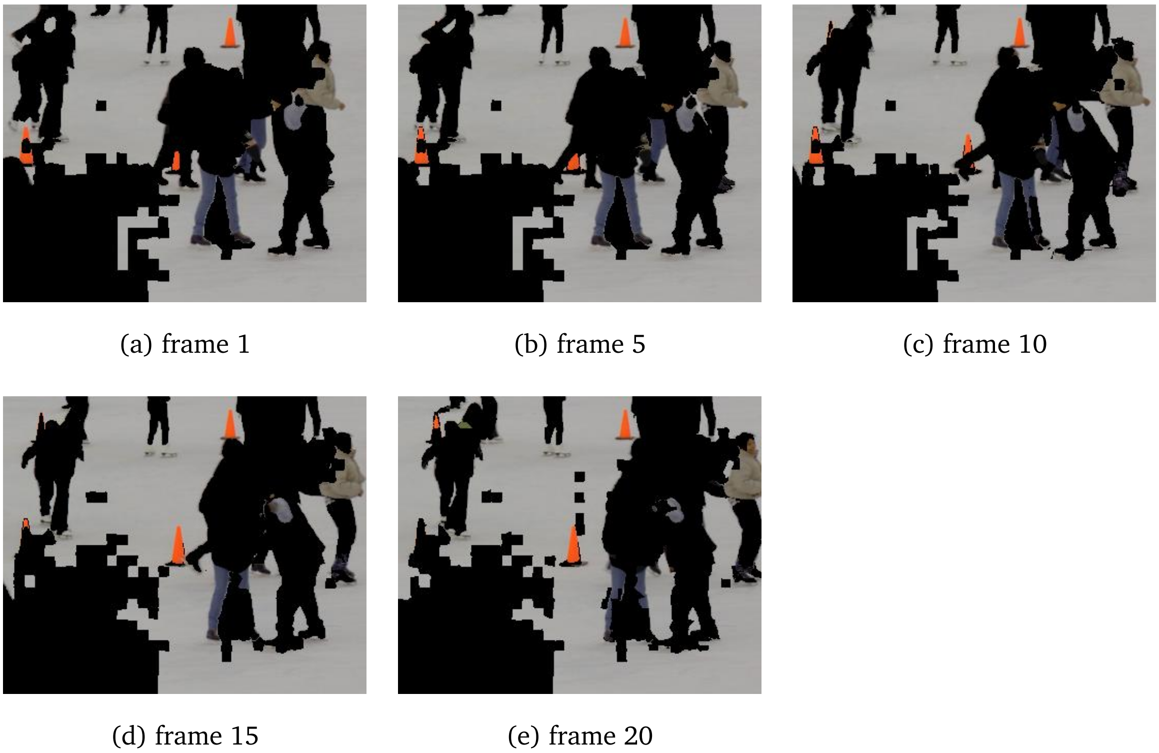
(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

Figure 7.8: Frame 1,5,10,15,20 of Static Region Query Result for The Pre-segmentation of The ‘Ice’ Video



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

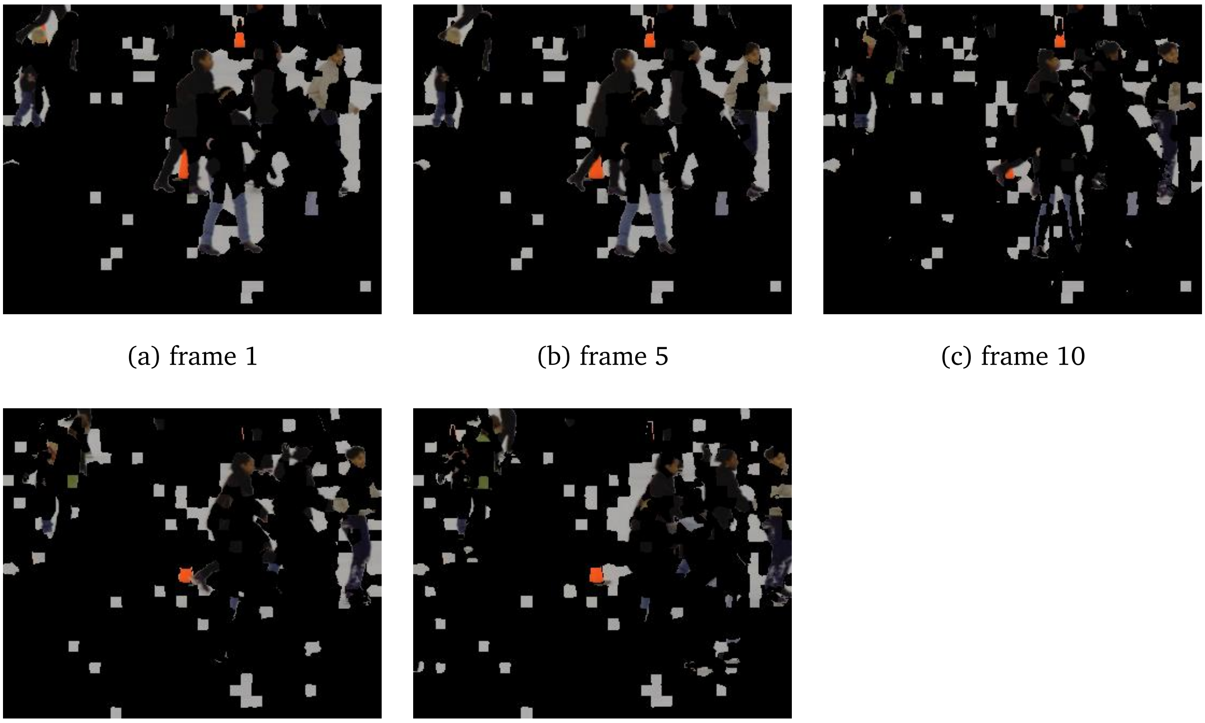
Figure 7.9: Frame 1,5,10,15,20 of Static Region Query Result for the $simplification1$ Set of The ‘Ice’ Video

7.6.2.3 Moving Foreground Filtering from the Pre-Segmentation Set

This experiment tries to retrieve content in the ice video which moves to the right direction. It is revealed from a pre-segmentation set. The complete query syntax is:

select svId,volnumber from superVoxel where motion = right and leftleaf= 0 and video = ice

The result can be seen in Figure 7.10.



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

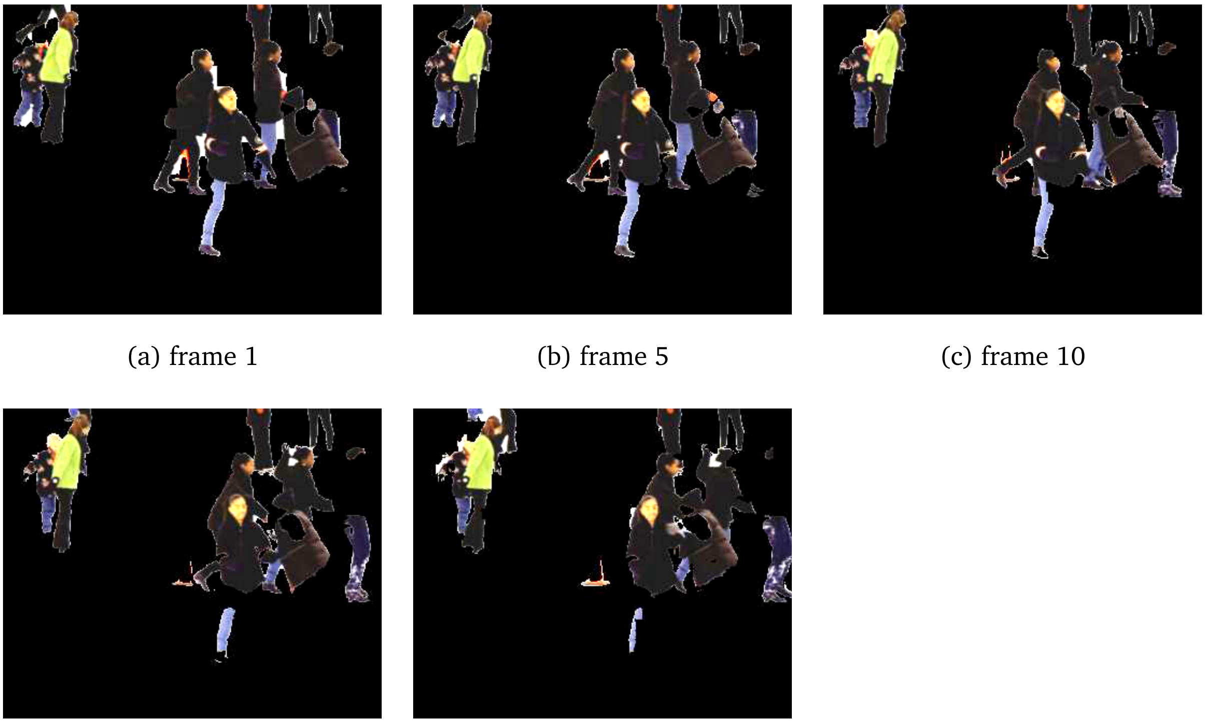
Figure 7.10: Frame 1,5,10,15,20 of Right Motion Region Query Result for The Pre-segmentation of The 'Ice' Video

7.6.2.4 Moving Foreground Filtering from the Simplified Set

This experiment tries to retrieve content in the video which moves to the right direction in the 'Ice' video. It is retrieved from the $simplification1$ set. The complete query syntax is:

The result can be seen in Figure 7.11.

```
select svId,volnumber from superVoxel where motion = right and lastleaf = 1 and video = 'ice'
```



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

Figure 7.11: Frame 1,5,10,15,20 of The Right Motion Region Query Result for Simplified Segmentation of the 'Ice' Video

As can be visually seen from the result of the pre-segmentation set and simplified set above, the pre-segmentation set gives more supervoxels and, therefore, more regions per frame. Small motions in every single region will affect the result. This can be seen in Figure 7.10 where some of the static background was misinterpreted as moving regions, while some regions inside the moving object are considered as static objects. This is because the motion can be identified in the motion boundary that is located on the border of the moving object. While in the middle of the object, if a partition is surrounded by an almost similar colour region, the motion will be absent. This is one of the weaknesses of this technique since the motion vector is not explicitly calculated.

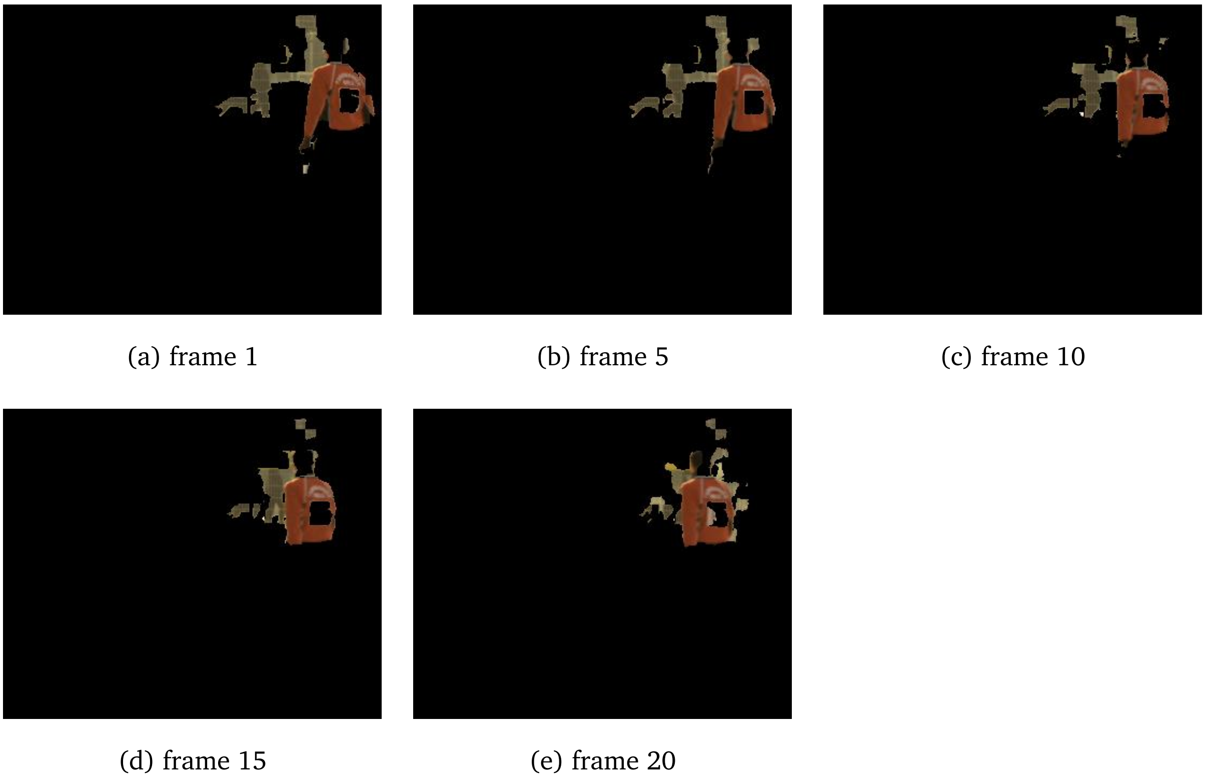
In contrast, Figure 7.11 presents the result of moving objects from the simplified set. Simplified sets provide a smaller number of regions with larger sizes. The result looks much better; as few static backgrounds were misinterpreted as moving objects. This is because, when they are merged with each other, misinterpreted motion due to centroid displacement has been compensated.

7.6.3 Motion and Colour

This experiment tries to retrieve content in the video which moves to the left direction and maroon colour from the 'Soccer' video. It will be retrieved from the $simplification1$ set. The complete query syntax is:

select svId,volnumber from superVoxel where motion = left and textColour = 'maroon' and lastleaf = 1 and video = 'soccer'

The result can be seen in Figure 7.12.



(a) frame 1

(b) frame 5

(c) frame 10

(d) frame 15

(e) frame 20

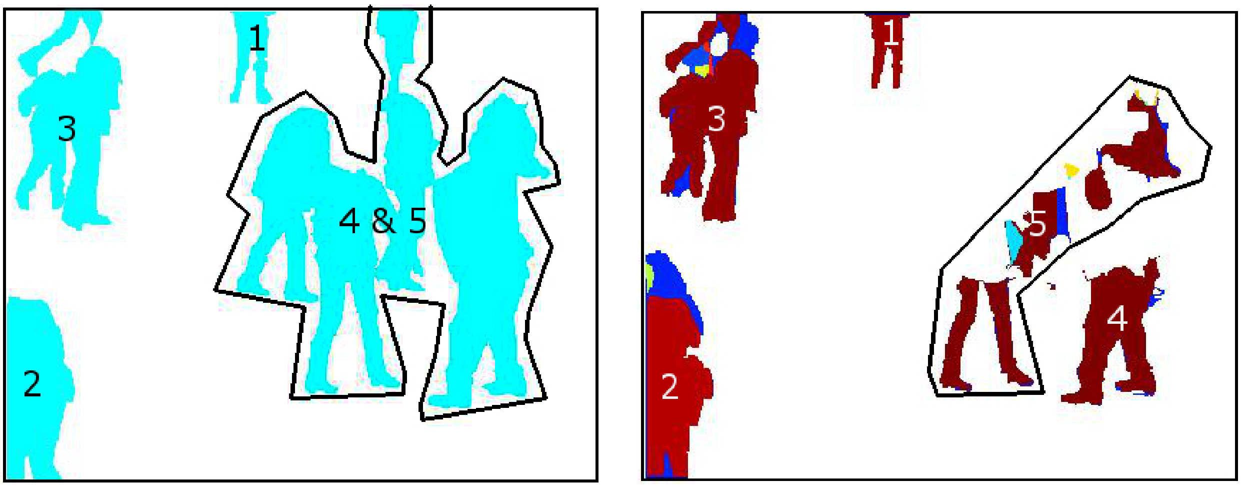
Figure 7.12: Frame 1,5,10,15,20 of Left Motion Region and Colour = maroon Query Result for a Simplified Segmentation of The 'Soccer' Video

7.6.4 Top Salient Candidate

This experiment tries to retrieve salient content in the video. Salient content is revealed from the entire tree. For example, if one desired the five most salient content in the 'Ice' video, the query can be formulated as follows:

SELECT svId FROM Supervoxel WHERE area > 10000 and TopSaliency <=5 and Video = 'ice'

The result can be seen in Figure 7.13.



(a) Ground Truth

(b) The Top 5 Salient Nodes

Figure 7.13: Top 5 Salient Candidate and The Corresponding Ground Truth of the 'Ice' Video

As can be seen in Figure 7.13, the five most salient nodes correspond to the object in the ground truth. The image on the right side shows the first rank (denoted by letter 1) to the fifth most salient (denoted by letter 5). The red colour denotes the area of the salient region corresponding to the ground truth. As can be seen, the majority of the first rank salient partitions correspond to the object in the ground truth. The second to the fifth salient partitions correspond to the ground truth object to some extent. The ground truth of salient 4^{th} and 5^{th} rank correspond to the same ground truth. This is an example of the ground truth quality problem. As can be seen, this ground truth object consists of more than one object that occlude each other, and are therefore considered as one ground truth object.

This experiment demonstrates a potential application to request a summary of the content of a video. Further steps to recognize and describe the salient nodes lead to automatic video content description.

7.7 Conclusion

In this chapter, a prototype of the video database management system has been provided. The metadata is produced based on a hierarchical BPT structure. The functionality of an extended query language to support spatio temporal requests has been demonstrated. Measuring the output quality of the testing query is not straightforward, since instead of providing quantified output a visual output is presented at the evaluation.

The colour query request has been tested for the pre-segmentation set, $simplification1$, $simplification2$ and $simplification3$. As can be seen in Figure 7.5, the result from the pre-segmentation set shows an incomplete result. The $simplification1$ set in Figure 7.6(a) does not show any result. This is because in $simplification1$, the merging has been carried out and the colour feature has been undergone considerable changes from its original colour due to averaging operation. The $simplification2$ and $simplification3$ which have lower peaks show better results as the colour feature are still retained. In general, colour query requests are answered more precisely in the initial pre-segmentation set. High level simplification ($simplification1$ is the highest level and the simplest tree) causes a loss of information in respect to the colour detail, meaning that colour information requests cannot be answered correctly. As can be seen in the $simplification2$ and $simplification3$, besides the requested colour object, the objects around them are also present on the output. Figure 7.7 shows a similar outcome: the red colour object exists in the output even though the query requests silver or grey objects. This is from $simplification2$ set.

Regarding temporal activity, a query of a static object is carried out. Figures 7.8 and 7.9 compare the static object query for the pre-segmentation set and the simplification set. As can be seen, the result from the simplification set gives a more continuous object compared to the discrete and incomplete object from the pre-segmentation set. The moving object query is shown in Figures 7.10 and 7.11. The result confirms that the simplification sets give better output, as can be seen in Figure 7.11. This is because the motion misclassification that occurs at the smaller partitions is compensated for when they are merged to form bigger partitions.

In the BPT structure, the evolutions of partitions are recorded. Salient partitions are identified by exploiting its formation history. Analogous to the way the pruning node is identified by exploiting the distance between child nodes and parent nodes, the salient nodes are identified in the same way. Nodes are rank ordered according to the child-parent distance: the higher the distance the more salient a node. The top ranks are then compared to the ground truth objects. It is arguable that objects in the ground truth are salient, and are therefore selected by human subject who created it. Figure 7.13 shows the top ranks salient nodes from 'Ice' video. It can be seen that the top most regions are aligned to the ground truth.

This chapter has demonstrated how the intermediate metadata is able to answer content queries. As previously mentioned, this is not designed to provide exact semantic meaning, such as retrieving a frame with football players in it, but it can provide intermediate descriptions of the target scene. For example, if the user knows that the target is a football player with a red T-shirt, one can describe the query as: search for a moving red object, as demonstrated in Figure 7.12. Of course if in the video collection, there is another video with a moving red object, this will also be selected as the output. In order to avoid this, more detailed descriptions of the target need to be defined, such as: search the moving red object and static green object. The top salient functionality demonstrates the ability to search for important objects that may be semantically meaningful, as demonstrated in Figure 7.13, and therefore, a summary of the video content can be created. This could be the input to automatic recognition tasks. More intensive exploration and video testing still needs to be done in order to cope with more complex requests.

Chapter 8

Summary and Future Work

8.1 Conclusion

The main objective of this thesis has been to provide intermediate level metadata of video content that enables content-based information requests. The separation of low level processing and high level content analysis is performed by generating metadata. It is useful to facilitate higher-level processing such as cognition without the need to carry out lower level data processing such as segmentation or merging. A number of experiments in Chapter 7 provide evidence for some extent of cognition. The tasks can be completed by accessing the metadata without directly connecting the original video unless visualization of the result is required.

The metadata is recorded based on pre-segmentation, merging and simplification tasks in a binary partition tree framework. The functionality of the metadata is evaluated by formulating colour-related, motion-related and salient content queries. The experiment in Chapter 7 demonstrated a higher-level content analysis, where high level content requests can be processed. The metadata is designed to record intermediate level abstraction, whereby a content information request has to be expressed in a certain form of description. The metadata is not equipped with semantic descriptions such as 'football player', 'dancer', etc., but information such as colour, motions, and direction are recorded so that a semantic request can be described in simpler forms. For example, a request for content of 'football player in red T-shirt' can be described as: 'red object moving and static green object'.

An unsupervised scenario is desired due to its freedom from user intervention and to pave the way to a general solution and machine autonomous processing. The pre-segmentation, merging and simplifications are carried out in unsupervised manner. During these tasks, no user intervention is needed to guide the process. As demonstrated in Chapters 4, 5, and 6, where the metadata is prepared, the process is based on the image or video data only. The input needed is a K value when the pre-segmentation algorithm is SLIC, and the α and β to set the proportion of colour and motion factors in the similarity measure for supervoxel merging. No user input or provided prior knowledge is needed to drive the segmentation quality. This unsupervised functionality, however, affects the quality of the segment, which may be far from the semantic meaning. This quality is acceptable for two reasons: the output of segmentation and simplification is not designed to provide semantic objects; the binary partition tree records the merging history that enables searching from coarse to detailed partitions.

A binary partition tree is created to record the transformation of each initial partition to the root (representing the entire image/video). The BPT for a single frame/image has been discussed in Chapter 4, while Chapter 6 introduced the tree for an entire video. The forms of BPT for two different purposes are identical; the only difference between them is the data represented by the node. In a single frame, every node in BPT represents a region/superpixel. In an entire video, every node of the BPT represents a volume/supervoxel.

BPTs record node transformation from low level nodes, which represent initial small partitions, to high level nodes, which represent the merging result and are therefore, bigger and more extensive documented structurally. The tree structure allows an operation to track the evolution of the nodes. An analysis of the transformation is carried out in order to identify critical merging, where the nodes have undergone significant changes. Significant changes indicate that a pair of nodes belong to different objects. In this event, the nodes experience critical merging, and the node is considered to be a pruning node.

The branch of the tree under the pruning nodes have experienced insignificant changes during their merging operation. It is reasonable, therefore, to cut these branches from the tree under the pruning nodes. The remaining nodes at higher levels to the critical nodes to the root remains in the BPT. As illustrated in Figure 6.7, the simplified tree is simple, and consists of fewer nodes. It also demonstrates three levels of simplification, which are dependent on the level of critical merging occurring on the pruning nodes. The critical merging rate is indicated by the value of the peak (local maxima) on the node evolution. Higher peaks are usually identified in the later merging iterations, and therefore, in the higher levels of the tree (closer to the root). Figure 6.7(a) confirms this indication that the highest simplification level ($simplification1$) gives fewer numbers of nodes, and therefore, fewer partitions. The higher levels of simplification are prone to under-segmentation problems due to excessive merging, however. Under-segmentation error is clearly seen in Figure 6.7(a) where the face has been merged with the background. In contrast, lower level segmentation still maintains the detail, but it suffers from over-segmentation error. A high over-segmentation rate is observed in the pre-segmentation result as presented in Figures 6.8 and 6.9. The simplification results with better over-segmentation rates are compared in Table 6.7 and illustrated in Figures 6.11, 6.12 and 6.13.

Pre-segmentation plays an important role in preparing the hierarchical partition tree in the BPT structure. According to Table 4.1 in Chapter 4, the watershed algorithm produces highly over-segmented initial partitions with the advantages of superior boundary recall. In contrast, mean shift and SLIC produce moderate over-segmentation rates with less boundary recall. The number of initial partitions directly affects the complexity of BPT, which means that BPT created with watershed are much complex and have more nodes than those from mean shift and SLIC. Simplification is highly affected by BPT complexity, and Table 4.5 shows a comparison of the evaluated methods, indicating that, on average watershed takes longer to complete all tasks. The most stable boundary recall is shown by a simplified tree from SLIC at around 0.82.

A good segmentation result is expected to be close to the human segmentation, which is represented by the ground truth. Although it is still arguable that some ground truth is subjective, the ability to asses segmentation quality reliably still depends on the ground truth. As mentioned before, pre-segmentation produces an initial over-segmentation partitions, and therefore, a simplification task needs to be performed in order to obtain set of segments that closer to the expected ground truth. The quality of simplification is assessed by comparing the boundary of partitions and the ground truth. The quantification is presented in boundary recall, precision and under-segmentation error. According to Figures 4.2(a), 4.3(a) and 4.4 (a) the best boundary recalls are achieved by simplified partitions using the histogram distance measure in watershed, mean shift and SLIC. The histogram distance method needs double the computation time, however. With a slight lower boundary recall, Euclidean and absolute distance take less time to create and simplify the BPT. Comparing Tables 4.2, 4.3 and 4.4, in general, for the same input, Euclidean distance gives the best performance. Considering the boundary recall, speed, over segmentation-rate and under-segmentation error a combination of SLIC and Euclidean distance is the most reliable combination of methods. This evaluation was conducted on eight test videos on a frame segmentation basis.

A frame-to-frame approach dedicated to deal with streaming condition where one frame available at a time. Segmentation and simplification are carried out for each individual frame. Matching tasks aims to correlate regions in the current and the previous frame. A temporal adjacency graph is documented as a complement to RAG. The quantity of partitions in each frame affects the number of matching operations required.

Video content in a particular frame is usually inherited from the previous frame. The regions in the current frame therefore have a correlation to their predecessor. Correlated regions could stay in the static position or move in a certain direction. Limited window searching aims to establish all region correlations across the frame by moving the window in the current frame and seeking the most similar regions in a window twice the size in the previous frame. This is work for the entire frame, however, which needs heavy computation. For instance, region correlation of a frame with 1000 regions can be calculated as follows. It is assumed the twice bigger windows contain nine regions and therefore establishing region correlation across two frames is equal to nine thousand iterations.

The salient region correlation method tries to minimize the iteration by selecting only the most salient region. Salient regions are indicated by the high child-parent distance. The highest child-parent distance over the tree is considered to be the most salient. The salient region list is prepared for all frames, and correlation is built among the salient regions. It is fast, but incomplete and according to the figures in Table 5.2 and Figure 5.11 there is a possibility of matching salient regions that are in fact uncorrelated.

Considering video as three-dimensional matrices is an alternative way to avoid the heavy computation required to build the temporal region correlation. To be able to represent video in that way, the entire video has to be available as an input. A set of supervoxels is produced in pre-segmentation of three-dimensional matrices. These has spatial and temporal dimension at the same time. Table 6.3 shows the pre-segmentation using watershed algorithm that is highly over-segmented (at 26.165 on average) with short duration at around 3.1 frames. SLIC supervoxels more manageable with a stable number of partitions since it depends of the input parameters.

As can be seen in Table 6.4 the SLIC produces on average 2259 initial partitions with an average duration of 9.7 frames. Acording to Table 6.9, the speeds of the entire process from initial segmentation to the simplification of SLIC are faster than that of watershed. Average running time recorded at 575 and 875 milliseconds for SLIC and watershed respectively. This is because the quantity of the initial partition affects the entire iteration. Considering the quality of segmentation indicated by the boundary recall rate, the under-segmentation error and over-segmentation rate presented in Tables 6.5, 6.6, 6.7, and 6.8. The BPT created over the initial segmentation produced by SLIC is more reliable than that of watershed. It is fast and maintain good quality of segmentation.

According to Figures 7.5, and 7.6 the colour query answered precisely using the pre-segmentation set. This happens because in pre-segmentation set the original colour that was set is retained, while, in the simplification set, supervoxels have been merged with each other, and therefore the colour has been mixed. It can be seen in Figure 7.6(a), the $simplification1$ set (the coarsest level) the colour request does not give any output.

A motion-related query is demonstrated in Figures 7.8, 7.9, 7.10, and 7.11. According to the experiment, the motion-related query gets a better response on the simplification set. As can be seen in Figure 7.8, a request for the static region answered with many holes in the middle of object compared to Figure 7.9, which shows a complete requested object. This occurs due to misclassification of the motion on the small partitions, which was compensated for when they were merged to the bigger partitions. This is the drawback of those methods that assume the centroid displacement represents the motion of the regions. This drawback is eliminated when the merging task has been conducted, however.

The motion-related combined with colour-related query on an intermediate metadata can be a powerful tools to retrieve objects in a video database. As mentioned in the beginning that the objective of this thesis is to provide intermediate level metadata. The content information request in semantic level cannot directly answer, but one can describe the semantic so that the metadata can answer it. For example, intermediate metadata cannot answer a semantic request such as: ‘search football player with red

T-shirt', or 'search ice skating video'. A further description need to be formulated, for example instead of 'search the football player with red T-shirt', the query can be formulated as 'search the moving red object', therefore, the proposed metadata can answer it as demonstrated in Figure 7.12. Of course if there are more video with red moving objects, the answer will be selected as the output as well. A more precise description need to be formulated to get the desired output such as: 'search red moving object and green static object'.

Salient supervoxels are formulated as a child node that is distant from its parent. As mentioned before, this indicates a critical merging between a pair of nodes of the tree. Critical merging suggests that the merging parties belong to different objects. In general, saliency is defined as an area in visual space that exhibits differences to its surrounding. By exploiting child-parent distance, a rank-ordered salient node is prepared. An example of top-most salient supervoxels is obtained from the 'Ice' video (Figure 7.13). This ability enables a video content summary to be composed that can be displayed as a list of important objects. Further work needs to be done to generate textual description automatically at the semantic level.

There are some issues that need to be addressed such as speed and inaccuracies. The issue of speed is one of the most significant problems. Table 6.9 shows that the running time needed for pre-segmentation, merging and simplification of 20 frames is quite slow compared to the video frame rate, which is 25 per seconds. The fastest speed is achieved by SLIC with 28 ms per frame on average for the 256 x 350 pixel/frame, in our testing machine. In fact, commercial video broadcasts using much higher resolutions (around 1024 x 1024), and therefore, to be usable in practical conditions the speed needs to be accelerated.

Inaccuracies identified in pre-segmentation and simplification results. In the current implementation, the motion vector is not explicitly calculated. Rather, it is estimated by the centroid displacement of the supervoxel. Due to the nature of motions, they are usually detected in the border of a region, but the risk of misinterpreting motion has been shown to have unwanted effects, particularly in the pre-segmentation set. Although this issue has been compensated for in the simplification set referred to Figure 7.11, more precise motion information leading to better interpretation of the segment is needed. Obtaining the motion vector, however, may reduce the speed, and therefore, overall, a moderate consideration of this issue has to be taken.

Inaccurate colour is identified in the simplification set that is demonstrated in Figure 7.6. This occurs due to the averaging operation when the merging task is executed. The second condition is because a limitation of semantic name of colour that adopted in the experiments. 16 semantic colour names are used. There is another option to use more semantic colours, but the problem with this is that sometimes these are not close to what humans usually used. One possibility is for the user interface to choose the colour sample.

8.2 Future Work

Intermediate metadata of video content has been provided, and some functionalities were demonstrated. There are some limitations that need to be addressed, however, as well as opportunities to develop the techniques in practical conditions. Some possible expansions of this work are:

- As demonstrated in Chapter 7, some experiments to exploit the content query are carried out. A description of requested content has yet to be defined in order to get the desired output, and the user has to define the description based on the available data recorded. It would be helpful if a machine provided those descriptions and let the user write to express his/her request in semantic human language.

- Streaming conditions where only a single frame is available at one time, need frame-by-frame approaches whereby segmentation and simplification are carried out in the frame. The upcoming frame is then processed individually and therefore, a correlation between the regions in a current and previous frame has to be made. In order to implement that scenario in practical conditions, an implementation of segmentation, simplification and region matching at frame rate speed has to be achieved.

- Pre-processing and metadata standardization in order to allow multimedia data interchange leading to the new design of multimedia communication.

- Semantic recognition can be performed for the top salient nodes. This could be done by selecting the highest rank salient nodes in the tree and performing recognition by drawing on prior knowledge such as an MPEG-7 shape database.

- Employing a region based database to carry out content searching without knowing any information about the target object. This can be done by an incomplete input such as only using the prominent colour surrounded by other colours.

- Tree based video coding can also be developed from the region database. By considering static regions in the tree that might be isolated in many high level branches in the BPT, tree based video coding can be generated by sending complete tree at the beginning followed by dynamic nodes only in the remaining frames.

- Provide a hierarchical region extraction service in order to allow the research community to make use of the metadata and spatio temporal query syntax to achieve higher level of cognition using the machine learning algorithm.
- Develop an integration with audio to text conversion and summarization in order to provide more reliable metadata.
- Speed up the pre-segmentation, merging and simplification by implementing the algorithm in a dedicated hardware in order to achieve real time processing speeds.

Appendices

Appendix A

Publications

1. Arief Setyanto, John C Wood, and Mohammed Ghanbary. Evolution Analysis of Binary Partition Tree for Hierarchical Video Simplified Segmentation. In Computer Science and Electronic Engineering Conference, volume i, pages 52 - 57, 2014.

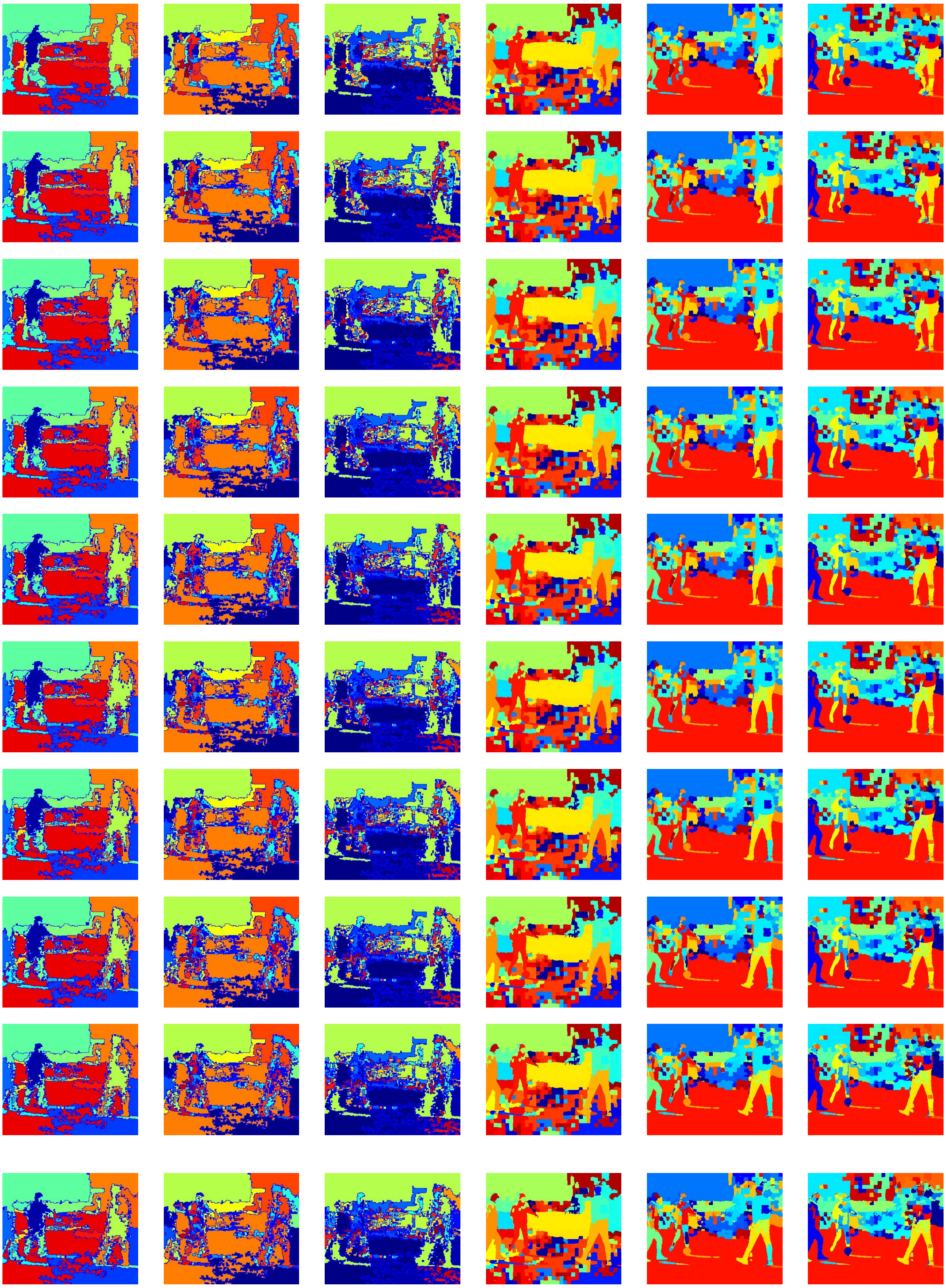
2. Arief Setyanto, John Charles Wood, and Mohammed Ghanbari. Platform for Temporal Analysis of Binary Partition Tree. In Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), 2013, pages 45 - 50, Poznan, Poland, 2013.

3. Arief Setyanto, John Charles Wood, and Mohammed Ghanbari. Genetic Algorithm for Inter-frame Region Object Temporal Correlation in Binary Partition Tree. In System Engineering and Technology (ICSET), 2012 International Conference on, pages 1 - 5, 2012.

Appendix B

Simplification Result

Color code simplification result for ’soccer’ video for first, second and third saliency peak. Complete simplification result available in the Compact Disc. Video format of Query Result is also available in digital format.



(a) WS simp 1 frame 1 to 10

(b) WS simp 2 frame 1 to 10

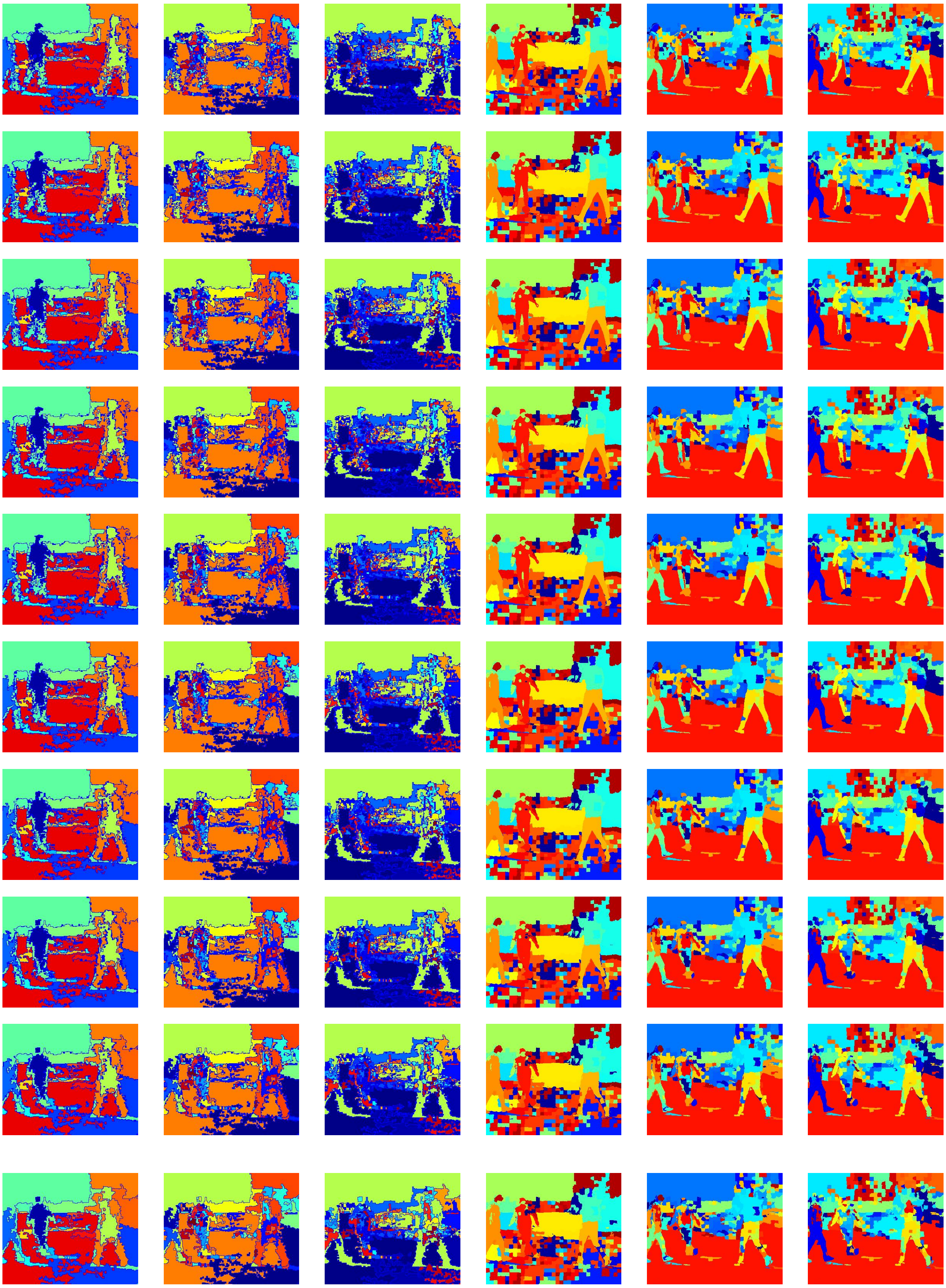
(c) WS simp 3 frame 1 to 10

(d) SLIC simp 1 frame 1 to 10

(e) SLIC simp 2 frame 1 to 10

(f) SLIC simp 3 frame 1 to 10

Figure B.1: Visual result of simplification peak 1, 2 and 3 of frames 1 - 10 of soccer video clips on watershed and SLIC pre-segmentation



(a) WS simp 1 frame 11-20

(b) WS simp 2 frame 11-20

(c) WS simp 3 frame 11-20

(d) SLIC simp 1 frame 11-20

(e) SLIC simp 2 frame 11-20

(f) SLIC simp 3 frame 11-20

Figure B.2: Visual result of simplification peak 1, 2 and 3 of frames 11 - 20 of soccer video clips on watershed and SLIC pre-segmentation

Bibliography

[1] M Ghanbari. *Standard codecs: Image compression to advanced video coding, 3rd Edition*. IET Telecomunication Series 54, 2011.

[2] M Sonka, Vaclav Hlavac, and R Boyle. *Image processing, analysis, and machine vision*. SPRINGER-SCIENCE+BUSINESS MEDIA, BV., 1999.

[3] Dengsheng Zhang and Guojun Lu. Segmentation of moving objects in image sequence: A review. *Circuits, Systems, and Signal Processing*, 20(2):143–183, mar 2001.

[4] P Salembier and L Garrido. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 9(4):561–76, jan 2000.

[5] CC Dorea, Montse Pardàs, and Ferran Marques. Trajectory tree as an object-oriented hierarchical representation for video. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(4):1–14, 2009.

[6] a.Y.C. Chen and J.J. Corso. Propagating multi-class pixel labels throughout video

frames. In *Image Processing Workshop (WNYIPW), 2010 Western New York*, pages 0–3, 2010.

[7] Huihai Lu, John C. Woods, and Mohammed Ghanbari. Binary Partition Tree for Semantic Object Extraction and Image Segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(3):378–383, mar 2007.

[8] Weiming Hu, Nianhua Xie, Li Li, Xianglin Zeng, and Stephen Maybank. A survey on visual content-based video indexing and retrieval. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 41(6):797–819, 2011.

[9] Fernando Pereira. MPEG-7: the generic multimedia content description standard, part 1. *MultiMedia, IEEE*, 9(June):78–87, 2002.

[10] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Efficient hierarchical graph-based video segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2141–2148. Ieee, jun 2010.

[11] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181, sep 2004.

[12] Yong Jae Lee, Jaechul Kim, and Kristen Grauman. Key-segments for video object segmentation. In *2011 International Conference on Computer Vision*, pages 1995–2002, nov 2011.

[13] Ian Endres and Derek Hoiem. Category independent object proposals. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence*

and Lecture Notes in Bioinformatics), 6315 LNCS(PART 5):575–588, 2010.

[14] S Lefèvre, J Holler, and Nicole Vincent. A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval. *Real-Time Imaging*, 9(2003):73–98, 2003.

[15] Ramírez A Graciela and Chacón M Mario I. New Trends on Dynamic Object Segmentation in Video Sequences : A Survey. *REVISTA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y COMPUTACIÓN,*, 11(1):29–42, 2013.

[16] KN Ngan and H Li. Image/Video Segmentation: Current Status, Trends, and Challenges. In *Video segmentation and its applications*, pages 1–24. Springer, New York, 2011.

[17] N Otsu. A threshold selection method from gray-level histograms. *IEEE TRANSACTIONS ON SYSTREMS, MAN, AND CYBERNETICS*, SMC-9(1):62–66, 1979.

[18] Paul L. Rosin. Unimodal thresholding. *Pattern Recognition*, 34(11):2083–2096, nov 2001.

[19] C. Chow and T Kaneko. Automatic boundary detection of the left ventricle from cineangiograms. *Computers and Biomedical Research*, 5:388–410, 1972.

[20] Soon H. Kwon. Threshold selection based on cluster analysis. *Pattern Recognition Letters*, 25(9):1045–1050, jul 2004.

[21] Agus Zainal Arifin and Akira Asano. Image segmentation by histogram thresholding using hierarchical cluster analysis. *Pattern Recognition Letters*, 27(13):1515–1521, oct 2006.

[22] S.-Y. Chien, Y.-W. Huang, B.-Y. Hsieh, S.-Y. Ma, and L.-G. Chen. Fast Video Segmentation Algorithm With Shadow Cancellation, Global Motion Compensation, and Adaptive Threshold Techniques. *IEEE Transactions on Multimedia*, 6(5):732–748, oct 2004.

[23] Xiaopeng Ji, Zhiqiang Wei, and Yewei Feng. Effective vehicle detection technique for traffic surveillance systems. *Journal of Visual Communication and Image Representation*, 17(3):647–658, jun 2006.

[24] Thomas Meier and KN Ngan. Automatic segmentation of moving objects for video object plane generation. *Circuits and Systems for Video Technology, IEEE Transaction on*, 8(5):525–538, 1998.

[25] P Dollár and CL Zitnick. Structured Forests for Fast Edge Detection. In *IEEE International Conference on Computer Vision, ICCV,* Sydney, Australia, 2013.

[26] YY Boykov and MP Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In *Computer Vision, 2001. ICCV 2001.*, volume I, pages 105–112, 2001.

[27] Dingming Liu and Jieyu Zhao. Spatio-temporal video object segmentation using moving detection and graph cut methods. In *2011 Seventh International Conference on Natural Computation*, pages 1859–1862. Ieee, jul 2011.

[28] J Shi and J Malik. Motion segmentation and tracking using normalized cuts. In *Computer Vision, 1998. Sixth International Conference on*, 1998.

[29] Wenbing Tao, Hai Jin, and Yimin Zhang. Color image segmentation based on mean shift and normalized cuts. *Systems, Man, and Cybernetics, Part B: Cybernet-*

ics, IEEE Transactions on, 37(5):1382–9, oct 2007.

[30] Ertem Tuncel and Levent Onural. Utilization of the Recursive Shortest Spanning Tree Algorithm for Video-Object Segmentation by 2-D. *Circuits and Systems for Video Technology, IEEE Transactions on*, 10(5):776–781, 2000.

[31] Chenliang Xu, Caiming Xiong, and JJ Corso. Streaming hierarchical video segmentation. In *European Conference on Computer Vision*, 2012.

[32] Pei Yin, Antonio Criminisi, John Winn, and Irfan Essa. Tree-based Classifiers for Bilayer Video Segmentation. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, jun 2007.

[33] H T Nguyen, M Worring, and a Dev. Detection of moving objects in video using a robust motion similarity measure. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 9(1):137–41, jan 2000.

[34] CC Dórea. *Hierarchical Partition-based Representations of Motion-coherent Regions for Video Object Segmentation*. PhD thesis, Technical University of Catalonia, 2007.

[35] M.a. El Saban and B.S. Manjunath. Video region segmentation by spatio-temporal watersheds. In *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)*, volume 1, pages I–349–52. Ieee, 2003.

[36] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 246–252. IEEE Comput. Soc, 1999.

[37] Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood, and Larry Davis. Real-time foreground- background segmentation using codebook model. *Real-*

Time Imaging, 11(3):172–185, jun 2005.

[38] Shao-yi Chien and Liang-gee Chen. Efficient moving object segmentation algorithm using background registration technique. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(7):577–586, jul 2002.

[39] Ioannis Patras. Video segmentation by MAP labeling of watershed segments. *IEEE transactions on pattern analysis and machine intelligence*, 23(3):326–332, 2001.

[40] Fabrice Moscheni, Sushil Bhattacharjee, and Murat Kunt. Spatiotemporal Segmentation Based on Region Merging. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(9):897–915, 1998.

[41] a. Cavallaro, O. Steiger, and T. Ebrahimi. Tracking video objects in cluttered background. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(4):575–584, apr 2005.

[42] Junqiu Wang and Yasushi Yagi. Consecutive tracking and segmentation using adaptive mean-shift and graph cut. In *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, 2007.

[43] Arief Setyanto, John Charles Wood, and Mohammed Ghanbari. Platform for Temporal Analysis of Binary Partition Tree. In *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA), 2013*, pages 45 – 50, Poznan, Poland, 2013.

[44] Thomas Meier and KN Ngan. Video segmentation for content-based coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1190–1203, 1999.

[45] C Gomila and F Meyer. Graph-based object tracking. In *Image Processing, 2003. ICIP 2003.*, pages 41–44, 2003.

[46] Eli Saber, Yaowu Xu, and A. Murat Tekalp. Partial shape recognition by sub-matrix matching for partial matching guided image labeling. *Pattern Recognition*, 38(10):1560–1573, oct 2005.

[47] Zhao Liu, Hui Shen, Guiyu Feng, and Dewen Hu. Tracking objects using shape context matching. *Neurocomputing*, 83:47–55, apr 2012.

[48] Thomas Schoenemann and Daniel Cremers. A combinatorial solution for model-based image segmentation and real-time tracking. *IEEE transactions on pattern analysis and machine intelligence*, 32(7):1153–64, jul 2010.

[49] Longin Jan Latecki, Vasileios Megalooikonomou, Qiang Wang, and Deguang Yu. An elastic partial shape matching technique. *Pattern Recognition*, 40(11):3069–3080, nov 2007.

[50] Yi-Zhe Song, Chuan Li, Liang Wang, Peter Hall, and Peiyi Shen. Robust visual tracking using structural region hierarchy and graph matching. *Neurocomputing*, 89:12–20, jul 2012.

[51] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[52] Geng Zhang, Zejian Yuan, Dapeng Chen, Yuehu Liu, and Nanning Zheng. Video Object Segmentation by Clustering Region Trajectories. In *International Conference on Pattern Recognition (ICPR)*, pages 2598–2601, 2012.

[53] Arief Setyanto, John Charles Wood, and Mohammed Ghanbari. Genetic Algorithm for Inter-frame Region Object Temporal Correlation in Binary Partition Tree. In *System Engineering and Technology (ICSET), 2012 International Conference on*, pages 1 – 5, 2012.

[54] Yi-ping Hung, Yu-pa Tsai, and Chh-chum Lai. A Bayesian approach to video object segmentation via merging 3D watershed volumes. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 1, pages 496–499. IEEE Comput. Soc, 2002.

[55] D DeMenthon and Remi Megret. Spatio-temporal segmentation of video by hierarchical mean shift analysis. In *Statistical Methods in Video Processing Workshop*, page 20, 2002.

[56] Mirko Ristivojević and Janusz Konrad. Space-time image sequence analysis: object tunnels and occlusion volumes. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 15(2):364–76, feb 2006.

[57] Arief Setyanto, John C Wood, and Mohammed Ghanbary. Evolution Analysis of Binary Partition Tree for Hierarchical Video Simplified Segmentation. In *Computer Science and Electronic Engineering Conference*, pages 52–57, 2014.

[58] Luis Garrido Ostermann. *Hierarchical Region Based Processing of Images and Video Sequences: Application to Filtering, Segmentation and Information Retrieval.* PhD thesis, Universitat Politecnica de Catalunya, 2002.

[59] Shirin Ghanbari, John C. Woods, and Simon M. Lucas. BPT Using Multidimensional Information for Semi-Automatic Content Retrieval. *2009 Conference*

for Visual Media Production, pages 169–175, nov 2009.

[60] Veronica Vilaplana, Ferran Marques, and Philippe Salembier. Binary partition trees for object detection. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 17(11):2201–16, nov 2008.

[61] V Vilaplana and F Marques. Object detection and segmentation on a hierarchical region-based image representation. In *Image Processing (ICIP), 2010 17th*, pages 3933–3936, 2010.

[62] J Pont-Tuset and F Marques. Contour detection using binary partition trees. In *IEEE Image Processing (ICIP), 2010 17th Internation Conference On*, pages 1609–1612, 2010.

[63] Tomasz Adamek and NE O'Connor. *Stopping region-based image segmentation at meaningful partitions*. Springer Berlin Heidelberg, 2007.

[64] M Esche, M Karaman, and T Sikora. Semi-automatic object tracking in video sequences by extension of the MRSST algorithm. In *Analysis, Retrieval and Delivery of Multimedia Content*, pages 57–70. Springer New York, 2013.

[65] F. Dorea, C.C.; Pardas, M.; Marques. A MOTION-BASED BINARY PARTITION TREE APPROACH TO VIDEO OBJECT SEGMENTATION. In *International Conference on mage Processing (ICIP), 2005*, pages II,430–3, 2005.

[66] M Maziere and F Chassaing. Segmentation and tracking of video objects for a content-based video indexing context. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 2, pages 1191–1194, 2000.

[67] L Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE transactions on pattern analysis and Machine Intelligence*, 13(6):583–598, 1991.

[68] Dorin Comaniciu, Peter Meer, and Senior Member. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[69] Radhakrishna Achanta, Appu Shaji, and Kevin Smith. SLIC superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis an*, 6(1):1–8, 2012.

[70] P Salembier and F Marqués. Region-based representations of image and video: segmentation tools for multimedia services. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1147–1169, 1999.

[71] Zhi Liu, Wenbin Zou, and Olivier Le Meur. Saliency tree: a novel saliency detection framework. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 23(5):1937–52, may 2014.

[72] Jifeng Ning, Lei Zhang, David Zhang, and Chengke Wu. Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition*, 43:445–456, 2010.

[73] Zhi Liu, Liquan Shen, and Zhaoyang Zhang. Unsupervised image segmentation based on analysis of binary partition tree for salient object extraction. *Signal Processing*, 91(2):290–299, feb 2011.

[74] T Adamek and NE O'Connor. Using dempster-shafer theory to fuse multiple information sources in region-based segmentation. In *Image Processing, 2007. ICIP*

2007, volume 2, pages 269–272, 2007.

[75] Roberto Castagno, Touradj Ebrahimi, and Murat Kunt. Video segmentation based on multiple features for interactive multimedia applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):562–571, 1998.

[76] D.D. Giusto, F. Massidda, and C. Perra. A fast algorithm for video segmentation and object tracking. *2002 14th International Conference on Digital Signal Processing Proceedings. DSP 2002 (Cat. No.02TH8628)*, 2:697–700, 2002.

[77] Thanos Athanasiadis and Phivos Mylonas. Semantic image segmentation and object labeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(3):298–312, 2007.

[78] Hanfeng Chen. Supervised video object segmentation using a small number of interactions. *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, 3:III–365–8, 2003.

[79] Valérie Gouet-Brunet and Bruno Lameyre. Object recognition and segmentation in videos by connecting heterogeneous visual features. *Computer Vision and Image Understanding*, 111(1):86–109, jul 2008.

[80] Minh Hoai, Zhen-Zhong Lan, and Fernando De la Torre. Joint segmentation and classification of human actions in video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3265–3272. Ieee, jun 2011.

[81] Vijay Badrinarayanan, Ignas Budvytis, and Roberto Cipolla. Semi-supervised video segmentation using tree structured graphical models. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2751–64, nov 2013.

[82] Vijay Badrinarayanan, Ignas Budvytis, and R Cipolla. Mixture of Trees Probabilistic Graphical Model for Video Segmentation. *International Journal of Computer Vision*, 110(1):14–29, 2013.

[83] Yu-Ting Chen, Chu-Song Chen, Chun-Rong Huang, and Yi-Ping Hung. Efficient hierarchical method for background subtraction. *Pattern Recognition*, 40(10):2706–2715, oct 2007.

[84] Félix Raimbault, François Pitié, and Anil Kokaram. User-assisted sparse stereo-video segmentation. *Proceedings of the 10th European Conference on Visual Media Production - CVMP '13*, pages 1–10, 2013.

[85] Peter Ochs and Thomas Brox. Object segmentation in video: A hierarchical variational approach for turning point trajectories into dense regions. In *2011 International Conference on Computer Vision*, pages 1583–1590. Ieee, nov 2011.

[86] Anestis Papazoglou and Vittorio Ferrari. Fast Object Segmentation in Unconstrained Video. *2013 IEEE International Conference on Computer Vision*, pages 1777–1784, dec 2013.

[87] Sch Hoi, Lls Wong, and Albert Lyu. Chinese university of hongkong at trecvid 2006: Shot boundary detection and video search. In *TRECVid 2006 Workshop*, 2006.

[88] An-Ni Cai Zhi-Cheng Zhao. Shot Boundary Detection Algorithm in Compressed Domain Based on Adaboost and Fuzzy Theory. In Licheng Jiao, Lipo Wang, Xinbo Gao, Jing Liu, and Feng Wu, editors, *Advances in Natural Computation, Lecture Notes in Computer Science*, volume 4222 of *Lecture Notes in Computer Science*,

pages 617–626. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[89] JeonKyu Lee, JungHwan Oh, and Sae Hwang. STRG-Index: Spatio-Temporal Region Graph Indexing for Large Video Databases. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 718–729, 2005.

[90] Dingyuan Xia, Xuefei Deng, and Qingning Zeng. Shot Boundary Detection Based on Difference Sequences of Mutual Information. In *Fourth International Conference on Image and Graphics (ICIG 2007)*, pages 389–394. IEEE, aug 2007.

[91] X. Wu, Pong C. Yuen, C. Liu, and J. Huang. Shot Boundary Detection: An Information Saliency Approach. In *2008 Congress on Image and Signal Processing*, volume 2, pages 808–812. IEEE, 2008.

[92] Gc Chavez and F Precioso. Shot boundary detection at trecvid 2006. In *Proceedings of TRECVID*, 2006.

[93] Xu-Dong Zhang, Tie-Yan Liu, Kwok-Tung Lo, and Jian Feng. Dynamic selection and effective compression of key frames for video abstraction. *Pattern Recognition Letters*, 24(9-10):1523–1532, jun 2003.

[94] A. Hanjalic, R.L. Lagendijk, and J. Biemond. Automated high-level movie segmentation for advanced video-retrieval systems. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(4):580–588, jun 1999.

[95] Liang-Hua Chen, Yu-Chun Lai, and Hong-Yuan Mark Liao. Movie scene segmentation using background information. *Pattern Recognition*, 41(3):1056–1065, mar 2008.

[96] A Natsev, W Jiang, M Merler, and J R Smith. IBM research TRECVid-2008 video retrieval system, 2008.

[97] A. Hauptmann, Rv Baron, My Chen, M. Christel, P. Duygulu, C. Huang, R. Jin, Wh Lin, T. Ng, N. Moraveji, and Others. Informedia at TRECVID 2003: Analyzing and searching broadcast news video. *Proc. of TRECVID*, pages 2834–2837, 2003.

[98] Duy-dinh Le, Shin Satoh, and Michael E Houle. Face Retrieval in Broadcasting News Video by Fusing Temporal and Intensity Information. In *Image and Video Retrieval*, pages 391–400. Springer Berlin Heidelberg, 2006.

[99] Ronan Fablet, Patrick Bouthemy, and Patrick Pérez. Nonparametric motion characterization using causal probabilistic models for video indexing and retrieval. *IEEE Transactions on Image Processing*, 11(4):393–407, 2002.

[100] Faisal I. Bashir, Ashfaq A. Khokhar, and Dan Schonfeld. Real-Time Motion Trajectory-Based Indexing and Retrieval of Video Sequences. *IEEE Transactions on Multimedia*, 9(1):58–65, jan 2007.

[101] Eitetsu Oomoto and Katsumi Tanaka. OVID: Design and implementation of a video-object database system. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):629–643, 1993.

[102] Max J. Egenhofer. Spatial SQL: a query and presentation language. *IEEE Transactions on Knowledge and Data Engineering*, 6(414):86–95, 1994.

[103] Martin Erwig and Markus Schneider. STQL - a Spatio -Temporal Query Language. In *Mining Spatio-Temporal Information Systems*, pages 105–126. Kluwer International Series in Engineering and Computer Science, 2002.

[104] Howard D Wactlar and Michael G Christel. Digital Video Archives : Managing Through Metadata. Technical report, Computer Science Department Carnegie Mellon University, 2002.

[105] Carlo Zaniolo Cindy Xinmin Chen, Cindy Xinmin Chen, and Carlo Zaniolo. SQL ST : a spatio-temporal data model and query language. *19th International Conference on Conceptual Modeling*, pages 96–111, 2000.

[106] Mehmet Emin Dönderler, Ediz Åđaykol, Özgür Ulusoy, and UÇğur Güdükbay. BilVideo: A video database management system. *IEEE Multimedia*, 10:66–70, 2003.

[107] Mehmet Emin Dönderler, Özgür Ulusoy, and UÇğur Güdükbay. Rule-based spatiotemporal query processing for video databases, 2004.

[108] M. BasÌğtan, H. CÌğam, U. GuÌĹduÌĹkbay, and O. Ulusoy. Bilvideo-7: an MPEG-7- compatible video indexing and retrieval system. *IEEE Multimedia*, 17:62–73, 2010.

[109] J Lee and M E Celebi. STRG-QL: Spatio-Temporal Region Graph Query Language for Video Databases. In *Proceedings of the SPIE Electronic Imaging Conference*, volume 6820, pages 68200P–1—-12, 2008.

[110] JeonKyu Lee, JungHwan Oh, and Sae Hwang. STRG-Index: Spatio-Temporal Region Graph Indexing for Large Video Databases. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 718–729, 2005.

[111] Ivan Giangreco, Ihab Al, and Kabary Heiko. ADAM - A Database and Information Retrieval System for Big Multimedia Collections. In *Big Data (BigData Congress),*

2014 IEEE International Congress on, pages 406–413, 2014.

[112] R Achanta and S Susstrunk. Saliency detection using maximum symmetric surround. In *17 th IEEE International Conference on Image Processing*, pages 2653–2656, 2010.

[113] Dongju Liu and Jian Yu. Otsu method and K-means. *Proceedings - 2009 9th International Conference on Hybrid Intelligent Systems, HIS 2009*, 1(2):344–349, 2009.

[114] J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[115] Alex Levinshtein, Adrian Stere, Kiriakos N. Kutulakos, David J. Fleet, Sven J. Dickinson, and Kaleem Siddiqi. TurboPixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, 2009.

[116] Sylvain Paris. Edge-preserving Smoothing and Mean-shift Segmentation of Video Streams. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 460–473, 2008.

[117] S. Beucher and C. Lantuejoul. Use of Watersheds in Contour Detection, 1979.

[118] Fernand Meyer. Topographic distance and watershed lines. *Signal Processing*, 38(1):113–125, 1994.

[119] Leonard Kaufman and Peter J Rousseeuw. Introduction. In *Finding Groups in Data*, pages 1–67. John Willey and Son, 1990.

[120] Huihai Lu, John C. Woods, and Mohammed Ghanbari. Binary partition tree analysis based on region evolution and its application to tree simplification. *IEEE Transactions on Image Processing*, 16:1131–1138, 2007.

[121] Matthias Reso, Jorn Jachalsky, Bodo Rosenhahn, and Jorn Ostermann. Temporally consistent superpixels. *Proceedings of the IEEE International Conference on Computer Vision*, pages 385–392, 2013.

[122] Pablo Arbeláez. Boundary extraction in natural images using ultrametric contour maps. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, 2006.

[123] Peer Neubert and Peter Protzel. Superpixel Benchmark and Comparison. In *Forum Bildverarbeitung*, 2012.

[124] Jesse Davis and Mark Goadrich. The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd International Conference on Machine learning – ICML'06*, pages 233–240, 2006.

[125] Michael Affenzeller, Stefan Wagner, Stephan Winkler, and Andreas Beham. Genetic algorithms and genetic programming modern concepts and practical applications. *CRC Press*, 2009.

[126] Zoran Stejić, Yasufumi Takama, and Kaoru Hirota. Genetic algorithm-based relevance feedback for image retrieval using local similarity patterns. *Information Processing & Management*, 39(1):1–23, 2003.

[127] Mantas Paulinas and Andrius Ušinskas. A survey of genetic algorithms applications for image enhancement and segmentation. *Information Technology and*

control, 36(3):278–284, 2007.

[128] M. Sarfraz and M. Iqbal. Object Recognition Using Fourier Descriptors and Genetic Algorithm. *2009 International Conference of Soft Computing and Pattern Recognition*, pages 318–323, 2009.

[129] Chih-Chin Lai and Ying-Chuan Chen. A User-Oriented Image Retrieval System Based on Interactive Genetic Algorithm. *IEEE Transactions on Instrumentation and Measurement*, 60(10):3318–3325, oct 2011.

[130] Hun-Woo Yoo and Sung-Bae Cho. A Video Scene Retrieval With Interactive Genetic AlgorithmTitle. *Multimedia Tools and Applications*, 34(3):317–336, 2007.