Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Golnaz Abdollahian

Entitled CONTENT ANALYSIS OF USER GENERATED VIDEO

For the degree of Doctor of Philosophy

Is approved by the final examining committee:

E. J. Delp Chair	
C. A. Bouman	
D. S. Ebert	
Z. Pizlo	

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): E. J. Delp

Approved by: V. Balakrishnan 11/06/09
Head of the Graduate Program Date

Graduate School Form 20
(Revised 10/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL

Research Integrity and Copyright Disclaimer

Title of Thesis/Dissertation: CONTENT ANALYSIS OF USER GENERATED VIDEO

For the degree of Doctor of Philosophy

I certify that in the preparation of this thesis, I have observed the provisions of *Purdue University Executive Memorandum No. C-22,* September 6, 1991, *Policy on Integrity in Research.**

Further, I certify that this work is free of plagiarism and all materials appearing in this thesis/dissertation have been properly quoted and attributed.

I certify that all copyrighted material incorporated into this thesis/dissertation is in compliance with the United States' copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless Purdue University from any and all claims that may be asserted or that may arise from any copyright violation.

Golnaz Abdollahian
Signature of Candidate

11/06/09
Date

*Located at http://www.purdue.edu/policies/pages/teach_res_outreach/c_22.html

CONTENT ANALYSIS OF USER GENERATED VIDEO

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Golnaz Abdollahian

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2009

Purdue University

West Lafayette, Indiana

ACKNOWLEDGMENTS

I owe a great deal of intellectual and emotional debt to many people that supported me in the development of this thesis, and my work as a graduate student in the School of Electrical and Computer Engineering at Purdue University.

The faculty at Purdue University helped me to expand my intellectual development in ways that I could never have imagined. First and foremost, I will never be able to express my gratitude to my advisor Professor Edward Delp. He encouraged me to be a strong, independent thinker. I would like to thank him for challenging me, believing in me and his fatherly support through the tough times.

I would like to thank all my committee members: Professor Charlie Bouman, Professor Mireille Boutin, Professor David Ebert and Professor Zygmunt Pizlo for their advice and encouragement. I especially thank Professor Pizlo for devoting his valuable time to give me ideas and insights about conducting user studies from the disciplinary perspective of psychology. These user studies were crucial to my project's validation.

I would like to thank Professor Hong Tan for allowing me to use her research laboratory and the eye tracker equipment for my research. I also appreciate the great help I received from her graduate students especially, Chanon Jones, on how to work with the equipment.

I am deeply grateful to my many friends at Purdue University, especially the members of VIPER laboratory: Marc Bosch, Ying Chen, Dr. Nitin Khanna, Deen King-Smith, Dr. Liang Liang, Dr. Limin Liu, Kevin S. Lorenz, Ashok Mariappan, Anand Mariappan, Dr. Anthony Frank Martone, Aravind Mikkilineni, Ka Ki Ng, Francisco Serrano, Satyam Srivas and Fengqing (Maggie) Zhu. Without their presence I would not have had such a friendly and supportive work environment.

I feel very fortunate to come to Purdue and meet a diverse group of wonderful and talented people who have opened my horizons to different perspectives. I am a better person because of my great friends.

I would like to thank the Motorola Research Laboratory for funding this project. I would especially like to thank Dr. Cuneyt Taskiran and Mr. Kevin O'Connell from Motorola for their advice and support on the project.

I would also like to thank all my teachers and my professors at Sharif University of Technology in Iran. They have had a great impact for building the foundation of my education and encouraging me to continue and expand my studies at the graduate level.

Finally, I wish to thank the support and encouragement that I have received for many years from my family, especially my mother and father. Without their unconditional love, support, and trust, I would never have been able to follow my ambitions and achieve my goals as a student thousands of miles from home.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABBREVIATIONS

API	Application Programming Interface
DoG	Difference of Gaussian
GLA	Generalized Lloyd Algorithm
GLCM	Gray Level Co-occurrence Matrix
GPS	Global Positioning System
MB	Macro Block
MVF	Motion Vector Field
RGB	Red Green Blue
ROI	Region Of Interest
SIFT	Scale Invariant Feature Transform
SSIM	Structural SIMilarity
SVM	Support Vector Machine
UGV	User Generated Video

ABSTRACT

Abdollahian, Golnaz. Ph.D., Purdue University, December 2009. Content Analysis Of User Generated Video. Major Professor: Edward J. Delp.

Due to the availability of online repositories, such as YouTube and social networking sites, there has been a tremendous increase in the amount of personal video generated and consumed by average users. Such video is often referred to as *user generated video* (UGV) or consumer video as opposed to *produced video*, which is produced and edited by professionals. Examples of produced video are television programs, movies, and commercials. The large amount of user generated content available has increased the need for compact representations of UGV sequences that are intuitive for users and let them easily and quickly browse large collections of video data, especially on portable devices, such as mobile phones. UGV has special properties that are distinct from the properties of produced video, which makes the content analysis for this type of video more challenging. Many analysis techniques that are designed for produced video can not be directly used for UGV.

In this thesis we developed a system for the analysis of UGV. The system consists of two major components: camera motion-based video summarization and video annotation based on location information. UGV often has a rich camera motion structure that is generated at the time the video is recorded by the person taking the video, i.e. the "camera person." We exploit this structure by defining a new concept known as *camera view* for temporal segmentation of UGV. The segmentation provides a video summary with unique properties that is useful in applications such as video annotation. Camera motion is also a powerful feature for identification of keyframes and regions of interest (ROIs) since it is an indicator of the camera person's interests in the scene and can also attract the viewers' attention. We examined the effect of camera motion on human visual attention through an eye tracking experiment. The results showed a high dependency between the distribution of fixation points of the viewers and the direction of camera movement. Based on the results of this experiment, we introduced a new location-based saliency map which is generated based on camera motion parameters. This map is combined with other saliency maps generated using features such as color contrast, object motion and face detection to determine the ROIs. The output of the summarization step is a set of keyframes with highlighted ROIs. Several user studies were designed and conducted to assess the performance of our system. The subjective evaluation indicated that our system produces video summaries that are consistent with viewers' preferences.

The video annotation part in our proposed system generates a set of tags or keywords for the video which is provided to the user along with the video summary. Our proposed annotation approach uses the location metadata of the video and visual features to find tags that are most likely relevant to the video in a user-tagged image database. We examined different image matching methods to be used to measure the visual similarity between the video keyframes and images in the database. These methods have been tested and compared to each other over a variety of different cases.

1. INTRODUCTION

1.1 User Generated Video: Properties And Challenges

User generated video content is the new media to distribute the enormous amount of information among people in the world. Millions of videos are generated, shared and watched by people everyday. Such video is often referred to as *user generated video* (UGV) or consumer video as opposed to *produced video*, which is produced and edited by professionals, e.g. television programs, movies, and commercials. Availability of several online repositories and social networking websites such as Youtube has changed the pattern for video production, distribution and consumption. The length of videos has been adapted to today's fast-pace life. Recent videos generated by users are much shorter, on the order of a few minutes or even seconds. The video production time has been also drastically decreased compared to some years ago [1]. Camcorders have become smaller and lighter and most handheld devices (e.g. mobile telephones) are capable of recoding videos.

The possibility of posting a video on almost any website has led to rapid circulation of popular videos. Websites with simple and intuitive interfaces and equipped with powerful search engines make it convenient to access information, in the form of video, swiftly and at any time of the day. This differs from older structures of information access available to the public 20, or even 10 years ago. This has lead to UGV being used for different purposes such as personal music videos, digital diaries, self- publishing news, advertisements for small businesses and even digital resumes. This media is a shortcut to many paths that were barely possible for an average person a few years ago. Thanks to UGV, anyone can produce a music video and advertise it on a variety of social networking websites. In turn this “published” (in the sense of posting online) video can generate its own fans. This can be attested to

Fig. 1.1. Multimedia devices.

by the fact that there are Youtube celebrities that are sometimes more well known than “conventional” celebrities.

With the tremendous increase in the amount of user generated video, users can be burdened with a large collection of video on their devices that is hard to manage. Therefore, it is crucial to have access to compact representations and intelligent ways of indexing UGV sequences that are intuitive for users to let them easily and quickly browse large collections of video data, and search for a specific video, especially on portable devices, such as mobile phones.

Due to its characteristics, when compared to produced video, analyzing UGV presents unique challenges. Produced video has a rich shot–scene–sequence syntactic structure that is created by video editors who follow a well-known set of editing guidelines that are genre specific. In this type of video, properties of shots, e.g. types of camera views and distribution of shot durations, are related to the content and can be exploited to provide clues about the relative importance of a shot and its content. Therefore, most video analysis algorithms use the video shot as the basic building block and begin by segmenting the video into shots [2, 3].

UGV sequences typically are unedited and unstructured, where each UGV clip can be considered as a one-shot video sequence determined by the camera start and stop operations. The lack of a well-defined syntactic structure precludes the use of most content-based video analysis approaches for UGV.

However, UGV generally has a rich camera motion structure that is generated by the person taking the video, i.e. the "camera person" who "edits" the video in real-time by moving the camera, e.g. rather than having a general shot and then cutting to an interesting object, a camera zoom or a pan to the object is used. Such content-based camera motions are not typical in produced video where the same effect is obtained through video editing.

1.2 Overview of Our System

In this thesis we developed a system for UGV content analysis that

- generates a video summary as a keyframe representation
- identifies regions of interest (ROIs) in the keyframes to make it suitable for visualization on devices with small-size screens such as mobile phones
- determines a set of tags or keyframes for automatic annotation of the video

An overview of our system is shown in Figure 1.2. The system consists of two major components: camera motion-based video summarization and video annotation based on location information. The video summarization component exploits camera motion for temporal segmentation, keyframe selection, and also combines it with other features for identification of regions of interest. The output of this component is delivered to the user as a video summary and is also an input to the annotation component. The annotation component aims to tag the video using an approach that does not require training and object recognition. Our method finds images that are similar to the video keyframes, both visually and location-wise, in a separate anno-

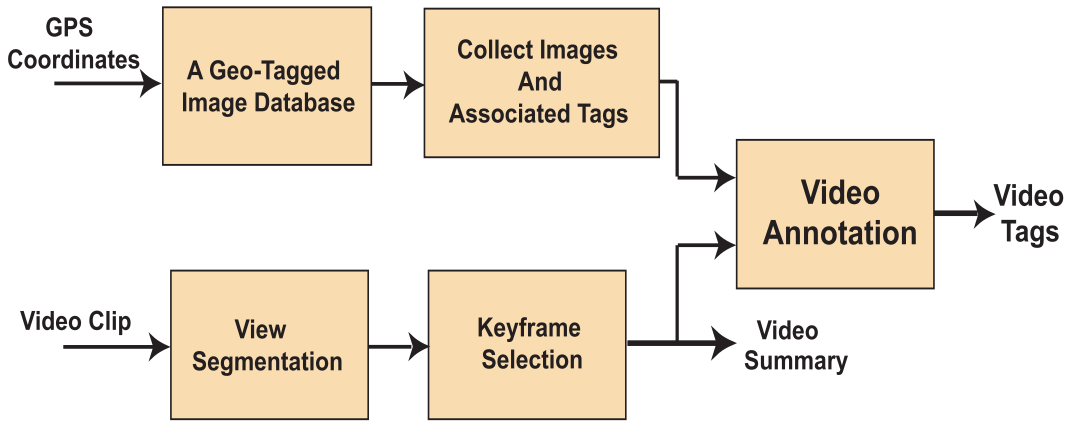


Fig. 1.2. Overview of our user generated video content analysis system.

tated image database. These images are used to find the most relevant tags to the video.

Figure 1.3 illustrates a block diagram of the motion-based video summarization system. The global motion is extracted from the video and classified into unintentional and intentional motion as a crucial preprocessing step in the analysis process. The intentional camera motion is then used for the content analysis. While one could use many other properties or features to analyze UGV, the goal here was to examine what can be done by mainly focusing on the camera motion.

Our temporal segmentation method makes use of camera motion by dividing the video into *camera views* which we define as the basic units of UGV. The video is segmented into different views that the camera is "observing" during displacement or change of viewing angle with respect to the scene. This segmentation has the special property that by selecting at least one frame from each segment in the video summary, we obtain a notion of all the scenes captured by the camera. Therefore, even with a simple keyframe selection strategy, the video summary will cover all the camera views. This property does not hold for methods that are based on other low-level features such as color [4]. In some video sequences the color distribution does not have significant variations throughout the video. There may be a few colors dominating the color distribution in all the views. The segmentation or summarization methods

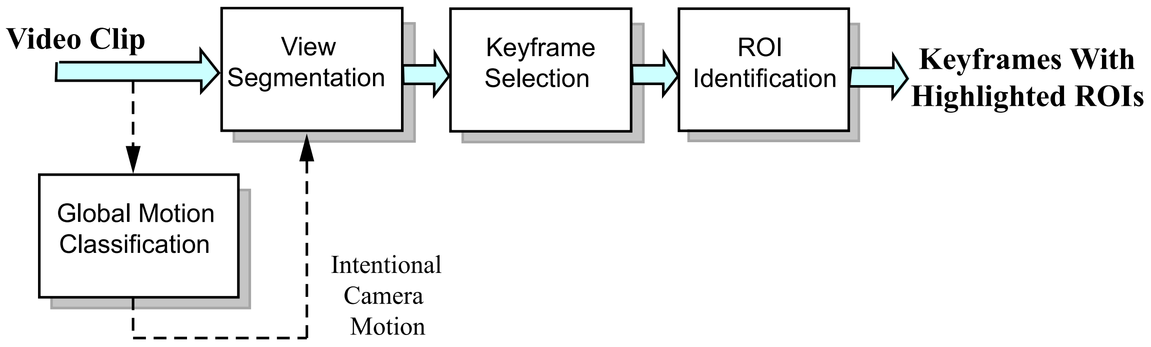

Video Clip
View Segmentation
Keyframe Selection
ROI Identification
Keyframes With Highlighted ROIs
Global Motion Classification
Intentional Camera Motion

Fig. 1.3. Block diagram of the video summarization method based on camera motion.

that are based on measuring the difference between the color distributions among the frames fail to capture all the scenes in this kind of video sequences. Our segmentation approach measures the displacement of the camera and is capable of detecting the changes in the camera view. This method of summarization is particularly useful for video annotation where we can associate location-based tags to the video based on the backgrounds and views without missing any scenes.

Since a majority of UGV is recorded, shared and downloaded and watched on mobile devices such as cellular phones, our goal was to design methods that suitable for this type of devices. Having efficient video representation for such devices is even more crucial due to the limited bandwidth, battery life and storage. Therefore, our goal is to use computationally efficient methods in each part of the system. Another challenge in dealing with video content on mobile devices is displaying the video or its summary on a small size screen. Most video summarization approaches, especially the ones that represent video as hierarchical trees and storyboards [5–8], are not suitable for such devices. Therefore, we employ a human attention model to extract saliency information within keyframes and identify regions of interest (ROIs) i.e. regions that capture the attention of a viewer while watching the video. In this way, the results are more consistent with the viewer's perception without using semantic analysis. We propose a new location-based saliency map which uses camera motion information to determine the saliency of pixels with respect to their spatial location in the frame. This map is combined with saliency maps based on other cues such as color contrast, local motion and face detection to extract ROIs in the keyframes.

Several user studies were conducted to verify our hypotheses and methods. A subjective study was done to validate our end-to-end summarization system. This study showed a high correlation between the output of our system with the user-selected results. In addition, we explored the effect of camera motion on the human visual system through an eye tracking experiment where we recorded the eye movements of subjects while they watched videos with various types of camera motion. The study was designed in a way so as to minimize the effect of visual features other than camera motion. The experiment showed how different types of camera motion led to different patterns in the distribution of eye fixation points, indicating high dependency between the direction of camera motion and visual saliency in video. Various statistical tests done on the collected data verified the statistical significance of the results. The result of this study was used to generate a novel location-based saliency map that was combined with other saliency maps in our system to identify viewer's regions of interest in video.

Video annotation is another challenging task in video content analysis that allows users to index their videos and be able to easily access them when they need. Manual annotation or "tagging" of a video is difficult and is usually done by providing a general description of the content and occasion of the video. These descriptions usually do not include the details of the scenes. Most previously proposed approaches for automatic video annotation are limited to a predefined set of concepts [9, 10]. The annotation is generally achieved through supervised learning and a classifier training process is needed for each concept. Recently, a number of methods have been proposed for multimedia annotation that exploit Web-based tags in order to extend the vocabulary and avoid supervised learning [11, 12].

Millions of pictures are uploaded to the photo sharing websites such as Flickr and Picasa everyday. These pictures are often associated with various “tags” or keywords including information about the content, time, date, camera used and more recently geo-location. In our system we exploit the data from such websites for UGV annotation. We assume that the location information of the video is provided as metadata. The system finds tags that have high probability of being relevant to the video without high-level complex analysis such as object and action recognition of the video sequence. It is also flexible in the sense that each step of the process can be enhanced independently without necessarily changing the structure of the system.

1.3 Literature Survey

Before describing our methods in more detail, we provide a review of techniques for video content analysis from the literature. There are several tasks that are crucial in the content analysis of video sequences. These tasks are the following.

1.3.1 Video Segmentation

The first task in most video content analysis systems is extracting the syntactic structure of a video sequence in the temporal domain and segmenting the sequence into smaller temporal units that are easier to manage.

Produced video sequences usually consist of several scenes that can be further divided into one or more video shots. A *shot* can be defined as a group of frames that has continuity in some general conceptual or visual sense. In most analysis approaches, the first step is the temporal segmentation of the sequence into separated shots. There are two types of transition from one shot to another: *abrupt* and *gradual*. An abrupt transition or camera cut occurs when the last frame of one shot is followed immediately by the first frame of the next shot. This type of shot boundary can be detected by comparing adjacent frames. In a gradual transition, it takes several frames to move from one shot to the next so there is no significant change between the consecutive frames during the transition. Thus, this type of transition is more difficult to detect and requires monitoring the visual changes in a longer time interval. The most frequent types of gradual shot transitions are *fade-in, fade-out, dissolve* and *wipe*. A fade in is a shot transition where the frame is dark in the beginning of the transition and gradually an image appears, brightening to full strength; a fade out is the opposite of a fade in; a dissolve is the superposition of a fade out and a fade in [13]; and, a wipe is a moving transition of a frame (or a pattern) across the screen that enables one shot to gradually replace another [2].

The shots can be further segmented into sub-shots and objects and can be represented by selecting a set of keyframes. Moreover, using similarity measures, shots can be clustered and organized to form scenes. Several methods have been proposed for shot boundary and scene change detection [2–4, 6, 7, 14–16]. Here, we review some of these methods. Basically, these methods vary in two major aspects: features being used for similarity measurement and whether or not they work in the compressed domain.

Most approaches use the color histogram [4, 5, 15, 16], edge properties [5, 6], motion information [15], luminance projection [16] or a combination of the above mentioned features [3, 5, 15] for distance measurement between the frames. Among these methods, color-histogram-based approaches give superior performance for abrupt shot or scene transition [2].

Methods that perform directly on the compressed video can save both computational time and memory space. For example, some methods employ DC images instead of the original-size frames, which are available for I frames in an MPEG encoded video and can be extracted for P and B frames [3, 4, 7]. There are also motion-based approaches that use motion vector fields (MVF) extracted from the motion vectors in MPEG stream in order to characterize camera operations [17].

Scene change detection is another challenging research topic in video analysis. A scene is composed of a series of consecutive shots that are coherent from the narrative point of view. They are either shot in the same place or share a similar thematic content [2]. The problem of scene change detection is how to group similar shots into different scenes. Some of the reported methods for scene change detection cluster the video shots based on both visual similarity between the shots and temporal adjacency and duration that reflect the localities of video contents in time [6,7]. The method in [7] clusters the similar shots together if they fall within a time window with a specified length. A scene transition graph (STG) is then constructed from the clustering results and the temporal relationships of the shots in the clusters. A node represents a cluster and an edge shows the flow of the story from one node to another. The story units are extracted by finding the *cut edges* of the STG. Removing each cut edge results in division of the STG graph into two disconnected subgraphs. Another approach is to employ audio features to distinguish different scenes, for example, in [18] a scheme for identifying scenes by clustering shots according to detected dialogs, like settings and similar audio is described.

Other approaches for video segmentation and keyframe selection employ the clustering of video frames [3, 19]. In these methods, frames are represented in a high dimensional vector space using low-level visual features and a distance metric is used to measure the similarity between the clusters. After clustering, the centroid of each cluster is chosen to form the keyframe summarization of the video. For example, in [3] the frames are clustered in a bottom-up fashion by combining the closest clusters at a time.

1.3.2 Video Summarization

Video summarization is an important video analysis task that exploits the structure of video to make the video easier to manage. This is done by decreasing the temporal redundancy. A great deal of research has been done in this area [20]. There are several ways to represent a video summary including mosaic images [8, 21], a set of keyframes [3, 16, 19], key objects [22] or a reduced-length video [14]. The crucial task in video summarization is identifying what is "important" or "salient" in a video. Finding a generic definition for visual importance is difficult due to the lack of tools for modeling human cognition and affection. As a result, many researchers have limited their attention to specific domains, such as news [23,24] or sports [25–27] programs that have predefined events as their highlights.

Some schemes for identifying highlights in UGV sequences are semi-automatic and dependent on manual selection of important segments [5, 22, 28]. For example, the system in [28] allows users to select their favorite clips in a sequence and the desired output total length and provides the appropriate boundaries for each clip based on the suitability of the analyzed video material and the requested clip length.

Some recent approaches use human attention models to define visual saliency or importance based on what captures a viewer's attention while watching a video [29–31]. Human attention models are used to create saliency maps that indicate how viewers are visually attracted to different regions in a scene and identify the regions of interest (ROIs). There are several applications that use ROIs for image and video adaptation to smaller screens. One such application is described in [32] where ROIs are used to model the information structure within the frames and generate browsing paths. Chen *et al.* [33] use a human attention model to create different cropped and scaled versions of images according to the screen size. Several factors such as contrast, size, location, shape, faces, foreground/background and local motion activities can influence visual attention and have been used to identify ROIs in images and videos [29, 34–37]. Most previous attention models have not considered camera motion as an independent factor in identifying visual saliency within a frame. For example, in [31] the global motion is subtracted from the macroblock motion vectors to obtain relative motion which is used for generating temporal saliency maps. Ma *et al.* in [30] use a camera attention model to assign an attention factor to each frame based on the type of motion and velocity of the camera but this factor is the same for all pixels within a frame. As we will show in this thesis, camera motion plays an important role in attracting viewer's attention while watching a video. This feature has been previously used in video summarization approaches as an indicator of the camera person's levels of interest [38, 39]. Here, we will show that this feature has also a major influence on viewers' visual attention and therefore can be used as a powerful tool for UGV content analysis.

1.3.3 Video Annotation And Retrieval

Most previously proposed approaches for automatic video annotation are limited to a predefined set of concepts [9, 10]. In these methods, features from different channels such as visual, auditory and textual modalities, are combined to detect a number of objects and events. The detection is generally achieved through supervised learning and a classifier training process is needed for each concept. These methods are restricted to a limited number of concepts and are computationally expensive.

Recently, a number of methods have been proposed for multimedia annotation that exploit Web-based tags. These annotation techniques exploit the user-generated tags in order to extend the vocabulary and avoid the supervised learning [11, 12]. For example, the system in [11], annotates images using semantically and visually similar images on the Web. The assumptions are that semantically similar images share similar keywords and that the image contains at least one accurate keyword. The work in [12] uses search and mining techniques to annotate news videos based on similar annotated videos. We proposed a method for automatic annotation of UGV that uses geo-tagged image databases. This method finds tags that have high probability of being relevant to the video in the image database. Our approach does not require any high-level complex analysis such as object and action recognition of the video sequence.

1.4 Overview Of The Dissertation

This dissertation describes the system we developed for content analysis of user generated video [38,40–43]. The main contributions of this dissertation are as follows.

1.4.1 Contributions Of The Dissertation

- **General properties of the system -** The focus of this system is mainly on user generated video. This type of video has special characteristics that introduces uniques challenges for the content analysis. Some of these characteristics are the lack of a scene-shot-frame syntactic structure, presence of noise and abnormal camera motion (e.g. blurry and shaky motions), lack of a clear storyline and editing that is usual for produced video, and diversity in the subject. We selected methods and features that exploit the general properties of UGV. Thus, as the subjective evaluations show, the outcome of our system is consistent with viewers preference as well as the camera person's interests.

  The system is designed so that it can be used for handheld devices. This factor implies two constraint on the problem; the methods used need to be computationally efficient and the results must be easy to visualize on a small-sized screen. Thus, the techniques and features used in each step of the analysis are light-weight and fast, but produce good quality results that satisfy users. Identifying ROIs in the keyframes enables the user to view only the important regions of the keyframes on their devices instead of the entire keyframe.

- **Use of camera motion -** Camera motion is a useful feature that can be extracted from a video sequence fairly easily. Several methods have been proposed in the literature to estimate and classify the camera motion. This feature is intuitive for UGV content analysis since UGV often has a rich camera motion structure used by the camera person to "edit" the video while shooting it. We conducted an eye tracking experiment on the effect of camera motion on human visual attention. This study showed that camera motion attracts viewers attention to a specific part of the frame based on the pattern of the motion. Therefore, we use this feature throughout the system. A new video segmentation method was proposed that is based on camera displacement in a video. We also proposed a new location based saliency map that is generated using camera motion parameters. This saliency map accounts for the effect of camera motion on attracting viewers' attention when ROIs in video frames are identified. Camera motion has also been used in the keyframe selection step in our system as an indicator of camera person's interests.

- **Temporal segmentation of UGV -** The new temporal concept of *camera view* was introduced that can be considered as the building block of a UGV. A video is segmented into camera views by identifying the view transitions based on the camera displacement. Each segment is devoted to one of the views that the camera has captured. Due to the fact that at least one keyframe is included in the video summary to represent each camera view, the video summary covers all the scenes viewed by the camera. Almost any other frame of the video has overlap with at least one of our keyframes. In a scene, the set of keyframes extracted by our system expands the entire video.

  Another advantage of this approach for segmentation is that we can use the video keyframes instead of the entire sequence for annotation since they have the notion of all the scenes. This is particularly useful for our annotation method since we use image databases for this purpose and it is easier to compare individual frames with the images in the database.

- **User Studies -** In addition to the eye tracking experiment to study the effect of camera motion, we conducted several other user studies to evaluate the summarization and ROI detection results. In the area of video content analysis, the quality of the results often can only be measured subjectively. In a user study, it is important to define the user tasks in a way to minimize the ambiguity of the task for the user. It is also crucial to have a controlled environment so that the effect of unrelated factors are minimized. In some approaches that were previously proposed in the literature, the authors have evaluated their results by asking users to rank the results using a predefined set of terms such as "good", "fair" and "bad". However, this approach may not be appropriate since these term are relative. We carefully designed and conducted our user studies so that the results are conclusive and statistically significant.

- **UGV annotation -** Due to the diversity in the content of UGV, annotation methods that need supervised learning and are limited to a predefined number of concepts do not have the sufficient vocabulary to tag the video. We used the geo-location information of video to first gather images taken in the same location as the video from an image database. Two image matching approaches were then examined in order to find the most visually similar images within the collected images to the video keyframes. Tags from the final set of images were ranked based on their frequencies and the most dominant tags were associated to the video.

1.4.2 Organization Of The Dissertation

This dissertation is organized as follows.

Chapter 2 describes our approach for UGV summarization based on camera motion. First step in the analysis is the global motion modeling and classification. The camera motion parameters are estimated and used in several steps in the system. Camera motion in video frames are labeled into various normal and abnormal motion patterns. Our methods for temporal segmentation of UGV into camera views and generating the video summary, in the form of a keyframe set, are also presented in this chapter.

To explore the information structure within the keyframes and identify the ROIs we consider several factors to develop saliency maps for the extracted keyframes (Chapter 3). The identified ROIs are highlighted in the keyframes that are delivered to the user as a video summary. Our eye tracking experiment to examine the effect of camera motion on human visual attention is also described in Chapter 3.

The experimental results for the generated video summary and ROI detection, comparisons to other well-known methods and the result from the user studies are reported in Chapter 4.

Chapter 5 describes our methods for automatic annotation of UGV using the geo-location information and user-tagged image databases.

Finally, concluding remarks and ideas for future work are discussed in Chapter 6.

2. MOTION BASED SUMMARIZATION

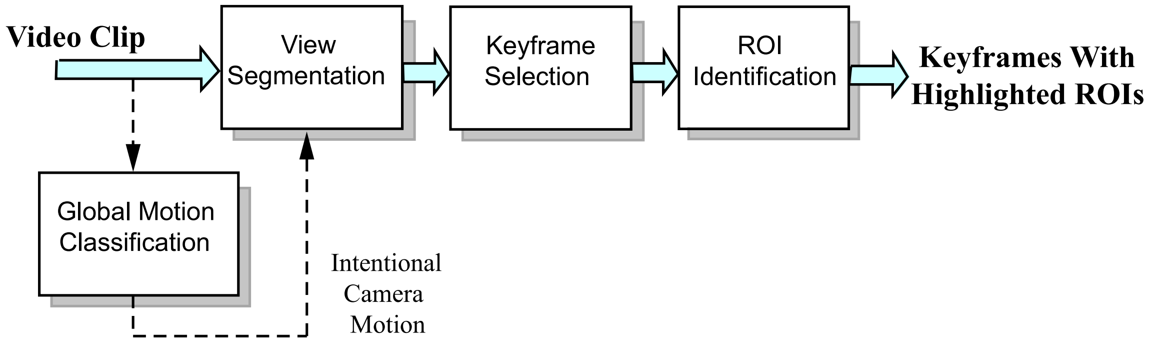

Video Clip
View Segmentation
Keyframe Selection
ROI Identification
Keyframes With Highlighted ROIs
Global Motion Classification
Intentional Camera Motion

Fig. 2.1. Block diagram of the video summarization method based on camera motion.

Our video summarization approach focuses on the use of camera motion for temporal segmentation and keyframe extraction (See Figure 2.1). First, we estimate the parameters for the global motion model. Then, the video frames are labeled based on a number of features extracted from the motion parameters in order to classify the motion and distinguish between the intentional motion and the abnormal motion. Camera motion is then used for segmenting the video into “camera views” as we will explain. Consequently, A set of keyframes are extracted from the camera views to form the video summary.

2.1 Global Motion Estimation

In produced video, camera movement has six degrees of freedom: tracking, dollying, and booming for translational movement along the coordinates, and tilting, rotating, and panning for rotational movement about these axes. Estimating motion parameters in this case is computationally expensive. However, to estimate the camera motion in a user generated video (UGV), we assume how a camera is typically used. In the majority of UGV, camera motion is limited to a few operations, e.g. pan, tilt, and zoom. More complex camera movements, such as rotation, rarely occur in UGV. Several motion models and estimation methods have been proposed in the literature for global motion estimation and camera motion characterization [44–46]. However our goal here is to be computationally efficient. Therefore, we use a simplified 3-parameter global camera motion model in the three major directions, horizontal (H), vertical (V), and radial (R). This model adequately describes the majority of the camera motion we have observed in UGV. The motion model is defined as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = R \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} H \\ V \end{pmatrix}, \tag{2.1}$$

where (x, y) and (x', y') are the matched pixel locations in the current and reference frame, respectively.

The Integral Template Matching algorithm [47] is used to estimate the motion parameters. In this method a template, T, in the current frame is matched against the previous frame using a full search in a predefined search window. The template is illustrated as the white region in Figure 2.2. The central part of the frame is excluded from the template to avoid the effect of object motion close to the frame center. This is motivated by the observation that if the moving object is of interest to the camera person, it will most likely be close to the frame center and if not, it usually does not stay in the camera view for a long time. Local motion in the center area greatly affects the accuracy of estimation especially for radial motion parameter [47]. In order to avoid this effect without increasing the computational complexity, we do not include the frame center in the template. Pixels close to frame boundary are also removed from the template because of errors in the pixel values due to compression and camera aberration.

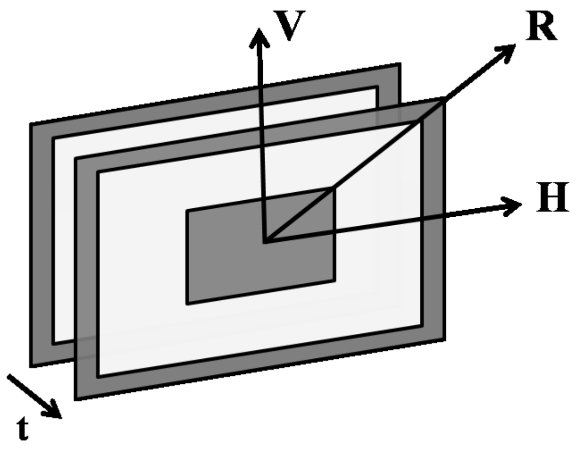


Fig. 2.2. Template used for motion parameter estimation is shown as the white area.

The parameters H, V and R are estimated by minimizing the $L1$ distance between the 2-D template in the current frame, T_i, and the previous template transformed using parameters H, V and R which is denoted by $\mathcal{F}\left(T_{i-1}, H, V, R\right)$.

$$(\hat{H}_{i,i-1}, \hat{V}_{i,i-1}, \hat{R}_{i,i-1}) = \arg \min_{(H,V,R)} \left|T_i - \mathcal{F}\left(T_{i-1}, H, V, R\right)\right| \tag{2.2}$$

In order to accelerate the template matching process, we estimate the initial values for H and V using the Integral Projection method [48]. In this technique, the 2-D intensity matrix of each frame is projected onto two 1-D vectors in the horizontal and vertical directions. Integral Projection is a special case of the Discrete Radon Transform which has been used to identify the global motion [46]. We define the projection vectors of video frame i onto the x and y axes as

$$\begin{aligned} \mathrm{Proj}_x(i,x) &= \frac{1}{N}\sum_{y=1}^{N} I_i(x,y) \\ \mathrm{Proj}_y(i,y) &= \frac{1}{M}\sum_{x=1}^{M} I_i(x,y) \end{aligned} \tag{2.3}$$

where $I(x,y)$ is the intensity of pixel at (x,y) and N and M are the frame height and width, respectively. The initial values, H_0 and V_0 parameters are then estimated between consecutive frames by matching the projections in each direction using Mean Absolute Difference (MAD) minimization as the matching criteria.

$$\begin{aligned}\hat{H}_{i,i-1} &= \arg\min_{H} \sum_{x} \left|\mathrm{Proj}_x(i,x) - \mathrm{Proj}_x(i-1,x-H)\right| \\ \hat{V}_{i,i-1} &= \arg\min_{V} \sum_{y} \left|\mathrm{Proj}_y(i,y) - \mathrm{Proj}_y(i-1,y-V)\right| \end{aligned} \tag{2.4}$$

The search ranges for H and V are limited to $[-30\ 30]$ and $[-15\ 15]$, respectively for videos of size 360×240 pixels. We experimentally found that the values larger than these ranges will result in highly blurred frames and will be classified as blurry in the next section. Therefore, the parameters that fall outside the range have the same effect as the ones that have maximum value. Once the values of H_0 and V_0 are obtained, a parabolic fitting is preformed to modify these values to the resolution of a fraction of a pixel.

The template matching starts from the initial point $(H_0, V_0, 0)$ and iterates through the values in the search window. The iteration stops when a local minimum is found. Figure 2.3 illustrates an example of H, V and R functions in a typical UGV with different camera operations. As the figure shows, during zoom operations, R has non zero value that does not change sign for at least a few frames. When the camera has shaky motion, H, V (and sometimes R) have fluctuations around zero. During a fast camera motion H or V or both functions, have large amplitude. These facts are useful in identifying the camera motion behavior, as described in the next section.

2.2 Motion Classification

After the camera motion parameters are estimated, video frames are labeled based on the type of motion as a preprocessing step in the analysis (See Figure 2.4). Camera motion in UGV usually contains both intentional motion, e.g. pan, zoom and tilt, and unintentional motion, such as shaky and fast motion. Fast camera motion causes blurriness in video frames. If the blurriness is so high that the frames have no visual information, they should not be included in the video summary. Shaky motion is due

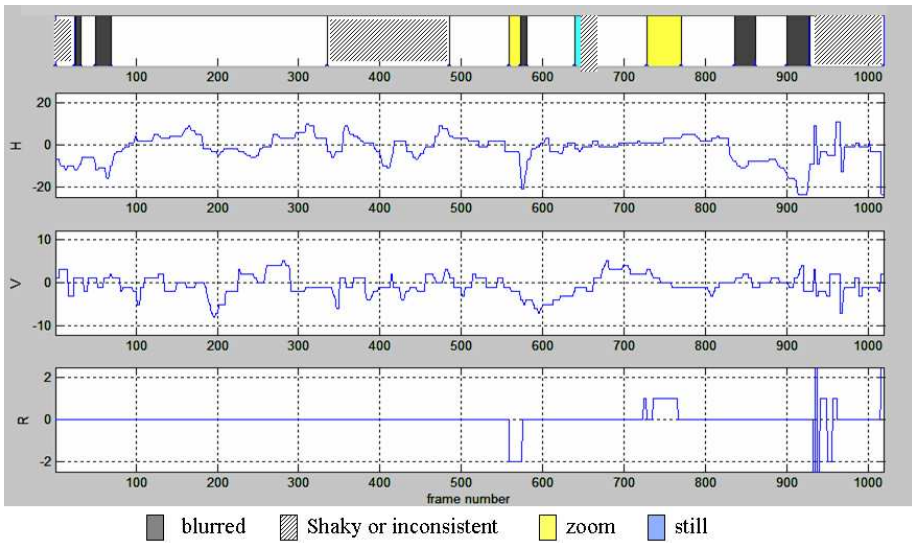


Fig. 2.3. An example of H,V and R versus frame number for a UGV.

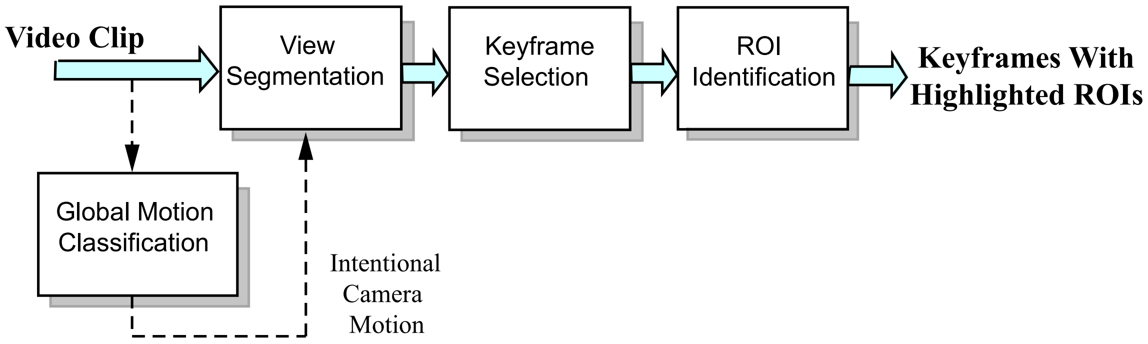

Video Clip
View Segmentation
Keyframe Selection
ROI Identification
Keyframes With Highlighted ROIs
Global Motion Classification
Intentional Camera Motion

Fig. 2.4. Block diagram of the video summarization method based on camera motion.

to frequent change of camera motion which is another unintentional camera motion pattern.

While intentional camera motion provides valuable cues about the relative importance of a segment, unintentional motion decreases the perceived quality of the video and can be misleading in the analysis. In our system video frames are classified into one of four classes using the decision tree structure shown in Figure 2.5.

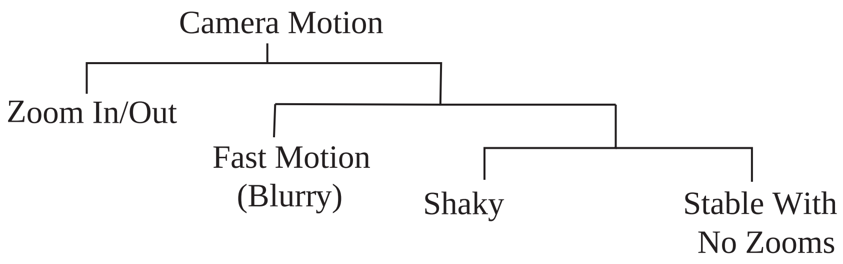


Fig. 2.5. Decision tree used to label video frames.

Due to their superior performance, we have used Support Vector Machine (SVM) classifiers [49] at each decision node of this tree. The LIBSVM library was used, which is an open source software library for SVM training and classification [50]. Forty video sequences, each with duration of a few minutes, were manually labeled and used as the training data. These sequences were all UGVs recorded by several users and contained different types of camera motion including both normal and abnormal motions. The video sequences were from a wide variety of outdoor and indoor scenes, e.g. playgrounds, parks, university campuses, beaches, museums, ranches and inside various buildings.

In order to label a frame we first classify it as having a zoom or not, using the 3D motion vector (H, V, R) as the feature vector. Then, blurry frames caused by fast camera motion and shaky segments caused by frequent change of camera motion are detected sequentially.

For blurry and shaky motion detection we use a method similar to [51] where SVM classifiers are trained on an 8-dimensional feature vector derived from the parameters H and V over a temporal sliding window. The eight features include average velocity, S_x and S_y, average acceleration, A_x and A_y, variance of acceleration, σ^2_{Ax} and σ^2_{Ay}, and average number of direction changes, D_x and D_y in vertical and horizontal directions. These feature values are

$$S_x = \frac{1}{N-1}\sum_{i=1}^{N-1}|H_{i,i-1}|, \qquad S_y = \frac{1}{N-1}\sum_{i=1}^{N-1}|V_{i,i-1}| \tag{2.5}$$

$$A_x = \frac{1}{N-2}\sum_{i=1}^{N-2}|H_{i,i-1} - H_{i+1,i}|, \qquad A_y = \frac{1}{N-2}\sum_{i=1}^{N-2}|V_{i,i-1} - V_{i+1,i}| \tag{2.6}$$

$$\sigma^2_{Ax} = \mathrm{Var}_{i=1}^{N-2}\,|H_{i,i-1} - H_{i+1,i}|, \qquad \sigma^2_{Ay} = \mathrm{Var}_{i=1}^{N-2}\,|V_{i,i-1} - V_{i+1,i}| \tag{2.7}$$

$$D_x = \frac{1}{N-2}\sum_{i=1}^{N-2} f(H_{i,i-1}, H_{i+1,i}), \qquad D_y = \frac{1}{N-2}\sum_{i=1}^{N-2} f(V_{i,i-1}, V_{i+1,i}) \tag{2.8}$$

where

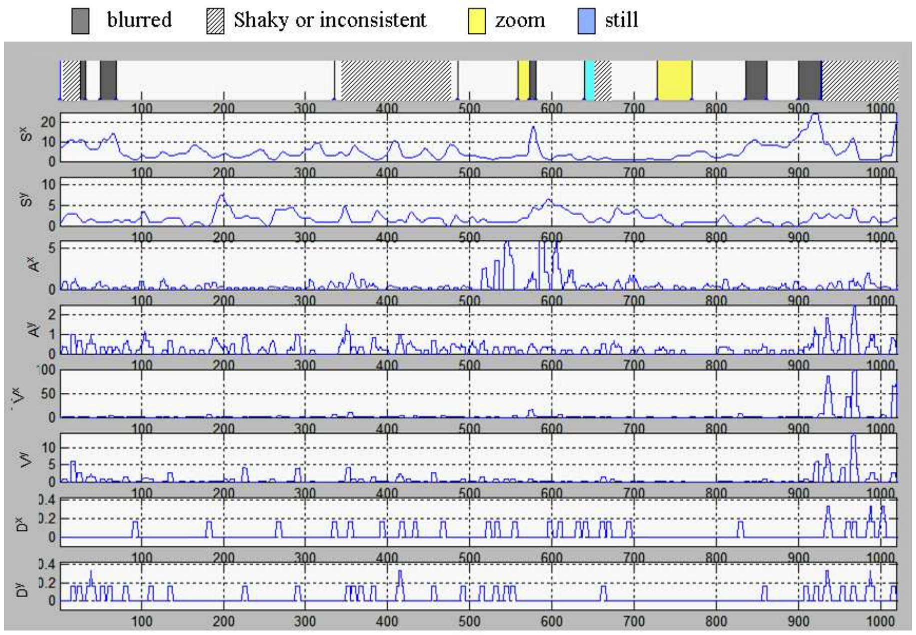
$$f(u,v) = \begin{cases} 1, & \text{if } \operatorname{sgn}[u] = \operatorname{sgn}[v] \\ 0, & \text{otherwise.} \end{cases}$$

Where, N is the size of the sliding window. The size of the sliding window is different for blurry and shaky motion classification. This is due to the fact that fast camera motion that causes blurriness may last for just a few frames but shaky behavior occurs during a period of time when the camera changes direction frequently. Therefore, to detect this behavior, camera motion must be monitored during a relatively longer time interval. The appropriate window sizes were experimentally found to be $N = 7$ for blurry motion and $N = 31$ for shaky motion detection. The frames that are not labeled as zoom, blurry or shaky are identified as stable motion with no zooms.

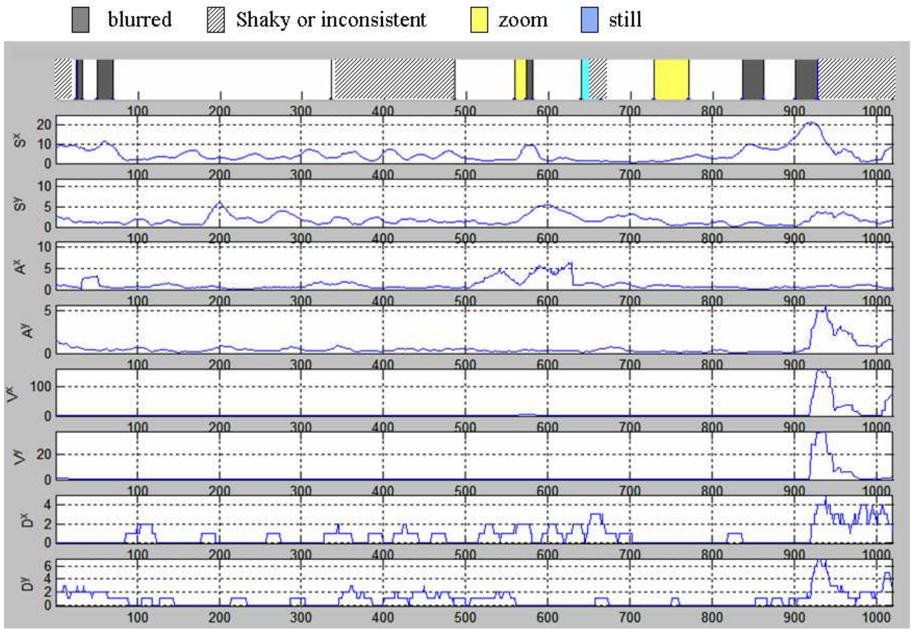
Figure 2.6 illustrates the eight features $S_x, S_y, A_x, A_y, \sigma^2_{Ax}, \sigma^2_{Ay}, D_x$ and D_y for a typical UGV clip using these two different window sizes corresponding to fast motion and shaky motion detection. The ground truth motion labels are shown at the top of the Figure. In part (a) the features are obtained over a sliding window of size $N = 7$ frames for blurry motion classification. It can be seen from the Figure that, average velocity functions, S_x and S_y have large values when fast camera motion occurs (the dark areas in the figure).

Part (b) shows the features over a sliding window of size $N = 31$ frames for shaky motion classification. As the Figure shows, when the camera has shaky motion (the shaded regions in the figure), one or both of the average direction change functions, D_x and D_y have large values. These features can be a good indicator of shaky behavior of the camera .

Some examples of video frames that are automatically labeled by this approach are shown in Figure 2.7.



(a)



(b)

Fig. 2.6. The eight features (from top to bottom: $S_x, S_y, A_x, A_y, \sigma^2_{Ax}, \sigma^2_{Ay}, D_x$ and D_y) used for (a) fast motion detection and (b) shaky motion detection in a UGV clip. The temporal window sizes are $N = 7$ frames in part (a) and $N = 31$ frames in part (b).

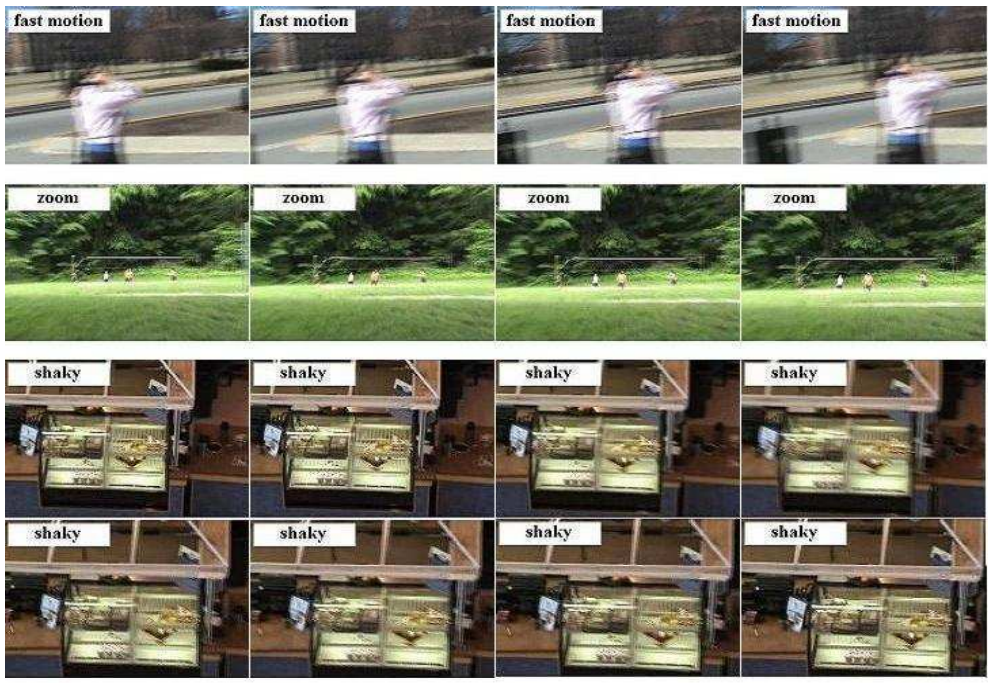


Fig. 2.7. Automatically Labeled Frames. In the top two rows, each row consists of four consecutive frames classified in the same motion group. The bottom rows are a set of eight consecutive frames labeled as shaky motion.

2.3 Video Summarization

Many different types of video summary representations, collectively known as *summary visualizations*, have been proposed [3, 8, 14, 16, 19–22]. Two most common types of summary visualizations are *video abstracts*, which are based on one or more keyframes extracted from each segment, and *video skims*, where a shorter summary video is generated from selected parts of the original sequence. In our system, the video is temporally segmented into smaller blocks and a set of keyframes are selected to generate the video abstract (See Figure 2.8).

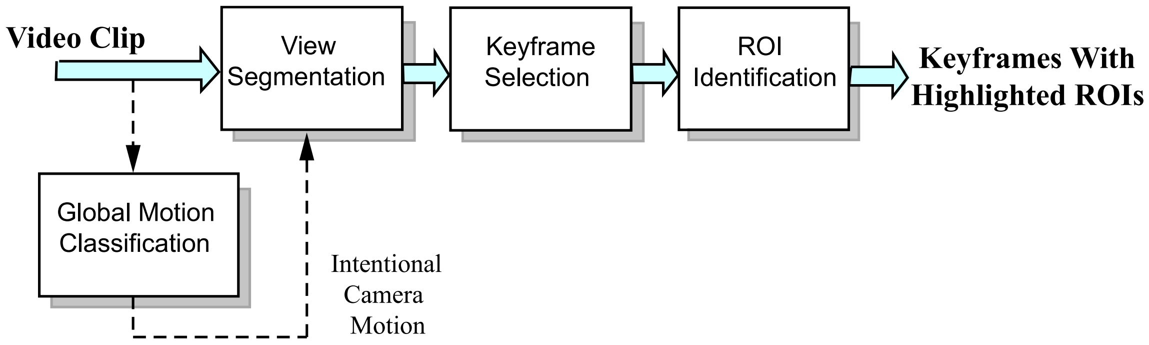

Video Clip
View Segmentation
Keyframe Selection
ROI Identification
Keyframes With Highlighted ROIs
Global Motion Classification
Intentional Camera Motion

Fig. 2.8. Block diagram of the video summarization method based on camera motion.

2.3.1 Temporal Video Segmentation

Dividing a video sequence into temporal segments that contain distinct scenes is usually the first step in creating video summaries. In the analysis of produced video this is generally achieved by detecting shot boundaries and segmenting the sequence into shots. However, UGV sequences do not have a shot-based syntax that can be used as a basis for temporal segmentation. In order to address the problem of temporal video segmentation of UGV sequences we propose a new segmentation approach based on “camera views.” First, we define some terminology we use below.

Two frames are considered to be *correlated* if they have overlap with each other i.e. at least a common part of the background is visible in both frames. A *camera view* is a temporal concept defined as a set of consecutive frames that are all correlated with each other. We use the term *view* occasionally to refer to camera view. The view transitions or view boundaries, which occur when the camera is displaced or there is a change of viewing angle, are detected to temporally segment the video. The boundaries are selected such that the frame right before the segmented camera view and the one right after that, are not correlated. At least one frame from each view is selected to be present in the video summary as described in the next section. With this approach, the summary provides the user with a notion of all the scenes captured by the camera. This characteristic of the video summary is particularly useful for applications such as video annotation [41]. The idea is similar to representation of a video with a panoramic mosaic image. However, we are not interested in combining the frames into one big mosaic since this way of representation is not suitable for small-displays and requires motion compensation and alignment of the frames.

View transitions can be abrupt or gradual, e.g., an abrupt view transition occurs when the camera quickly turns from one direction to another. In this case, the frames during a fast pan are usually blurred. These frames have been previously labeled as blurry in the motion classification step. As a result, these frames are not considered for inclusion in the video summary. Consequently, the sequence is divided into two distinct views. On the other hand, if the camera moves smoothly between two views, there is no clear-cut view boundary.

In order to detect the camera view boundaries, we define the displacement vector between frames i and j as $\mathbf{D_{i,j}} = [D^x_{i,j}, D^y_{i,j}, D^z_{i,j}]^T$, where D^x, D^y and D^z are total horizontal, vertical and radial interframe displacements given by

$$D^x_{i,j} = \sum_{\substack{k=i+1 \\ is\_zoom=0}}^{j} \frac{H_{k,k-1}}{w_f} \qquad D^y_{i,j} = \sum_{\substack{k=i+1 \\ is\_zoom=0}}^{j} \frac{V_{k,k-1}}{h_f} \qquad D^z_{i,j} = \sum_{\substack{k=i+1 \\ is\_zoom=1}}^{j} \frac{R_{k,k-1} - 1}{R_N}, \tag{2.9}$$

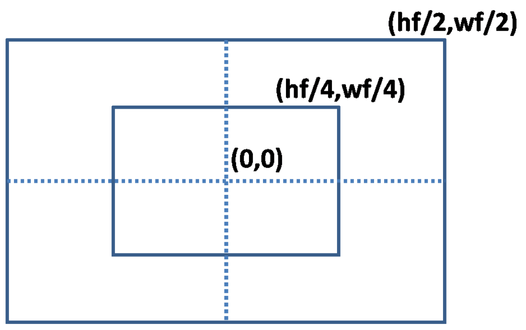


Fig. 2.9. Reference point used to obtain R_N.

where H, V and R are the motion parameters defined in Section 2.1. The displacement values in the x and y directions are normalized to the frame width, w_f, and height, h_f, respectively. These values are the minimum displacement needed to make the two frames uncorrelated with each other. The displacement caused by radial motion is not the same for all the pixels in the frame. The displacement at the point $\begin{pmatrix} x \\ y \end{pmatrix}$ is obtained as $\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = (R-1)\begin{pmatrix} x \\ y \end{pmatrix}$. The average displacement of the pixels in each quarter of the frame is $(R-1)\begin{pmatrix} \frac{w_f}{4} \\ \frac{h_f}{4} \end{pmatrix}$ (See Figure 2.9). Dividing the value in the horizontal direction by w_f and in the vertical direction by h_f will result in $\frac{R-1}{4}$ in each direction. Thus, we set R_N to 4 in Equation 2.9. The translational displacements are updated when there are no zooms ($is\_zoom = 0$) and the radial displacement is updated only during the zooms ($is\_zoom = 1$). This is due to the fact that estimates for H and V could be inaccurate during a zoom.

Segmentation of a video sequence into different camera views can be considered as the selection of a set of view boundary frames, $\{f_{k_1}, f_{k_2}, .., f_{k_M}\}$ for the segments $(f_{k_1}..f_{k_2})$, $(f_{k_2}..f_{k_3})$,. . ., $(f_{k_{M-1}}..f_{k_M})$. The view boundary frames are identified as follows: starting from the beginning of the sequence, a boundary frame is flagged whenever the magnitude of the displacement vector, $\|\mathbf{D}\|$, for the current frame and that for the previously detected boundary frame is larger than $T_d = 1$, which indicates that the frames f_{k_i} and $f_{k_{i+1}}$ have almost no overlap. This procedure is repeated until

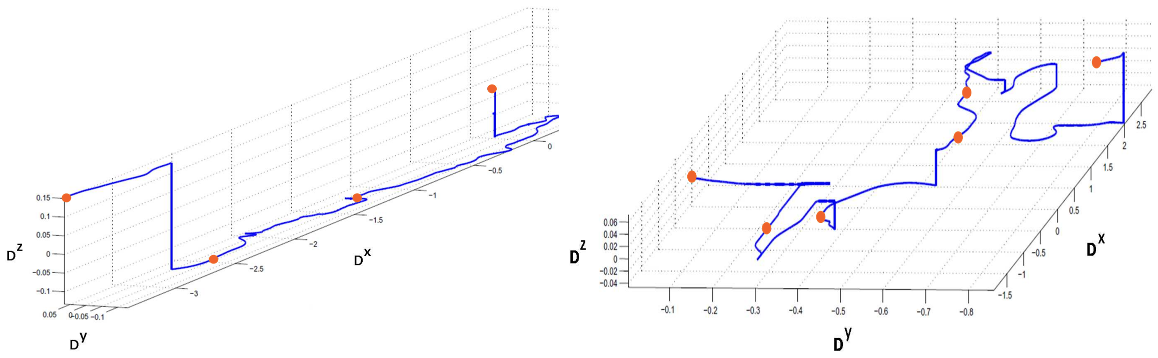


Fig. 2.10. The displacement curves and camera view boundaries indicated by red circles for two different UGVs.

the end of the video clip is reached. We place a constraint on the selection such that a boundary frame cannot be chosen during intervals labeled as blurry segments in the motion classification step. This is due to the fact that we do not want the video summary to include the blurry frames. Hence, the boundaries inside blurry regions are relocated to the end of the blurry segment. Figure 2.10 shows the 3D displacement curves for two UGVs. Each point on the curve is a frame number, i, mapped into the displacement space using $\mathbf{D_{0,i}}$. The red circles indicate the corresponding view boundaries detected by the algorithm.

2.3.2 Keyframe Selection

After dividing a UGV sequence into segments corresponding to different camera views as described in the previous section, a compact representation of the video based on these segments is generated as the video summary. In this section we describe our approach for selection of keyframes in UGVs in order to produce video abstracts. Moreover, these frames are used in other steps of the analysis such as the identification of ROIs in a scene. In order to represent the segment it is extracted from, a keyframe should be the frame with the highest subjective importance in the segment. However, defining relative "importance" in scenes with diverse content is a difficult problem. As mentioned in the previous section, the segmentation has the property that by using a simple strategy for keyframe selection, the video summary will cover almost all the views captured in the video. Thus, we would like to avoid complicated algorithms for this purpose but at the same time we would like our results to be consistent with what the camera person and the viewers find more interesting.

For this purpose, we conducted a preliminary experiment in which we asked a number of subjects to select a set of representative frames from several UGV clips with durations of 15–120 seconds. The clips were first segmented using our camera view segmentation approach. The subjects had access to individual video frames. We gave the following statement to the users, "If you wanted to summarize the important content of the video segment in minimum number of frames, which frame(s) would you choose?" The subjects also had the option not to select any frames if they thought the segment was of insignificant information. The subjects were also asked to provide the reason why they selected each of the keyframes. Based on the collected information we formed some initial conclusions. In general, the selected keyframes had the following properties: the frames were picked when there was a "nice" view of the objects of interest, a new action occurred, after a zoom-in (close-up view), after a zoom-out (overall view), a new semantic object entered the camera view (text, people, buildings), there was a pause after a continuous camera movement or a significant change in the background. The subjects did not select any frames from the segments that were highly blurred or shaky or when the camera motion does not change and the scene is static e.g., camera pans to the left on a homogeneous background.

Since our intention was to avoid the complex tasks of object and action recognition in our system, our keyframe selection strategy was only based on camera motion. This way the results are compatible with viewers' preferences as well as what the camera person deems important while shooting the video through intentional camera motion. The following frames are selected as keyframes:

- The frame after a zoom-in is usually a close-up view of object of interest and is included in the summary.

- The frame after a large zoom-out is chosen as a keyframe to give an overview of the scene.
- A camera move-and-hold is another indicative pattern and the frame where the camera is at pause is usually emphasized.
- For segments during which the camera has constant motion, all frames are considered to be of relatively same importance. In this case, the frame closest to the middle of the segment and having the least amount of motion is chosen as the keyframe in order to minimize blurriness.

3. KEYFRAME SALIECNY MAPS AND ROI DETECTION

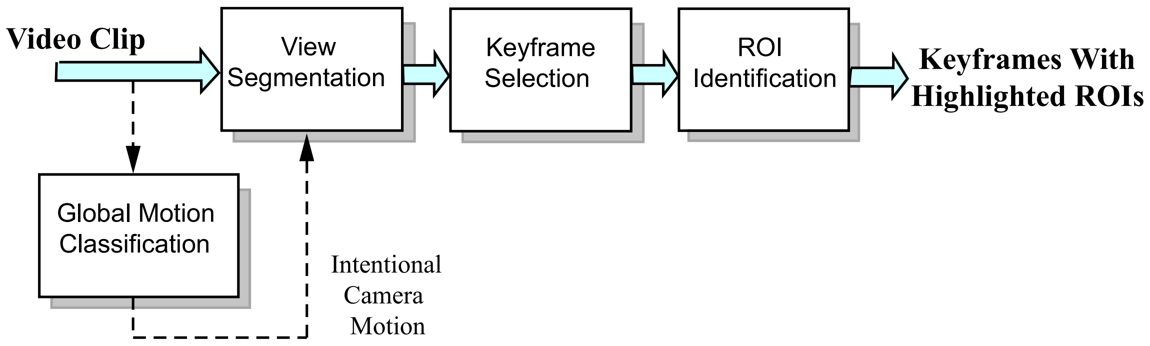

Video Clip
View Segmentation
Keyframe Selection
ROI Identification
Keyframes With Highlighted ROIs
Global Motion Classification
Intentional Camera Motion

Fig. 3.1. Block diagram of the video summarization method based on camera motion.

In order to extract higher-level information from the keyframes we employ a human attention model to obtain saliency information within keyframes. This information is used to identify regions of interest (ROIs), i.e. regions that capture the attention of a viewer while watching the video. This is the last block of the summarization component in our system as shown in Figure 3.1.

In order to identify the ROIs, we first need to generate the keyframe saliency maps. A *saliency map* or *importance map (S)* of a frame indicates how much a viewer is visually attracted to different regions in the frame. Studies based on eye movements have identified several factors that influence visual attention such as motion activity, contrast, size, shape, faces and spatial location [29, 34, 35]. In this thesis we propose a new location-based saliency map that employs camera motion information. We combine this map with a color contrast saliency map, a moving objects saliency map and highlighted faces to generate the keyframes saliency maps.

3.1 Color Contrast Saliency Map

One of the important factors that causes a region to stand out and be more noticeable is the contrast of that region compared to its neighborhood. This includes contrast in luminance and color. In [35], Ma and Zhang used the color components in LUV space as the stimulus in order to generate saliency maps for images. We use a similar technique in our system to [35], however, we use the RGB color space to generate the contrast-based saliency map. Our experiments showed better results in this space compared to using UV components since it combines the luminance and color contrasts.

First, the three-dimensional pixel vectors in RGB space are clustered into a small number of color vectors using generalized Lloyd algorithm (GLA) for vector quantization [52]. The expression for updating the cluster centroids, $\mathbf{c}_i$ and the distortion measure, D_i used in GLA is modified as follows

$$\mathbf{c}_i = \frac{\displaystyle\sum_n v(n)\mathbf{x}(n)}{\displaystyle\sum_n v(n)} \quad , \qquad \mathbf{x}(n) \in C_i \tag{3.1}$$

$$D_i = \sum_n v(n) \left\| \mathbf{x}(n) - \mathbf{c}_i \right\|^2 , \qquad \mathbf{x}(n) \in C_i \tag{3.2}$$

where $\mathbf{x}(n)$ represents the RGB vector of each pixel and $v(n)$ is the corresponding pixel weight which is obtained in such a way that pixels in noisy regions have smaller weights than the ones in the smooth regions [53]. At each iteration, the cluster with maximum distortion value is split into two new clusters until the predetermined number of clusters is obtained. A fixed number of clusters is used to avoid additional computational complexity for finding the optimum number of clusters since this number does not have a major effect on the quality of results. Based on the experimental results, we found 32 colors to be sufficient for our purpose. The color-quantized frame is then downsampled by a factor of 16 in each dimension in order to reduce the computational complexity. In our case, we chose the value of N_d to be 16.

Finally, the contrast-based saliency value for each pixel (i,j) in the downsampled image is obtained by

$$S_{\text{Contrast}}(i,j) = \sum_{q \in \Theta} d(p_{ij}, q) \tag{3.3}$$

where p_{ij} and q are the RGB pixel values, Θ is the neighborhood of pixel (i,j) and d is the Gaussian distance. In our experiments a 5×5 neighborhood was used. Figure 3.2 shows an example of the contrast-based saliency map for an extracted keyframe. The original frame in part (a) is color-quantized and downsampled by a factor of 16 resulting in parts (b) and (c) respectively. Part (d) illustrates the generated color contrast saliency map.

3.2 Moving Object Saliency Map

Regions of frames with significant motion with respect to the background can also attract attention. To determine the moving object saliency map, we examine the magnitude and phase of macro block relative motion vectors. Macro block (MB) motion vectors, $[\ M_x \quad M_y\ ]^\top_{mn}$, are generated using a full search algorithm for macro blocks of size 16×16 pixels. The global motion parameters H and V are subtracted from the motion vectors to compensate for global motion

$$\begin{bmatrix} \widetilde{M}_x \\ \widetilde{M}_y \end{bmatrix}_{mn} = \begin{bmatrix} M_x \\ M_y \end{bmatrix}_{mn} - \begin{bmatrix} H \\ V \end{bmatrix} \tag{3.4}$$

where $[\ \widetilde{M}_x \quad \widetilde{M}_y\ ]^\top_{mn}$ represents the relative motion vector for the macro block at location (m,n). Since motion vectors in flat regions are usually erroneous, we apply a threshold on the average norm of gradient for each MB and assign zero relative motion to the MBs with below the threshold values. The motion intensity I and motion

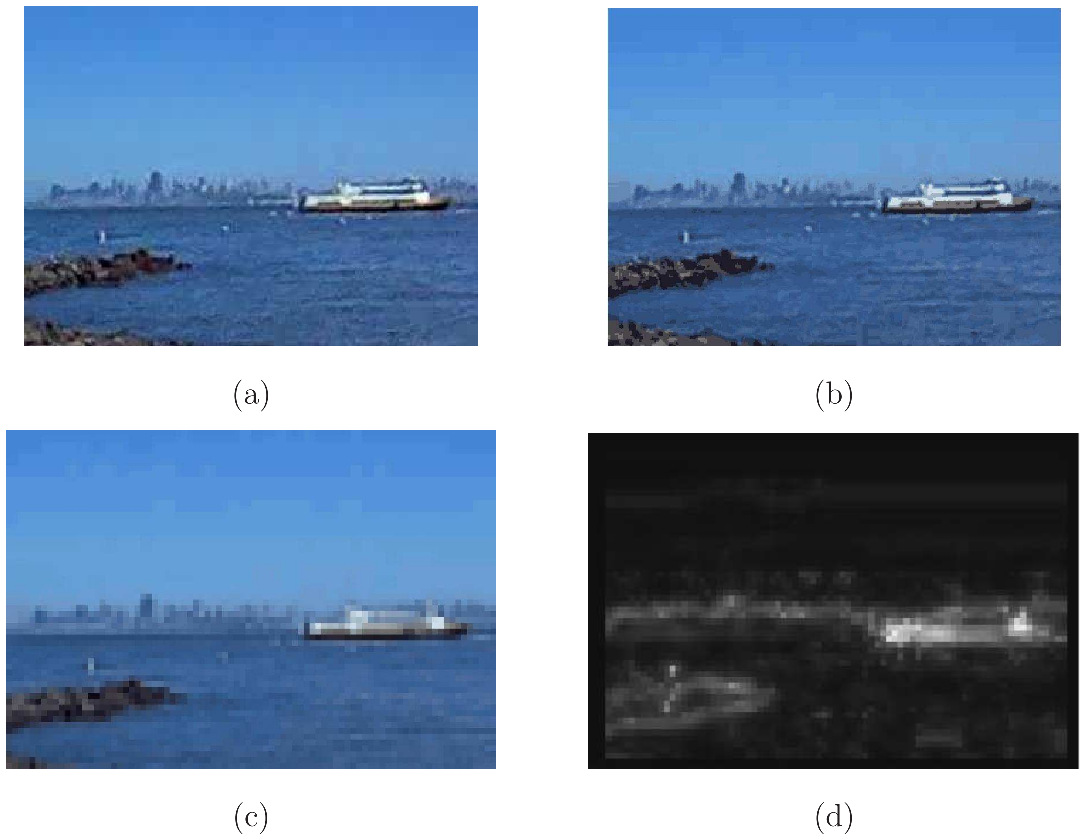


Fig. 3.2. Example of contrast-based saliency map for a keyframe. (a) Original keyframe, (b) Color-Quantized frame,(c) Enlarged downsampled frame and (d) Saliency map.

phase entropy Hp maps are then generated similar to [54]. The motion intensity I and motion phase ϕ for MB$_{mn}$ are defined as

$$I_{mn} = \sqrt{\widetilde{M}^2_{x\,mn} + \widetilde{M}^2_{y\,mn}} \tag{3.5}$$

$$\phi_{mn} = \arctan \frac{\widetilde{M}_{y\,mn}}{\widetilde{M}_{x\,mn}} \tag{3.6}$$

The phase entropy map, Hp, indicates the regions with inconsistent motion which usually belong to the boundary of the moving object. The value of Hp for each MB is determined as follows. First, an 8-bin phase histogram is obtained for a sliding window with size 5×5 MBs at the location of the MB. The phase entropy at MB_{mn} is then

$$Hp_{mn} = -\sum_{k=1}^{8} p_{mn}(k) \log p_{mn}(k), \tag{3.7}$$

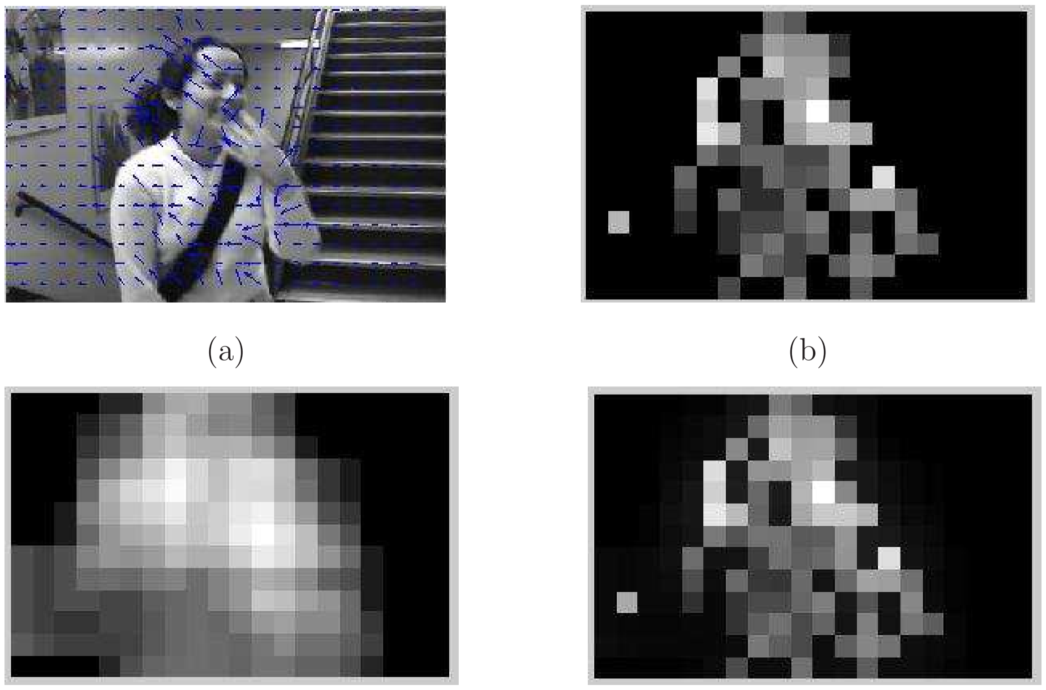
where $p_{mn}(k)$ is the probability of k^{th} phase whose value is estimated from the histogram. Combining the intensity and phase entropy maps results in a moving objects saliency map.

$$S_{\mathrm{MovingObjects}} = I + Hp \tag{3.8}$$

Figure 3.3 illustrates an example of a moving objects saliency map for a video frame. Part (a) illustrates the estimated motion vectors before global motion compensation. Part (b) and (c) are the generated maps based on magnitude and phase entropy of motion vectors respectively. Combining these two maps results is the moving objects saliency map in part (d).

3.3 Location-Based Saliency Map

A location-based saliency map indicates how the saliency values of pixels change with respect to their spatial location in the frame. Most approaches that have considered location-based saliency are based on experiments on still images which have resulted in central saliency, i.e. the center of the image is considered to be visually more important [34, 36, 37]. As we will show through our user study in Section 3.6, the direction of the camera motion also has a major effect on the regions where a viewer “looks” in the sequence. For UGV, in which there is not much object activity, usually the intentional motion of the camera determines the “story” by moving around the scene or zooming in/out. The human visual system has a tendency to follow this motion and particularly look for new objects that are about to enter the camera view. For example, if the camera is panning towards the right, the a viewer is more attracted to the right side of the scene or when the camera starts to zoom out, the attention to the borders of the frame increases. In the case of a zoom-in or still camera, the location saliency is similar to the one for still pictures and the



(a) (b) (c) (d)

Fig. 3.3. An example of moving object saliency map. (a) Motion vectors (not compensated), (b) Compensated motion intensity map, (c) Phase entropy map and (d) Final moving objects saliency map.

viewer's attention is more concentrated on the center of the frame. We conducted an eye tracking experiment to verify our hypothesis as we will describe in Section 3.6.

The global motion parameters were used to generate the location saliency maps for the extracted keyframes. Three individual maps for H, V and R directions are generated as in (3.9), (3.10) and (3.11) and combined to form the location saliency map (3.12).

$$S_H(i,j) = \max\left(0, 1 - \frac{|j - \frac{w_f}{2} - k_H H|}{\frac{w_f}{2}}\right) \tag{3.9}$$

$$S_V(i,j) = \max\left(0, 1 - \frac{|i - \frac{h_f}{2} - k_V V|}{\frac{h_f}{2}}\right) \tag{3.10}$$

$$S_R(i,j) = \begin{cases} 1 - \frac{r}{r_{max}} & R \geq 0,\ 0 \leq r \leq r_{max} \\ k_r \left(1 + \frac{r}{r_{max}}\right) & R < 0 \end{cases} \tag{3.11}$$

$$S_{\text{Location}} = S_H + S_V + S_R, \tag{3.12}$$

where (i, j) is the pixel location, and k_H, k_V and k_r are constants whose values were experimentally found to be optimum at 10, 5 and 0.5, respectively. The parameter r represents the distance of a pixel from the center of the frame and r_{max} is its maximum value in the frame. The parameters w_f and h_f are the frame width and height, respectively. After combining the H and V maps, the peak of the map function is at $(k_H H, k_V V)$. If there is no translational motion, the peak occurs at the center of the frame. The radial map, S_R, is either decreasing or increasing as we move from the center to the borders, depending on whether the camera has a zoom-in/no-zoom or zoom-out operation. Some examples of maps for various camera operations are shown in Figure 3.4. In part (a) the camera is zooming in so the values are larger at the frame center. Parts (b) and (d) show the maps for panning left and tilting down operations which moves the peak of attention toward the left and bottom part respectively. In part (c) camera is zooming out while panning toward the left so the attention is moved toward the borders with more emphasis on the left side of the frame.

3.4 The combined Saliency Map

In order to generate the combined saliency map from the above maps, first, the color contrast and moving object saliency maps are superimposed since they represent two independent factors in attracting visual attention. In addition to the low-level visual features mentioned above, specific objects such as faces, hands and texts can also draw the viewers' attention. These regions are semantically important and may have been overlooked in the low-level saliency maps. In our system, faces are detected and highlighted after combining the low level saliency maps, in our case color contrast and moving objects maps, by assigning the saliency value $S_{tot} = 255$ to the pixels

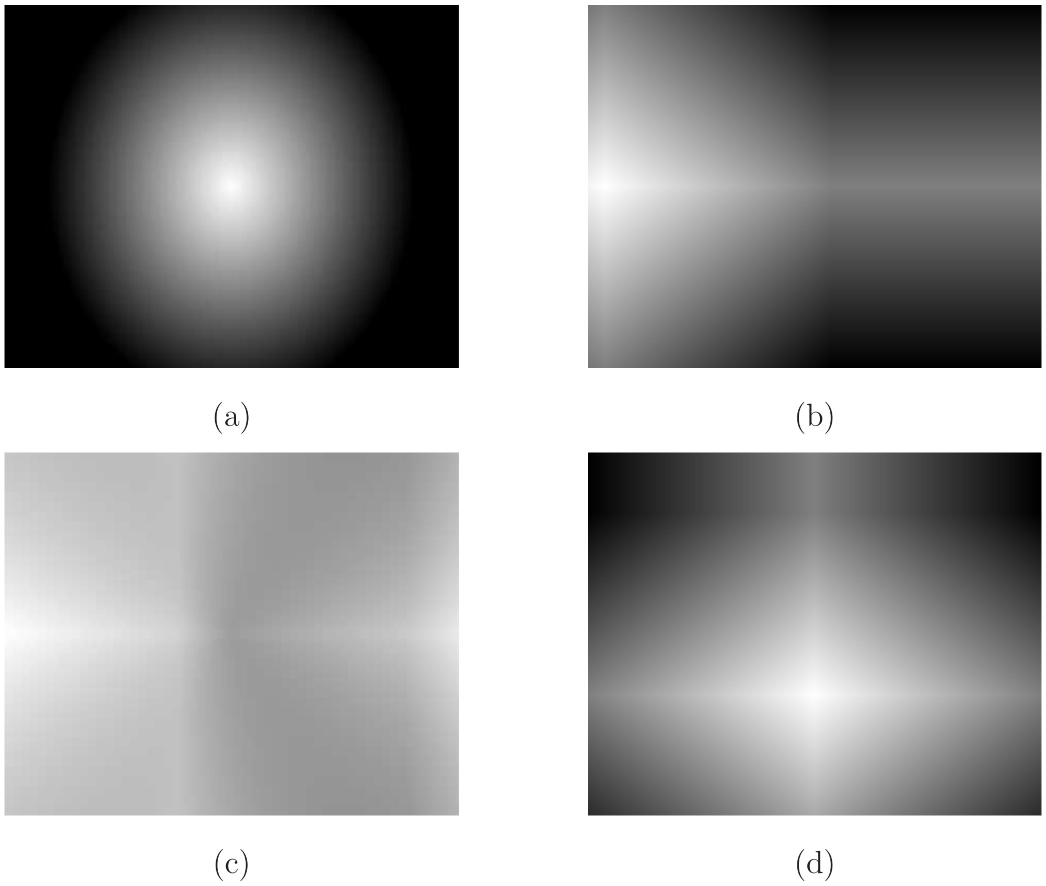


Fig. 3.4. Examples of motion-based location maps for: (a) Zoom-in, (b) Large panning toward left, (c) Zoom-out w/ panning left and (d) Tilting down.

inside the face regions as shown in Figure 3.5. We used an online face detection system [55] for this purpose.

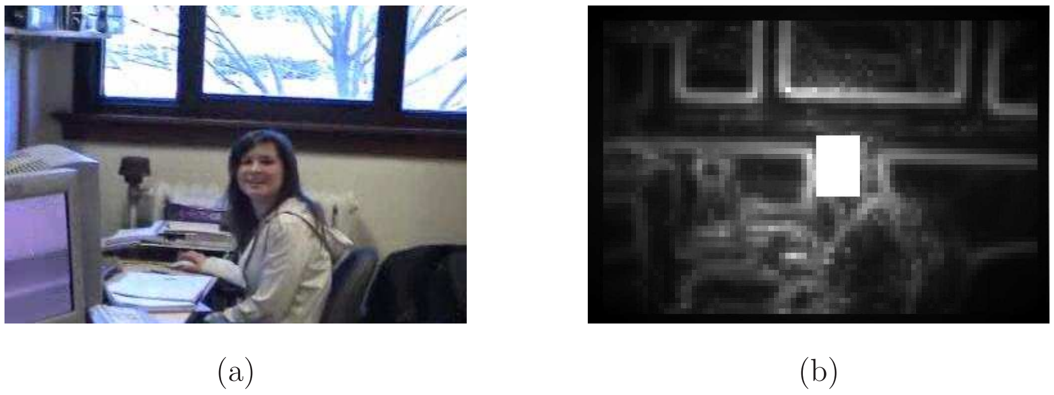


Fig. 3.5. Highlighting face areas in the saliency map.(a) Original frame and (b) The saliency map after face detection.

The location-based saliency map is then multiplied pixel-wise with this map to yield the combined saliency map, S_{tot}.

$$S_{tot} = \mathcal{F}d\left(S_{\text{Contrast}} + S_{\text{MovingObjects}}\right) \cdot S_{\text{Location}} \tag{3.13}$$

where $\mathcal{F}d$ is the process of highlighting faces on the map and S_{Location} is the location saliency map defined in (3.12). The values of S_{tot} are normalized to [0,255]. Examples of combined saliency maps for several keyframes are shown in Section 4.1.

3.5 Identification of ROIs

The saliency map is a gray level image that represents the relative "importance" of pixels in a frame. In order to extract regions of interest from the saliency map, region growing described in [35] is used in our system. In this method, fuzzy partitioning is employed to classify the pixels into ROIs, R_1, and insignificant regions i.e. areas that attract lesser attention from the viewer, R_0. The membership functions of these two fuzzy sets, $\mu_1(g_k)$ and $\mu_0(g_k)$, are shown in Figure 3.6 where g_k is the gray level value

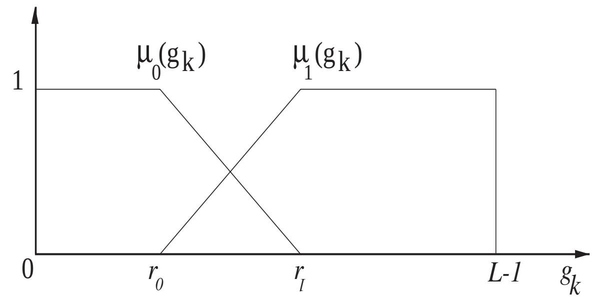


Fig. 3.6. Membership functions of the fuzzy sets: ROIs (R_1) and insignificant regions (R_0)

in the saliency map ranging from g_0 to g_{L-1}, and L is the number of gray level bins. The parameters r_0 and r_1 in Figure 3.6 are determined by minimizing the difference of entropy of the fuzzy sets as follows. The probability of each gray level, $p(g_k)$, is estimated from the histogram. The probability of each fuzzy set is defined as

$$\begin{aligned} P_0 &= \sum_{k=0}^{L-1} \mu_0(g_k)p(g_k) \\ P_1 &= \sum_{k=0}^{L-1} \mu_1(g_k)p(g_k) \end{aligned} \tag{3.14}$$

In order to determine the membership functions, we need to estimate the two parameters r_0 and r_1 in Figure 3.6. These parameters are determined by minimizing the difference of entropy of the fuzzy sets. This metric for image segmentation was originally proposed in [56]. The optimal values for r_0 and r_1 are obtained as

$$r_0^*, r_1^* = \arg\min_{r_0, r_1} |H_0(r_0, r_1) - H_1(r_0, r_1)|^2 \tag{3.15}$$

where H_0 and H_1 are the entropies of the two fuzzy sets and are determined as follows

$$\begin{aligned} H_0(r_0, r_1) &= \sum_{k=0}^{L-1} \frac{\mu_0(g_k)p(g_k)}{P_0} \ln \frac{\mu_0(g_k)p(g_k)}{P_0} \\ H_1(r_0, r_1) &= \sum_{k=0}^{L-1} \frac{\mu_1(g_k)p(g_k)}{P_1} \ln \frac{\mu_1(g_k)p(g_k)}{P_1}. \end{aligned} \tag{3.16}$$

The optimal values for r_0 and r_1 are found by exhaustive search. The seeds for region growing are defined to be the pixels with membership value of one and maximum local value in the saliency map. Adjacent pixels with saliency value $S_{tot}(i,j) > \frac{r_0+r_1}{2}$ and $S_{tot}(i,j) < S_{seed}$ become part of the seeds for the iterative growing. Examples of extracted regions for some saliency maps are shown in Figure 3.7. More examples will be presented when we describe our experimental results in Chapter 4.

The detected ROIs are highlighted in the keyframes presented to the user by plotting a bounding box around them. The center of the box is located at the mean of the ROI map and the box is extended by 1.5 times the standard deviation in each direction. A user has also the option to keep only the outlined ROIs as the video summary which is more suitable for small displays.

3.6 User Study on the Effect of Camera Motion on Human Visual Attention

In this section we investigate how camera motion influences visual attention through an eye tracking experiment. The results verified our hypothesis that was used for the location-based saliency map in Section 3.3. In this user study the eye movements of subjects were recorded using an infrared eye tracking system. The experiment was designed in a way so as to minimize the effect of visual features other than camera motion. For this purpose, a number of video sequences with various types of camera motion were generated.

Test Sequence Stimuli

Six videos, each with a different type of camera motion (pan right, pan left, tilt up, tilt down, zoom in and zoom out) were used. Each video consisted of several short video sequences with durations ranging from ten to forty seconds. All sequences in each video contained only one type of camera motion. Between the video sequences, a black frame with a bright point at the center was inserted with a duration of 2

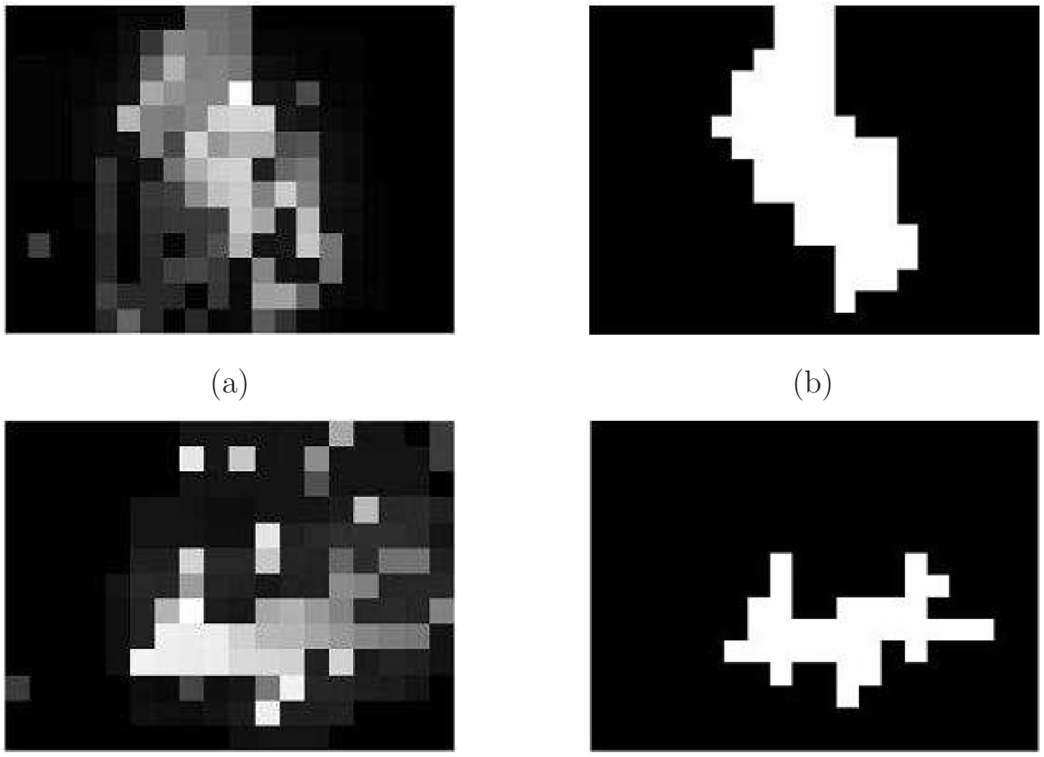


Fig. 3.7. Examples of region growing results: (b) and (d) are the extracted ROIs for saliency maps in (a) and (c) respectively.

seconds and the subjects were asked to fixate on the bright point when the black frame appeared. This was done to move the attention of the subject back to the center to avoid any bias when switching from one sequence to another. In order to minimize the effect of the visual features other than camera motion, we generated the video sequences as described below.

- For every scene with a specific direction of camera motion, we recorded the exact same scene with the opposite direction of camera motion. For example, if a sequence was constructed by panning the camera towards the right, another sequence was constructed from the same scene by panning the camera towards the left. As a result, the effect of any static visual feature is equal in both directions of camera movement.

- Moving objects, people and written texts were avoided in the videos since these factors significantly influence the human visual attention.

Each subject viewed only one of the six videos. The videos were displayed at a rate of 25 fps with display resolution of 478×403 pixels. The screen was located at the distance of 63 cm from the viewers' eyes.

3.6.1 Participants

Eighteen subjects took part in the controlled experiment. All had normal or corrected-to-normal vision. The subjects had not viewed any of the videos before and were not aware of our hypothesis. Each subject viewed the video corresponding to only one type of camera motion. Therefore, for each category of motion, data was collected from three subjects.

3.6.2 Apparatus

An infrared-based eye tracker (RK-726PCI Pupil/Corneal Reflection Tracking System, ISCAN) was used to record the eye movement. The eye tracker recorded at a sampling rate of 60 Hz and mean spacial accuracy of 1.° A chin rest with a forehead stabilizer (Table Mounted Head Restraint, Applied Science Group) was used to restrict the movement of the participant's head. Figure 3.8 illustrates the system.[1]

3.6.3 Calibration

Before each test, a nine-point static calibration procedure was performed during which a subject was asked to fixate at each point shown as a white cross on the

[1] I would like to thank Professor Hong Tan for allowing me to use her eye tracking equipment for this experiment.

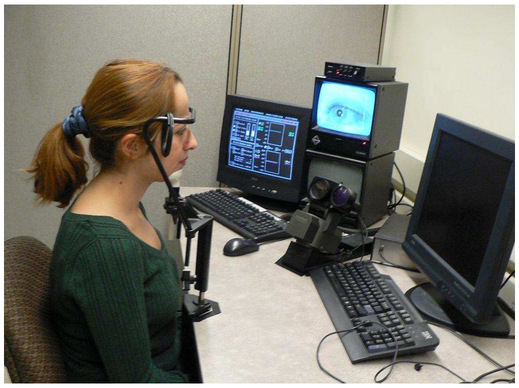


Fig. 3.8. The eye tracking experimental setup.

black screen. These points included the eight points on the border and the one at the center. Each subject was individually calibrated due to variation in eye geometry and physiology. Once the calibration was done, the eye tracker was able to record the eye movement.

In order to confirm the quality of the calibration and also compensate for the possible non-linearity in the eye tracker data, a nine-point dynamic calibration was performed prior to the experiment. This calibration was done by asking the subjects to follow a moving white cross on the black screen. The cross displayed each of the nine points for four seconds and moved to the next point.

The dynamic calibration data was used to find the relationship between the eye tracking points and their corresponding locations in the frame. This data indicated non-linearity in our eye tracking device that caused the calibration points located on the rectangular grid in the stimuli to be warped into points on trapezoids in the data plane as illustrated in Figure 3.9. In order to compensate for this nonlinearity, we used an 8-parameter model to transform each data point into its corresponding location in the stimuli space. The plane was divided to four segments (Figure 3.9) and a different set of transform parameters was obtained for each segment using the nine calibration points. The 8-parameter model is defined as

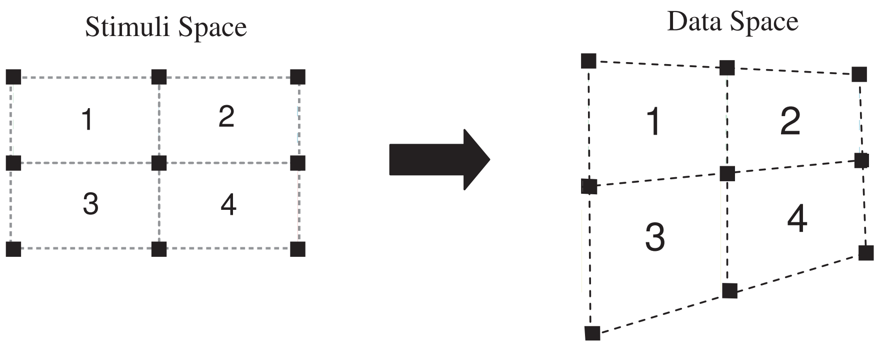


Fig. 3.9. Projection of the calibration points from stimuli space to the data space.

$$
\begin{aligned}
x' &= \frac{a_{i1} + a_{i3}x + a_{i4}y}{1 + a_{i7}x + a_{i8}y} \\
y' &= \frac{a_{i2} + a_{i5}x + a_{i6}y}{1 + a_{i7}x + a_{i8}y}, \qquad i = 1, 2, 3, 4
\end{aligned}
\tag{3.17}
$$

where (x, y) are the data point coordinates, (x', y') are the corresponding stimuli point coordinates and the index i indicates which of the the four trapezoids in the data space the point belongs to (Figure 3.9).

Data Analysis

The eye-gaze data provided a direct measure of the overt visual spatial attention of each subject in our test. The fixation points were extracted by discarding the blinks and saccades from the data. In order to detect the saccades, we used a velocity-based algorithm that discriminated between the fixations and saccades based on their angular velocities [57]. The point-to-point velocity of each sample was obtained by dividing the distance between the current and previous point by the sampling time. This velocity was converted to angular velocity using the distance between eyes and the visual stimuli. A sample point was labeled as a saccade and discarded from the data if its angular velocity was greater than a threshold. We used the threshold value of 20 degrees/sec.

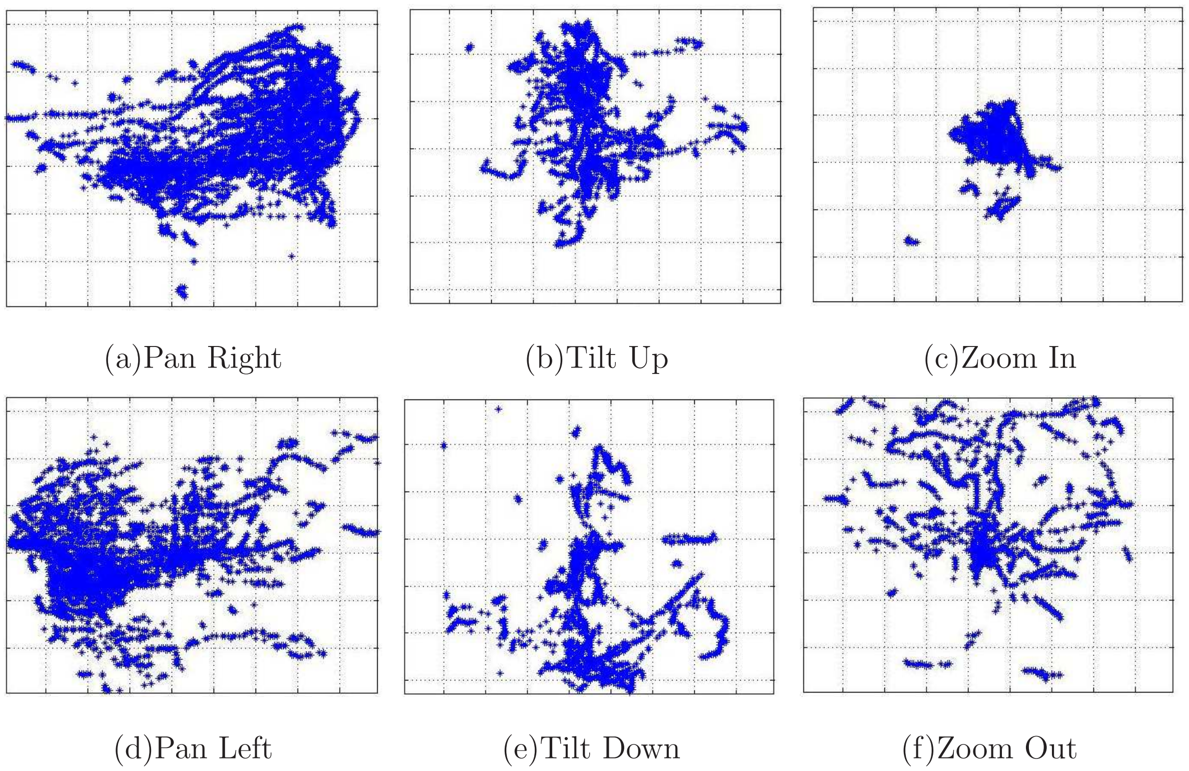


Fig. 3.10. Examples of the recorded fixation points for six different types of camera motion.

The distinct fixation points were then determined by clustering the sample points between each two consecutive saccades. Figure 3.10 shows the fixation points recorded from six different subjects for videos with different types of camera motion. As can be seen from the figure, in the case of panning and tilting, the center of the distribution of the fixation points is highly dependent on the direction of the camera motion. For example, in the video with camera panning toward the right (Figure 3.10−a), the majority of fixation points are at the right side of the frame. In the case of zoom in and zoom out, the camera motion affects the variance of the distribution. The zoom-out data points are more scattered over the frame area, whereas the zoom-in data points are more concentrated around the mean.

Table 3.1
Statistics of the fixation points for pan right and pan left camera motion.

Subject	Camera Motion	Number of Fixations	$\overline{x}$	σ_x	z
1	Pan Right	855	24.54	35.64	19.58
2	Pan Right	407	36.53	18.86	39.08
3	Pan Right	1682	46.5	26.2	72.79
4	Pan Left	2029	-24.2	35.9	-30.36
5	Pan Left	933	-29.46	26.62	-33.80
6	Pan Left	1191	-20.96	25.93	-27.90

1) Motion Type: Pan/Tilt

To show that the difference between the sample mean and the frame center, with $(0, 0)$ coordinates, in the pan and tilt videos is statistically significant, the Z test [58] is performed on the collected data. For pan left and pan right camera motions, the Z test is performed on the x coordinate of the fixation points. Similarly, the Z test is carried out in the y direction for tilt up and tilt down camera motions. Table 3.1 and 3.2 demonstrate the summary statistics and the corresponding z value for each set of fixation data in pan and tilt videos respectively. Here, the x axis points toward the right and the y axis points downward.

As indicated in Table 3.1, the samples means are located at the right side of the frame (positive values) for videos with pan right camera motion and at the left side (negative values) for videos with pan left camera motion. In a similar manner, depending on the direction of camera tilt, the sample means are shifted to the lower or upper side of the frame (Table 3.2). According to the Z tests, in all cases, the amount of displacement of sample means from the frame center is statistically significant with a 99% confidence level.

Table 3.2
Statistics of the fixation points for tilt up and tilt down camera motion.

Subject	Camera Motion	Number of Fixations	$\overline{y}$	σ_y	z
7	Tilt Up	256	-4.33	19.83	-3.49
8	Tilt Up	271	-15.87	21.28	-12.28
9	Tilt Up	578	-5.0	20.8	-5.78
10	Tilt Down	236	19.79	26.52	11.46
11	Tilt Down	250	19.84	17.13	18.31
12	Tilt Down	413	6.2	21.0	6.00

2) Motion Type: Zoom

The statistics of the fixation data for videos with zooming are summarized in Table 3.3. As illustrated in the table, the variances in both x and y directions are significantly larger in the case of zoom-out compared to the zoom-in case. This result implies that the direction of camera zoom (in or out) affects the covariances of the sample points. To verify the significance of the difference between the covariance matrices in the two cases, we put all the zoom-in data in one set and all the zoom-out data in another. A multivariate F test is then performed to verify the significance of the difference between the covariance matrices. We used the method proposed by Box [59] to determine the value of F. The two degrees of freedom for the F distribution are obtained using the number of samples in each class, number of classes and dimension of the sample points. In our case, the two degrees of freedom are $\nu_1 = 3$ and $\nu_2 = 5.96 \times 10^6$ where the latter value can be considered to be ∞. The critical value of F at the 99% confidence level is $F(0.01, 3, \infty) = 3.782$. The value of F for the eye tracking data is obtained as 128.02 (Table 3.4) which exceeds the critical value by a large margin. This concludes that the difference between the covariance matrices is statistically significant with a 99% confidence level.

Table 3.3
Statistics of the fixation points for zoom-in and zoom-out camera motions.

Subject	Camera Motion	Number of Fixations	$\overline{x}$	σ_x	$\overline{y}$	σ_y
13	Zoom In	151	4.52	13.12	0.39	8.29
14	Zoom In	91	-1.57	12.41	3.89	11.55
15	Zoom In	49	6.58	8.09	3.61	8.35
16	Zoom Out	283	5.95	26.97	11.08	23.56
17	Zoom Out	214	-1.83	24.92	14.51	19.24
18	Zoom Out	223	0.42	19.35	22.69	23.63

Table 3.4
F statistics for comparing the covariance matrices in zoom-in and zoom-out data.

Subjects	Camera Motion	Number of Fixations	$\begin{pmatrix} \overline{x} \\ \overline{y} \end{pmatrix}$	$\begin{pmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{pmatrix}$	F
13,14,15	Zoom In	291	$\begin{pmatrix} 3.13 \\ 2.48 \end{pmatrix}$	$\begin{pmatrix} 141.17 & -14.19 \\ -14.19 & 95.13 \end{pmatrix}$	128.02
16,17,18	Zoom Out	720	$\begin{pmatrix} 1.38 \\ 16.5 \end{pmatrix}$	$\begin{pmatrix} 571.82 & -15.98 \\ -15.98 & 520.7 \end{pmatrix}$	

4. EXPERIMENTAL RESULTS AND SUBJECTIVE EVALUATION

The output of the video summarization method is a set of keyframes with highlighted ROIs. In order to examine the performance of our video summarization approach and its various components, we tested it using several UGVs and conducted two user studies. Section 4.1 shows the outcome with respect to saliency maps and ROIs. In Section 4.3, we describe the subjective evaluation of our end-to-end system that compares the output of our system with user-selected outputs.

4.1 Keyframes With Highlighted ROIs

Here, we present examples of the saliency maps and ROIs generated by our system for a number of keyframes extracted by our system from UGVs. In Figure 4.1, the keyframe is selected after a zoom-in when the camera is at hold. As shown in part(e), the location-based saliency map has central saliency, i.e. the saliency value is maximum at the frame center. Parts(b)-(d) show different steps of generating color contrast saliency maps. This map is multiplied by the location-based map which results in the combined saliency map in part(f). The combined saliency map is then used to determine the fuzzy map and extract the ROI in the frame as shown in part(h).

Figure 4.2 shows another example in which the object has local motion. In this video the person shown in the frames is playing a musical instrument. The corresponding motion vectors are displayed in part(b). The camera does not have any motion therefore the location map for this frame has central saliency. Color contrast and moving object saliency maps are generated and combined as explained before. Also, the object's face is highlighted in the map. The location saliency map is then multiplied to the combined saliency map and results in the final saliency map in part(f). The fuzzy growing algorithm is then used and ROIs are extracted from the fuzzy map.

The two keyframes in Figure 4.3 are obtained from two different UGVs, both at a pan-right camera motion. Thus, the location maps have larger values at the right side of the frames putting more emphasis on this region when multiplied with the color contrast maps. The ROIs are shown in part(d). In both cases the extracted regions are located at the right side of the frames.

Since we do not use any object segmentation, the boundary of the extracted ROIs do not necessarily overlap with the object boundaries. This may result in partial association of some objects in the extracted regions. Figure 4.2 is an example of this case. As mentioned in Section 3.5, ROIs are highlighted by plotting a bounding box in the extracted keyframes when presented to the user as the video summary. The user has the option to crop the keyframes to view only the ROIs when watching the video summary. This way of visualization is especially useful for hand held devices with small-size screens.

Figure 4.4 shows several examples of the output of our system for different video sequences. For comparison, we tested the same set of frames using the saliency model proposed by Itti *et al.* [29] which is one of the most widely used visual attention models. This attention model is based on visual features such as color, intensity and orientation. Figure 4.5 shows the top five detected salient locations using this model. We used the iLab C++ Neuromorphic Vision Toolkit [60], an implementation of this model for generating saliency maps which was available online. The individual frames were used as inputs to the system. The white circle in each frame is the most salient location or the first attended point and the yellow circles are the next successive attended areas in the order that the red arrows indicate.

Comparing Figures 4.4 and 4.5, we see that in most cases one or more of the salient locations in Figure 4.5 falls into the ROI detected by our system. One noticeable difference occurs when there are people in a frame. In most UGVs people are the subject of the video however in [29], the authors do not consider faces as a higher priority feature. For example, in the scenes shown in frames (i) and (j) in Figure 4.5, the child is the main subject and the camera has a close up shot of him in many of the frames. Though, in frame (i), the most salient point is identified on the shoes of the person next to the child and in frame (j), none of the top 5 regions include the child.

The main distinction between the two systems is evident in the frames during intentional camera motion. As we show in our eye tracking study in Section 3.6, this factor draws the overall attention of the viewer to part of the frame depending on the direction of the motion. For example, the camera is panning towards the left in frame (b) in Figure 4.4. As a result, the visual attention is more focused along the horizontal line at the center of the frame and toward the left. In the same frame in Figure 4.5, we can see that both the deck of the ship and the horizon view of the city are among the salient points however the camera motion gives higher priority to the horizon region. Similar situation occurs in frame (g). Also, in frames (f) and (l) the camera has a zoom-out and therefore a larger region is selected as ROI compared to the zoom-in case in frame (k).

4.2 Comparison To A Traditional Shot Boundary Detection Method

In this section, we compare the results of our temporal video segmentation, based on camera motion, to a standard method of shot boundary detection. As mentioned before, shot detection has been used as a common way of temporally segmenting a video to smaller blocks. One of the standard methods of detecting shot boundaries is by comparing the color histogram of consecutive frames. For the comparison purpose, we obtained the color histogram difference function between the consecutive frames. This metric was used by Yep and Liu [4] to detect the abrupt and gradual shot transitions for different video sequences. The RGB color histogram difference between frames f_1 and f_2 is defined as

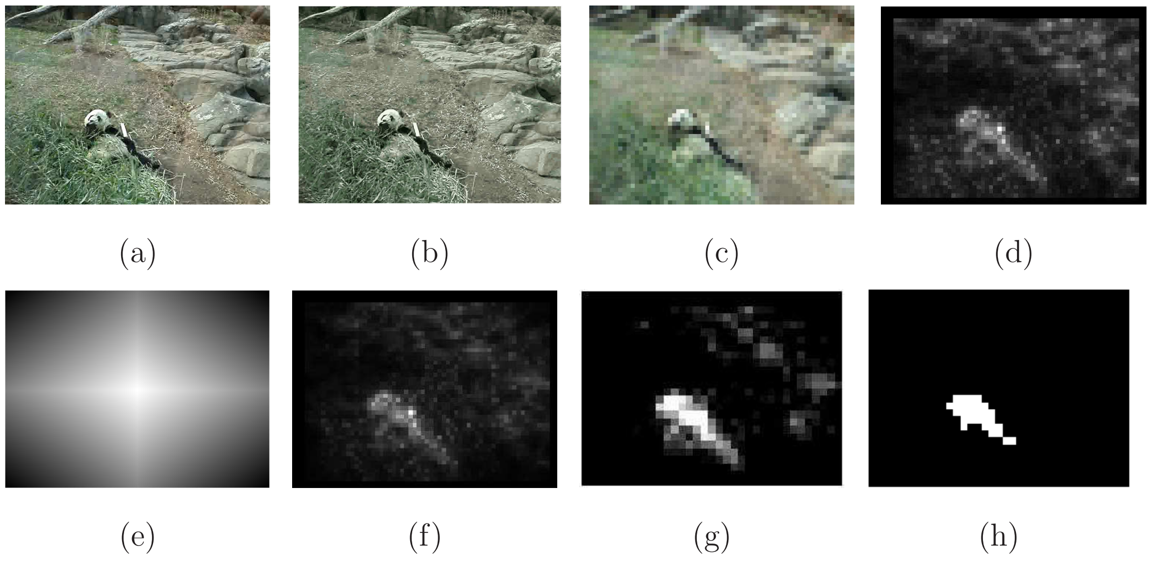


Fig. 4.1. (a) Original keyframe, (b) Color-Quantized frame,(c) Down-sampled frame, (d) Color contrast saliency map,(e) Location-based map, (f) Combined saliency map,(g) Fuzzy map and (h) The ROI obtained from our system.

$$d_{RGB}(f_1, f_2) = \sum_i (|h_1^r(i) - h_2^r(i)| + |h_1^g(i) - h_2^g(i)| + \left|h_1^b(i) - h_2^b(i)\right|) \tag{4.1}$$

where, h_1^r, h_1^g and h_1^b denote the histogram of red, green and blue components of frame 1 respectively. Figure 4.6 illustrates the histogram difference functions, d_{RGB}, for three video sequences. The keyframes extracted by our algorithm are shown at the right side of the figure for each video sequence. The d_{RGB} plots are shown with the same scale for all three videos. Video sequence (a) has the largest camera displacement and the camera angle has a 360^o rotation. However, since different views have similar color distributions, the magnitude of d_{RGB} is relatively small between the frames. In this sequence green is the dominant color throughout the video. Video sequence (b) has more color contrast. The two spikes in d_{RGB} correspond to the two instances where the lens cap swings in front of camera and causes an effect similar to a abrupt shot transition. Video sequence (c) has the highest color contrast among

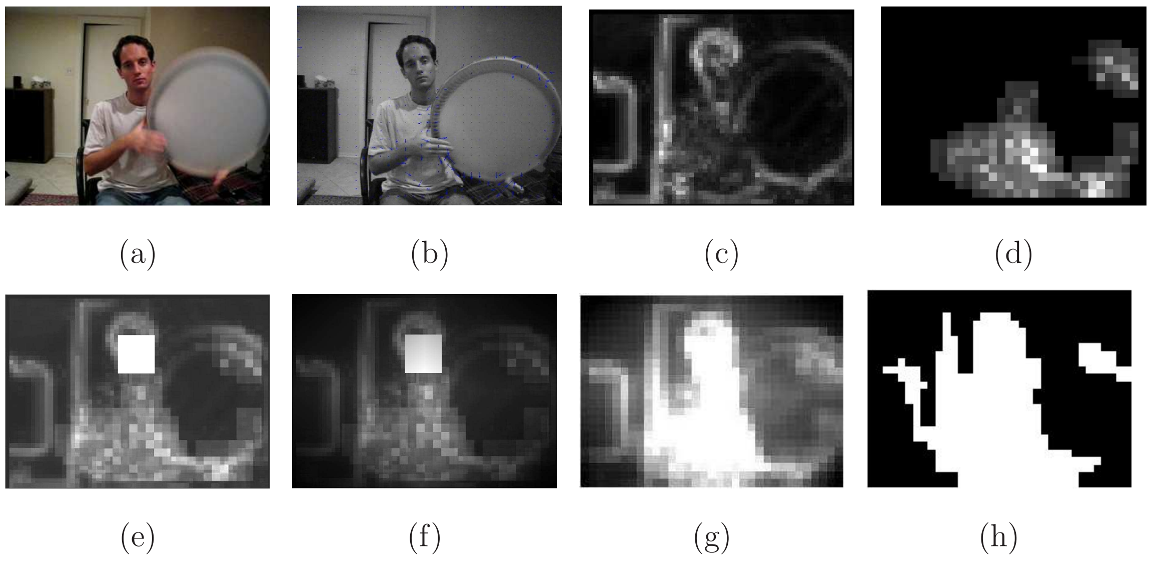


Fig. 4.2. (a) Original keyframe, (b) Motion vectors,(c) Color contrast saliency map, (d) Moving object saliency map,(e) Combined color,moving object and face maps, (f) Applied location saliency map on the combined map,(g) Fuzzy map and (h) The extracted ROI.

the three however, the camera in this video does not have a significant motion and our system detected only 3 camera views.

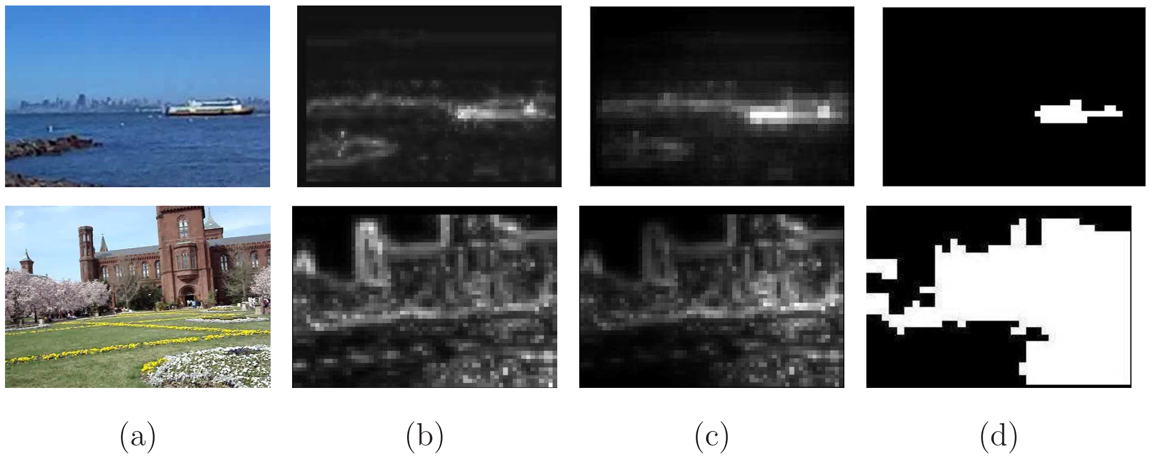


Fig. 4.3. (a) Original keyframe, (b) Saliency map before using the location map,(c) Saliency map after using the location map and (d) The ROI obtained.

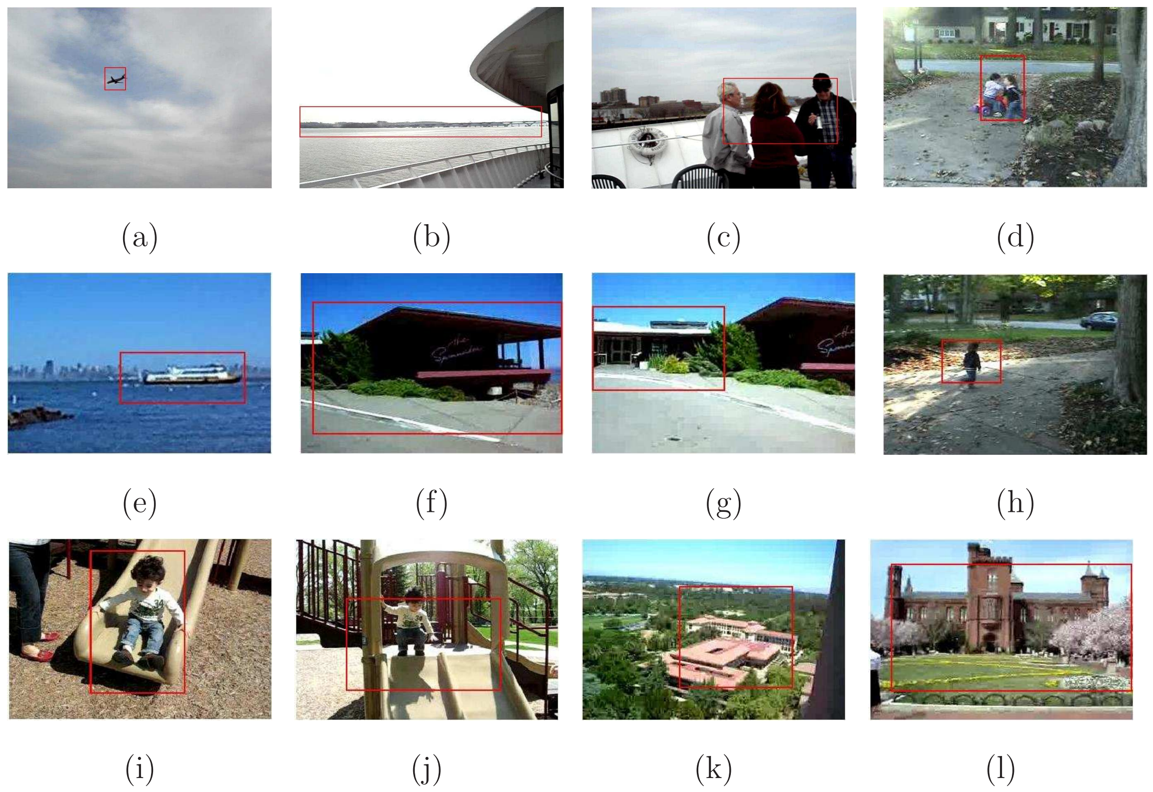
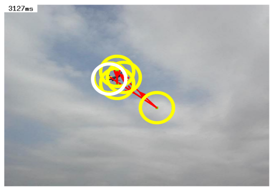
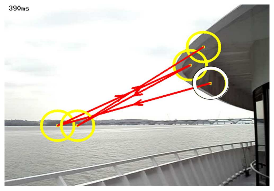


Fig. 4.4. Examples of highlighted ROIs in keyframes obtained by our system from different video sequences.

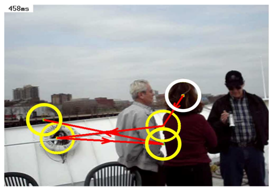
Therefore, color characteristics of the frames can change independently of how the camera moves depending on the scene and its color distribution. In some video sequences the color distribution does not have significant variations throughout the video. There may be a few colors dominating the color distribution in all the views. The segmentation or summarization methods that are based on measuring the difference between the color distributions among the frames fail to capture all the scenes in this kind of video sequences. Our segmentation approach measures the displacement of the camera and is capable of detecting the changes in the camera view. This method of summarization is particularly useful for video annotation where we can associate location-based tags to the video based on the backgrounds and views without missing any scenes.



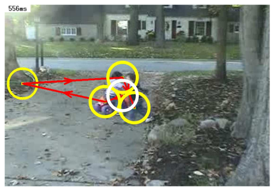
(a)



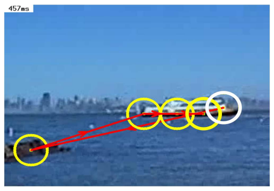
(b)



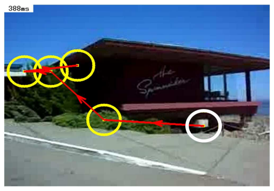
(c)



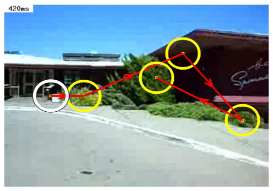
(d)



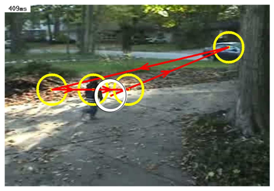
(e)



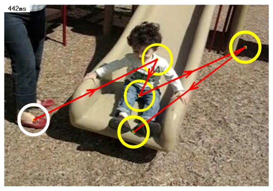
(f)



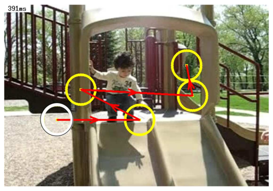
(g)



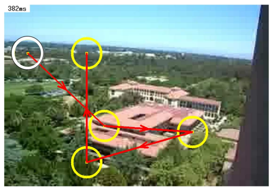
(h)



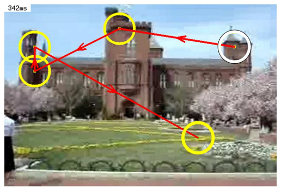
(i)



(j)



(k)



(l)

Fig. 4.5. Top five salient locations detected using Itti *et al.*'s model. The white circles show the most salient points in the frames and the yellow circles are the next four. The red arrows indicate the order.

4.3 Subjective Evaluation of Video Summarization Results

As previously mentioned, the outcome of our system is a collection of keyframes with highlighted ROIs (Figure 4.4). We conducted a user study of our overall system to compare our results to viewers' preferences. In our system, a video is first divided into camera views and the keyframes are extracted from each camera view to represent the scene. Our system chooses the frames that are emphasized by the camera person through specific patterns in the camera motion. The view segmentation is done in such a way that by selecting one frame from each camera view, the set of keyframes will cover almost all the scenes captured in the video. Consequently, if the ROIs are correctly detected, their set should contain all the important objects of the video. In

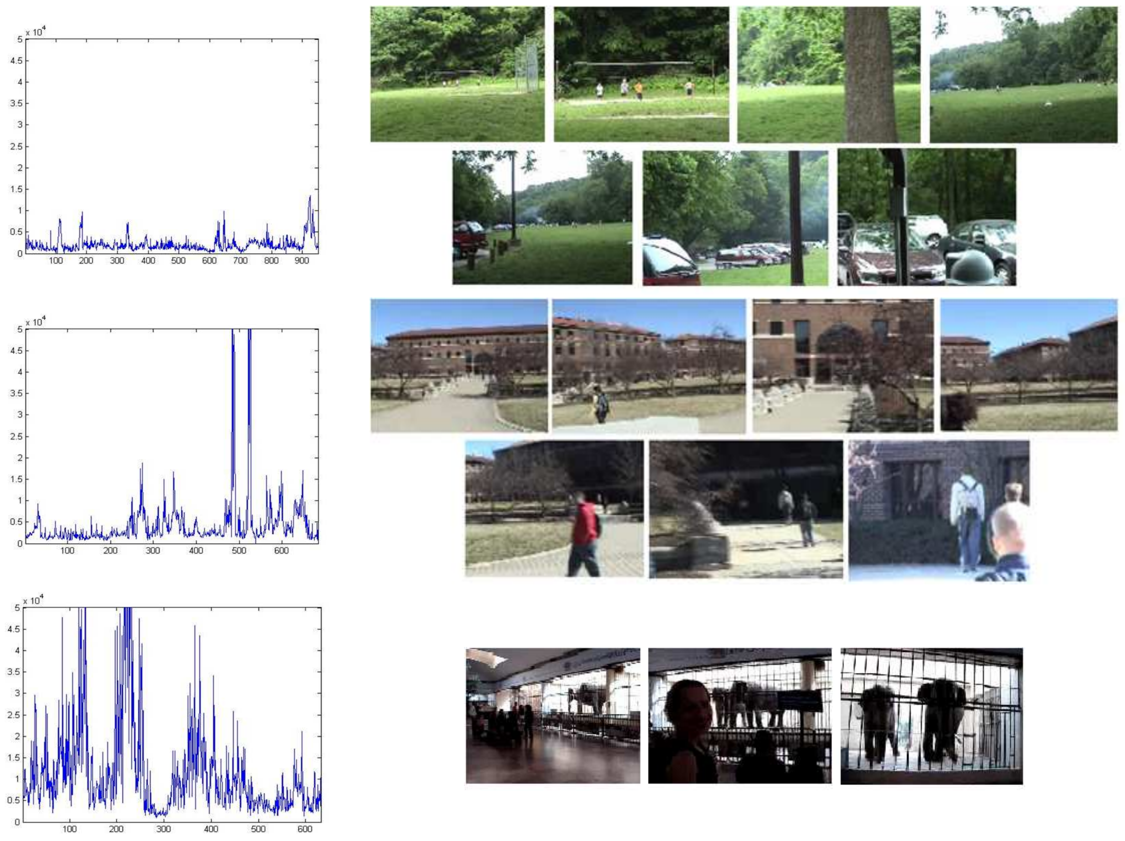


Fig. 4.6. Color histogram difference function between consecutive frames (Left) for three different video sequences. The keyframes extracted by our system are shown on the right side.

order to evaluate the end-to-end system, a subjective assessment was conducted as described below.

4.3.1 Method

Ten subjects participated in the user study. Each subject watched ten UGV sequences and was asked to select a fixed number of "keyframes" and arbitrary number of "objects of interest" from each clip with the following descriptions.

- Keyframe: If you want to summarize the video in the specified number of frames, which frames would you choose to represent the video?

- Object of interest: What is (are) the most important object(s) in the video?

For each video, the subjects were required to select exactly the same number of keyframe as determined by our system. This number depended on the number of camera views and the specific camera patterns detected by our system. Ten videos in the range of 30 seconds to 2 minutes were used for the experiment. The videos contained different types of intentional motion such as pan, tilt, zoom and hold and also abnormal motion such as blurry and shaky motions. The videos were from a wide range of scenes such as the zoo, university campuses, playgrounds, beaches, a cruise and parks. The subjects had access to each individual frame and were able to play the video frame by frame or with the normal frame rate. They were asked to watch the videos at least once before making the selection.

4.3.2 Data Analysis - Keyframe Selection

In order to compare the results of our system to the results from the user study, we consider the outputs of our system as the reference set and measure the closeness of the user-selected data to this set by determining the precision, recall and the F measure defined in Equations 4.2-4.4. The F measure is the harmonic mean of precision and recall [61]. For each frame selected by a user, if it is "close enough" to one of our system outputs, we consider it as a *correct hit* otherwise as a *false alarm*. For each output of our system, if a subject does not select a correct frame, it will be considered a *missed point* for that subject.

$$precision = \frac{\#correct\ hits}{\#system\ outputs \times \#subjects} \tag{4.2}$$

$$recall = 1 - \frac{\#missed\ points}{\#system\ outputs \times \#subjects} \tag{4.3}$$

$$F = \frac{2 \times precision \times recall}{precision + recall} \tag{4.4}$$

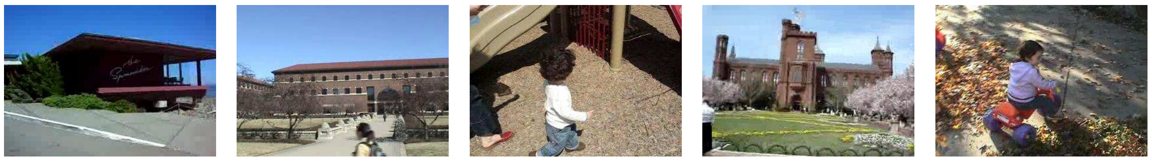
Several frames of a video can represent the same content, e.g. when the camera has small movement on a static scene or when the objects in the scene slightly move.

Figure 4.7 illustrates examples of this scenario. In this figure, either the camera, the object in the scene or both have small motion from the top frame to the bottom frame but the frames are semantically similar.

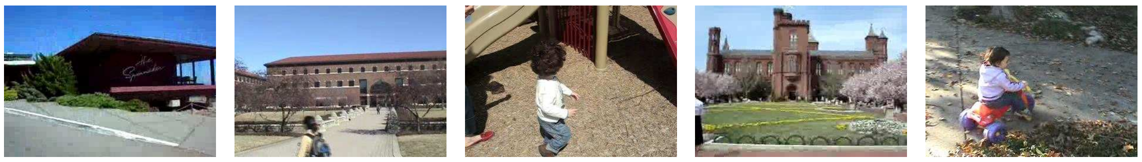
Therefore, the first step in comparing two frames is motion compensation between the two. The 3-parameter motion compensation described earlier is used for this purpose. We limit the search range to 20 pixels for a frame size of 360x240 pixels. Larger displacements are considered as significant motion and are not compensated. Since the motion estimation is not very accurate and also because of the artifacts caused by video compression, two similar frames still may not match, especially around the edges. Thus, we use a Gaussian filter to blur the edges.

Next, we use the SSIM metric [62] to measure the distance between the blurred images. A threshold is used on the SSIM distance to decide whether or not two frames are similar. This threshold is obtained separately for each keyframe of each video. The reason is that during some intervals, there maybe a small change in the visual content of the frame although it changes semantically while in others, the visual content changes rapidly. Thus, we determine the threshold based on local variation among the frames in the neighborhood of each keyframe. The threshold for each keyframe is the average distance between that keyframe and its two neighboring keyframes.

For each frame selected by a user, we compare it to the 4 closest indices among the keyframes determined by our system and find the most similar keyframe based on the SSIM distance. If this distance is less than the threshold associated with that keyframe, it is considered as a correct hit for that keyframe and otherwise a false alarm. Table 4.1 shows the statistics for the 10 videos. Average precision, recall and F measure and their variances and standard errors are listed in Table 4.2. It can be observed that the average precision, 87.32%, is higher than average recall of 70.38%. This is due to the fact that several user-selected frames may be matched to one system-selected keyframe so some of the outputs are missed by the user. The results indicate high correlation between the outcome of our system and the user-selected



(a) frame 495 (b) frame 54 (c) frame 275 (d) frame 356 (e) frame 57



(a) frame 503 (b) frame 70 (c) frame 278 (d) frame 365 (e) frame 88

Fig. 4.7. Examples of frame pairs in a video where the two frames are semantically similar even though the camera and the objects in the scene have moved.

frames. This not only shows that camera motion patterns affect how users select the keyframes but also validates our view segmentation step. Our view segmentation divides the video into groups of correlated frames therefore the set of selected frames from all the subshots cover almost all scenes in the video.

Figure 4.8 illustrates a number of typical failure cases of our system based on the user study. In video sequence (a) the child shown by the red arrows, makes a short appearance in the video and runs to the garage. The camera follows the child and holds when she enters the garage. Frame 371 is selected by our system as the keyframe and the entire area inside the garage is considered to be the ROI. However, most users selected the keyframes during the segment where the child is running, e.g. frames 251, 268 and 303. They identified the child as the ROI. This issue is due to the fact that the process of selecting keyframes is completely based on camera motion and does not consider object motion. Sequence (b) shows a similar case where different users chose different frames as representative frames based on the position of the children in the scene. Video sequence (c) shows an example where the results can be controversial. The camera holds on the view of the castle (frame 62) and pans to the left and moves back to the same view of the castle at frame 234. Our system

Table 4.1
Comparison between the data from the user study and our system.

video clip	#algorithm's outputs	# user selections	#correct hits	# missed points	#false alarms	precision	recall	F measure
1	10	100	87	34	13	0.87	0.66	0.75
2	10	100	83	36	17	0.83	0.64	0.72
3	6	60	51	13	9	0.85	0.78	0.82
4	7	70	57	28	13	0.81	0.60	0.69
5	3	30	30	9	0	1.00	0.70	0.82
6	9	90	83	20	7	0.92	0.78	0.84
7	7	70	64	13	6	0.91	0.81	0.86
8	7	70	66	20	4	0.94	0.71	0.81
9	5	50	39	13	11	0.78	0.74	0.76
10	12	120	97	47	23	0.81	0.61	0.69

Table 4.2
The statistics for recall, precision and F parameters.

	Mean	Variance	Standard Error
Precision	0.8732	0.0049	0.0221
Recall	0.7038	0.0057	0.0239
F measure	0.7775	0.0039	0.0196

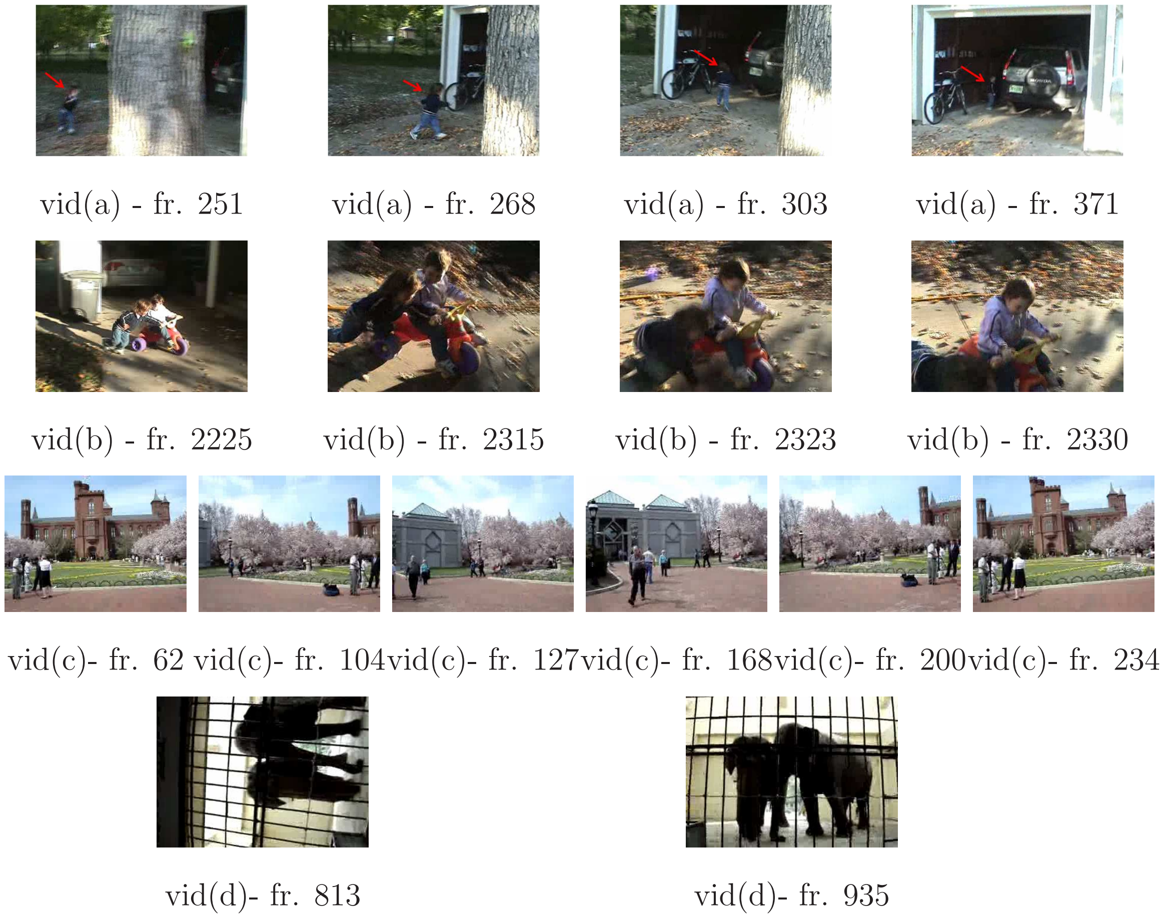


Fig. 4.8. Examples where the output of our system fails with respect to the user study. Each row contains the frames extracted from a video sequence, e.g. vid(a) represents the frames from video sequence (a), and “fr.” stands for the word frame.

does not discard frames from similar views taken at different times since those frames might be semantically different due to a change of the objects in scene. In case (d) the camera has a 90° rotation and some users selected the rotated frame (frame 813) as the keyframe whereas our system selected frame 935. Since the motion model we used is a 3-parameter model, it does not account for rotation. We noted that intentional rotation is not a frequent pattern in home videos.

4.3.3 Data Analysis - Objects of Interest

For each object that a user selected, we checked if it has been included in the identified ROIs of at least one of the extracted keyframes from the output of our system. The users identify the object of interest by including a description of the object and the frame it appears in the video. If the object appears in more than one occasions, the user did not need to cite the same object more than once. The subjects were allowed to select as many objects that they thought were important. A total number of 360 objects were selected by the ten subjects from the ten videos which results in an average of 3.6 objects per video per subject. 314 of these objects (87.22%) appeared in the highlighted set of keyframes extracted by the algorithm. Most of the objects missed where small objects that appeared in video for a very short period of time. Again, the results not only assess the quality of saliency map and ROI identification step but also validate our view segmentation which makes the set of keyframes an enclosing set of important objects in the video.

5. VIDEO ANNOTATION USING LOCATION METADATA

After generating a video summary, the system uses the keyframes and the Geo-location metadata of the video to annotate the video. The annotation component in our system is highlighted in Figure 5.1. The system utilizes user generated tags in a geo-tagged image database for automatic video annotation.

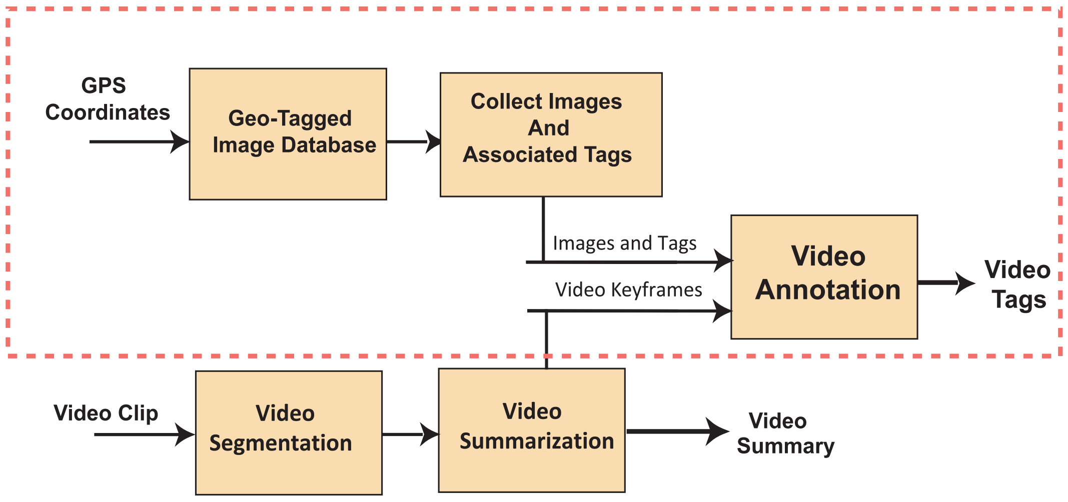


Fig. 5.1. Video annotation component in our system outlined by the dashed rectangle.

Millions of pictures are uploaded to the photo sharing websites such as Flickr and Picasa everyday. These pictures are often associated with various tags or keywords including information about the content, time, date, camera used and more recently geo-location. Our goal is to find tags that have high probability of being relevant to the video in such databases. The system does not require any high-level complex analysis such as object and action recognition of the video sequence. It is also flexible in the sense that each step of the process can be enhanced independently without necessarily changing the structure of the system. The major steps in our UGV annotation approach are the following.

- Geo-Matching
- Keyframe Matching
- Tag Processing

Location information, visual similarity and the distribution of tags are the important features used in the process. Figure 5.2 illustrates the block diagram of our annotation approach. Each step of the process is explained in more detail in the sections below.

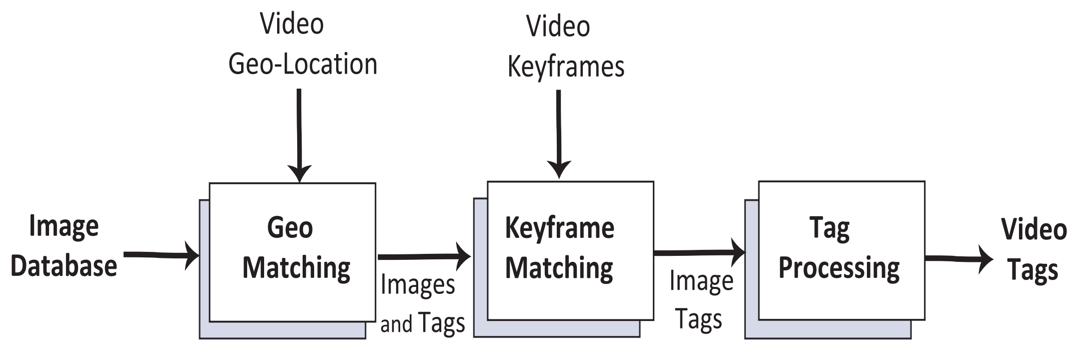


Fig. 5.2. Video annotation steps.

5.1 Geo-Matching

In this step we use the geo-location information of the video to find annotated images with the same coordinates in an image database and use them to annotate the video. The input video and the images in the database are assumed be geo-tagged, i.e. their location information is available. The assumption that the video is geo-tagged is a feasible assumption since most mobile devices that are equipped with a camera usually have a GPS unit as well. However, to ensure the usability of the system, we added the feature so that the user can input an address to the annotation system.

The system converts the address to the corresponding GPS coordinates using the Google Map API.

The GPS coordinates of the video or the equivalent address is used as a query to search for images taken in the same location as the video or nearby. A predefined number of images and their tags are collected to be used in the next step. We use the Flickr API to search for and download images and their associated tags from the Flickr photo sharing website.

5.2 Keyframe Matching

After collecting images taken in the same location as the video and their tags, we need to decide which tags are related to the video. For this purpose, the extracted video keyframes are used as input queries to find the most visually similar images in the set of collected images. The algorithm ranks the images in the set based on their distance from the keyframes and the most similar images are selected. We examined two different approaches for keyframe matching. The first method is based on using low-level visual features and the second one uses local descriptor to find matching points between the images and the keyframes.

5.2.1 Image Matching Based on Low-Level Visual Features

In this approach color and texture features are used to measure the visual similarity. To compare two images, or in this case one of our keyframes and an image from the database, first we divide each into non-overlapping macroblocks (MBs) with equal sizes (See Figure 5.3). We divided the height and width of each image into 10 segments therefore we obtain 100 MBs for each image. The features used are independent of the MB size so two images with different sizes can also be compared to each other. For each MB in the query image, we find the best matching MB in the test image and determine the distance between them. The overall distance between two images I_1 and I_2, is defined as the sum of distances between the best matching MBs, i.e.

$$D(I_1, I_2) = \sum_{i=1}^{N} D(MB_{1,i}, MB_{2,m(i)}) \tag{5.1}$$

where $MB_{1,i}$ is the i^{th} MB in image 1 and $MB_{2,m(i)}$ is its best matching MB in image 2 and N is the number of macroblocks in image 1. The distance measure between the MBs is based on texture and color features as described below.

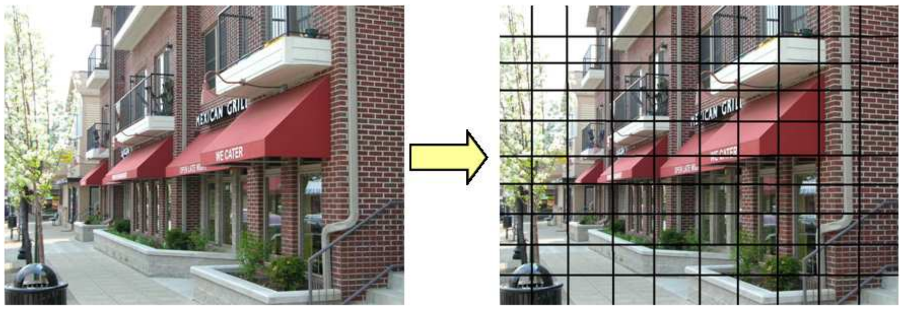


Fig. 5.3. Dividing the image into non-overlapping macroblocks.

Color Features

Color is one of the most widely used features in image retrieval. The color feature used here is the color histogram. First, an image is color-quantized uniformly in the RGB color space. The color histogram vector of each MB in the quantized image is then used as the color feature vector of that MB. The number of levels in each dimension is 8 resulting in a 512-bin histogram.

Texture Features

Texture is another important feature in content-based image retrieval. We tried two sets of texture features: Gabor filters and Gray Level Co-occurrence Matrices (GLCM).

Gabor Filters

Gabor filters have proved to be very useful texture analysis and are widely adopted in the literature. A two dimensional Gabor filter $g(x, y)$ can be defined as [63]

$$g(x,y) = \exp\left[-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2}\right) + 2\pi jWx\right], \quad a > 1, m, n : integer \tag{5.2}$$

where W is the modulation frequency. The self-similar Gabor wavelets are obtained through the generating function

$$g_{mn}(x,y) = a^{-m} g(x', y') \tag{5.3}$$

where $x' = a^{-m}(x \cos \frac{\pi n}{K} + y \sin \frac{\pi n}{K})$ and $y' = a^{-m}(-x \sin \frac{\pi n}{K} + y \cos \frac{\pi n}{K})$. The parameters m and n are the scale and orientation of the wavelet function, and K and S are the total number of orientations and scales, respectively.

The Gabor wavelets are a nonorthogonal basis set. This implies redundancy in the filtered images. We chose the filter parameters the same as in [63] which are designed to reduce the redundancy. This selection of parameters ensures that the half-peak magnitude support of the filter responses in the frequency spectrum touch each other (See Figure 5.4).

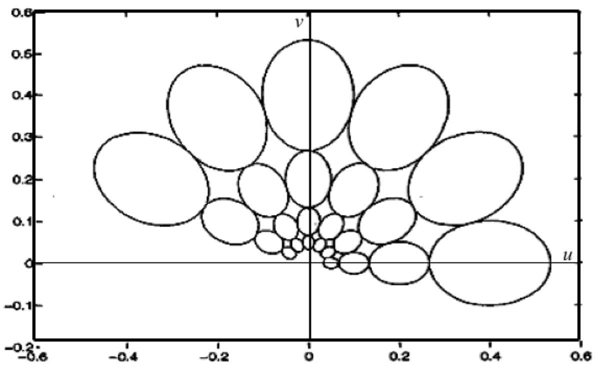


Fig. 5.4. The contours illustrate the half-peak magnitude of the frequency response of Gabor filter banks. The minimum and maximum frequencies are chosen to be 0.05 and 0.4 respectively. $S = 4$ and $K = 6$ in this case.

The Gabor wavelet transform of an image I is defined as

$$W_{mn}(x,y) = I(x,y) * g_{m,n}(x,y) \tag{5.4}$$

where $*$ denotes the 2-D convolution. Assuming that texture features are locally homogeneous, the mean μ_{nm} and standard deviation σ_{mn} of the magnitude of the transform coefficients, $|W_{mn}|$, are used to represent the texture in the area. The texture feature vector consists of μ_{nm}'s and σ_{mn}'s for $0 \leq m \leq 3$ and $0 \leq n \leq 7$. Thus, for each MB we have 64 features, i.e. $\mathbf{f}_g = [\mu_{00}\ \sigma_{00}\ \mu_{01} \ldots \sigma_{37}]$.

GLCM Features

The GLCM features were suggested by Haralick *et al* [64] for texture analysis. GLCM conveys the information about spatial relationship of pixels by means of the occurrence of pairs of gray levels in the image. The GLCM is an estimate of the two dimensional probability distribution of the pixels and depends on the the distance between the pixels and their angular displacement [64]. Assume that the gray level values are quantized uniformly to N_g levels. A $N_g \times N_g$ gray level co-occurrence matrix $P_{\mathbf{d}}$ is obtained for a displacement vector $\mathbf{d} = (d, \theta)$ as follows. Each entry of the matrix, $P_{\mathbf{d}}(i,j)$, is the number of occurrences of gray level pairs i and j that distance $\mathbf{d}$ apart:

$$P_{\mathbf{d}}(i,j) = |\{\{(u,v),(t,s)\}|I(u,v) = i, I(t,s) = j\}| \tag{5.5}$$

where, I is the gray level image, (u,v) and (t,s) are the pixel locations in the image where $(t,s) = (u,v) + \mathbf{d}$, and $|\ .\ |$ denotes the cardinality of a set. The matrix $P_{\mathbf{d}}$ is normalized to overall sum of the matrix so that the matrix represents the co-occurrence probability estimation, i.e. $P_{\mathbf{d}} = \frac{P_{\mathbf{d}}}{\sum_{i,j} P_{\mathbf{d}}(i,j)}$. The texture features are then obtained from $P_{\mathbf{d}}$ matrix as follows.

1. Angular Second Moment

$$f_{\mathbf{d}}^1 = \sum_{i,j=1}^{N_g} P_{\mathbf{d}}(i,j)^2 \tag{5.6}$$

2. Dissimilarity
$$f^2_{\mathbf{d}} = \sum_{i,j=1}^{N_g} P_{\mathbf{d}}(i,j)|i-j| \tag{5.7}$$

3. Correlation
$$f^3_{\mathbf{d}} = \sum_{i,j=1}^{N_g} P_{\mathbf{d}}(i,j)\frac{(i-\mu_i)(j-\mu_j)}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}} \tag{5.8}$$

   where μ_i and μ_j are the means obtained as $\mu_i = \sum_{i,j} iP_{\mathbf{d}}(i,j)$ and $\mu_j = \sum_{i,j} jP_{\mathbf{d}}(i,j)$, and σ_i^2 and σ_j^2 are the variances, i.e. $\sigma_i^2 = \sum_{i,j}(i-\mu_i)^2 P_{\mathbf{d}}(i,j)$ and $\sigma_j^2 = \sum_{i,j}(j-\mu_j)^2 P_{\mathbf{d}}(i,j)$.

4. Entropy
$$f^4_{\mathbf{d}} = \sum_{i,j=1}^{N_g} -P_{\mathbf{d}}(i,j)\log P_{\mathbf{d}}(i,j) \tag{5.9}$$

5. Contrast
$$f^5_{\mathbf{d}} = \sum_{i,j=1}^{N_g} P_{\mathbf{d}}(i,j)(i-j)^2 \tag{5.10}$$

These 5 features are calculated for three different length of distance vector $d = \{1, 2, 5\}$ and four different angles $\theta = \{0^o, 45^o, 90^o, 135^o\}$. As a result, for each MB we have 60 co-occurrence features.

Similarity Measurement

We investigated the use of both Gabor and GLCM texture features for matching. The results were similar for both set of features. However, in the experiments we conducted using our image database, Gabor filters were found to be computationally more expensive than GLCM features. Therefore, we used the GLCM set of texture features for our image similarity measurement. The metric used for comparison between the texture feature vectors was the Euclidean distance.

In order to obtain the color distance measure the Bhattacharyya similarity measure was used. The Bhattacharyya similarity measure, ρ, between two normalized histogram p and q can be defined as

$$\rho = \sum_i \sqrt{p_i q_i} \tag{5.11}$$

where, p_i and q_i are the values of the i^{th} bin of the normalized histograms. We use $1 - \rho$ as the distance metric for comparing color histograms. Note that the minimum distance for this measure is zero if the histogram vectors are exactly the same and one if they are orthogonal to each other. Hence, to combine the texture and color metrics, we have to normalize the texture distance. We do this by dividing the distance values by the maximum texture distance obtained in the image dataset used for the keyframes retieval. The total distance between two MBs is then obtained as the weighted sum of the color distance, D_C, and the normalized texture distance, D_T.

$$D_{tot} = W_C D_C + W_T D_T \tag{5.12}$$

The weighing parameters, W_C and W_T were selected 0.5 in our experiments.

5.2.2 Image Matching Using Local Descriptors

Another class of image matching methods find correspondences between similar objects in the images. These methods look for instances of the same object and not a class of objects of same type. First, a set of interest points or keypoints are determined in each image by the algorithm. Then, a local descriptor is assigned to each keypoint based on some characteristics of its neighboring pixels. Images are matched by finding the best matching keypoints based on a distance metric. Harris corner detector [65], Harris-Laplace [66], SIFT [67] and ASIFT [68] are a few examples of such proposed techniques. These image matching techniques are particularly useful in applications such as image registration, panorama stitching, robot localization, object tracking and 3D scene reconstruction. Recently, they have been used in image and video retrieval.

Among the several descriptors that have been proposed in the literature, Scale Invariant Feature Transform (SIFT) is one of the most successful descriptors and has been broadly used for many applications. SIFT descriptors are invariant to rotation, scale and translation. According to a performance study by Mikolajczyk and Schmid [69], SIFT descriptors proved to have superior performance compared to many other descriptors. A number of variations of SIFT has been proposed to improve the efficiency or the performance of this method, e.g. SURF [70], PCA-SIFT [71] and ASIFT [68]. We used SIFT features for the purpose of keyframe matching. There are four major steps for extracting SIFT features [72]: scale-space extrema detection, keypoint localization, orientation assignment and keypoint descriptors. Here, we briefly describe these steps.

SIFT Descriptors

The scale space of an image, I, is defined as the convolution of the image a Gaussian kernels of different variances.

$$\begin{aligned} L(x, y, \sigma) &= G(x, y, \sigma) * I(x, y) \\ G(x, y, \sigma) &= \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \end{aligned} \tag{5.13}$$

The keypoints are selected as the scale-space extrema in the function, $D(x, y, \sigma)$, which is obtained by convolving the image with the difference-of-Gaussian (DoG) function.

$$\begin{aligned} D(x, y, \sigma) &= [G(x, y, k\sigma) - G(x, y, \sigma)] * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{5.14}$$

The DoG function is an approximation of the scale-normalized Laplacian of Gaussian whose extrema are found to be the most stable image features compared to a number of other features such as gradient and Harris corners [65]. Once, $D(x, y, \sigma)$ functions are obtained in different scales, the local extrema of these function are found by comparing each point to the eight neighbors in the current D function and the nine neighbors in the scale above and below. Next, the unstable keypoints that have low contrast or are located along the edges are discarded from the keypoint set. In order to achieve rotation invariance in the feature, the orientation of each keypoint is obtained and assigned to the keypoint. To obtain the orientation of a keypoint at location (x_i, y_i) and scale σ_i, a histogram of the orientations of gradients of $L(x, y, \sigma_i)$ is estimated in a circular window centered at (x_i, y_i). The gradient orientations, added to the histogram, are weighted by a Gaussian function and by the magnitude of the gradients. The peak of the histogram is selected as the orientation of the keypoint. If there are more than one peak in the histogram, more keypoints are added at the same location and scale but the orientation of the other peaks.

So far each keypoint contains information about the location, scale and orientation. However, a descriptor is needed to distinguish between different keypoints. The descriptors are determined as follows. First, a spatial window around the keypoint is considered in the Gaussian image, L, in the scale of the keypoint. The orientation and magnitude of the gradient vectors are obtained for each pixel in the window 5.5. The magnitude values are weighted by a Gaussian function to achieve better stability. The coordinates of the descriptor and the gradient orientations are rotated according the the keypoint orientation to achieve rotation invariance. A separate orientation histogram is estimated for each region of size 4×4 samples. A total of $4 \times 4 = 16$ regions are used for each keypoint descriptor. In Figure 5.5 only $2 \times 2 = 4$ regions are displayed. Each histogram contains 8 bins resulting in a $4 \times 4 \times 8 = 128$ element feature vector for each keypoint. This vector is normalized to have unit length so that it is invariant to changes in image contrast. Moreover, since the descriptor is obtained from the gradient of the image, it is invariant to illumination changes. Figure 5.6 illustrates keypoint descriptors obtained by SIFT for two images.

Matching SIFT Keypoints

The distance metric used for comparing keypoints is the Euclidean distance between the corresponding descriptors. For each keypoint in one image, the nearest

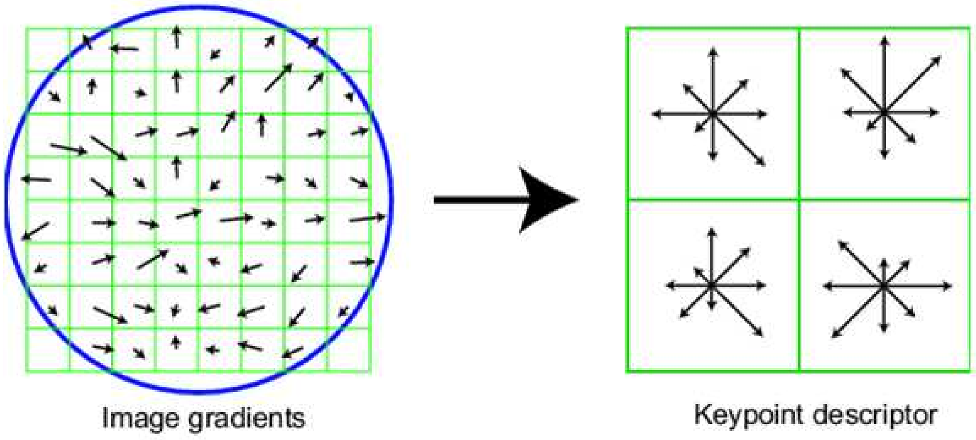


Fig. 5.5. A SIFT keypoint descriptor.

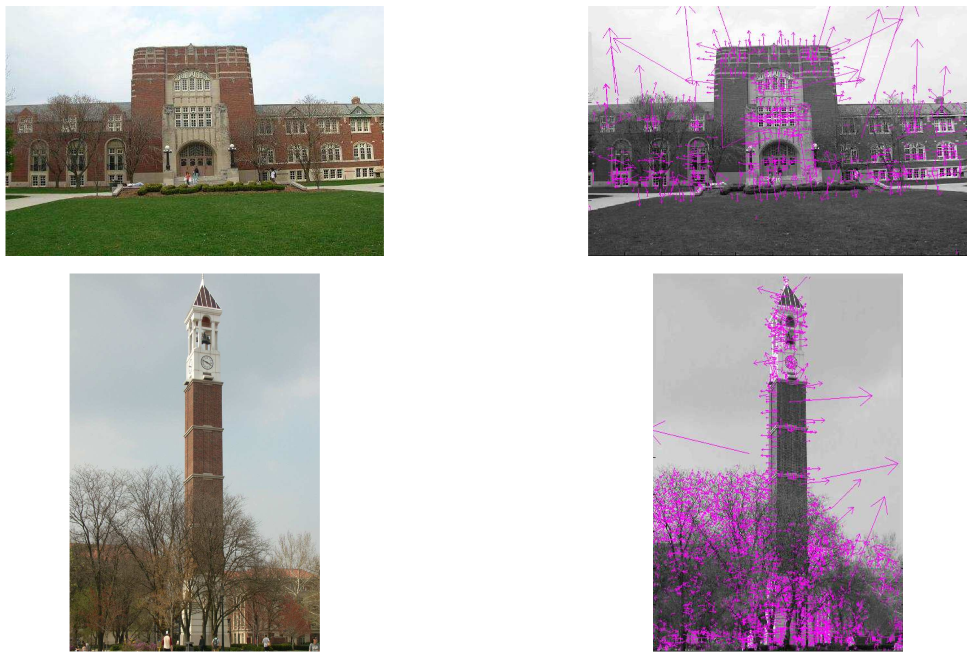
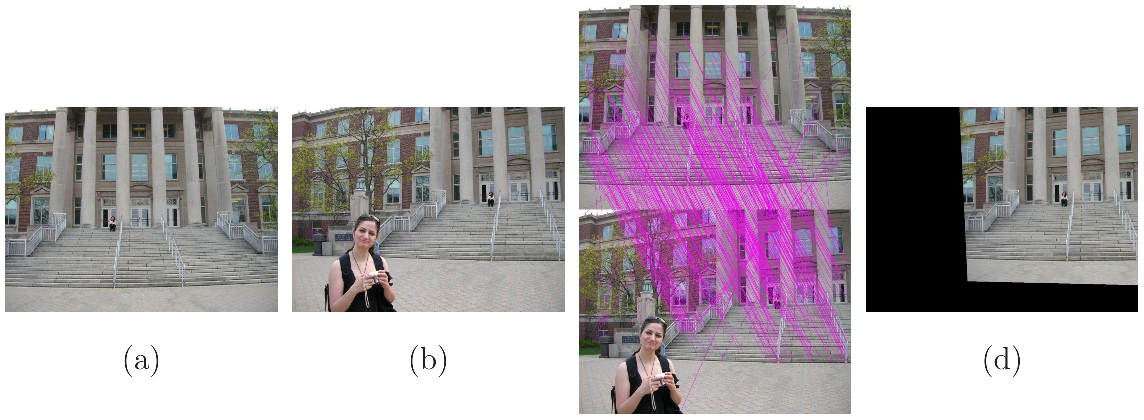


Fig. 5.6. SIFT descriptors (magenta arrows) located at the keypoints. The magnitude and orientation of each arrow show the relative scale and orientation associated with the corresponding keypoint.

neighbor in the set of keypoints in the second image is chosen as the match. For some keypoints there may not be a match in the second image. These cases are detected by setting a constraint on the ratio of distance of the nearest neighbor to that of the second nearest neighbor. This ratio has to be less than 0.8 which means in order for a match to be reliable, the nearest point must be significantly closer than other points. Otherwise, the keypoint is considered to have no match in the second image. Since the matching is done only based on local descriptors, there can be mismatches in the results. These mismatches can be due to occlusion, depth discontinuity and repetitive patterns. The RANSAC algorithm [73] can be used to detect the outliers based on fitting the matching points into a global motion model. An eight-parameter model is used here. Figure 5.7 shows an example of using SIFT followed by RANSAC for image matching. The matched keypoints between images in parts (a) and (b) are shown in part (c). Note that this image contains some outliers. RANSAC is used to find a transform and identifying the mismatches. The transform parameters obtained by RANSAC are used to warp Figure 5.7-(a) to match Figure 5.7-(b). The transformed image is shown in Figure 5.7-(d).



(a) (b) (c) (d)

Fig. 5.7. Matching images (a) and (b) using SIFT algorithm. The result of keypoint matching is shown in (c). After applying the RANSAC algorithm, the set of transform parameters is used to warp image (a) to match image (b), resulting in the transformed image in (d).

5.3 Tag Processing

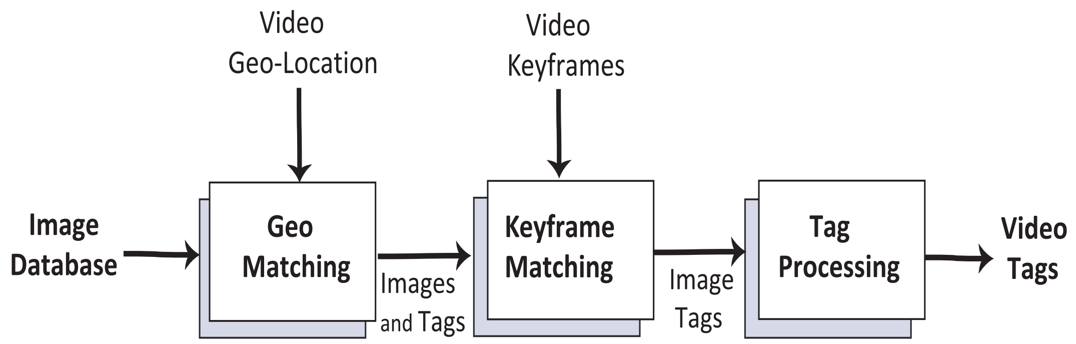


Fig. 5.8. Video annotation steps.

At this point we have gathered a set of images that are taken in the same location as the video and are visually similar to the video keyframes. The tags associated with this set of images are extracted for this step which we call the tag processing step (See Figure 5.8). Identifying whether or not a tag is actually related to the video needs understanding the nature of the tag and high level video processing such as object recognition. However, we use a simple approach to identify the tags that have higher probability of being relevant to the video. More sophisticated tag processing methods can replace this unit without changing the structure of the system. In order to select the tags, a tag histogram is created from the extracted tags. Tags are then sorted based on the frequency of their appearance. The most frequent tags are identified as "dominant tags" and are used to annotate the video. This set of tags are suggested to the user as potential tags for the video. The user has the option to keep or discard each of these tags. The number of tags to be returned by the system is a predefined user-selected parameter. Figure 5.9 shows the dominant tags obtained from Flickr images for the keyframe located on the top right corner of figure.

User generated tags can be very noisy. Some tags are personal and cannot be generalized to other photos or videos taken by another user, for example names of the people in the photos. Moreover, some photos can contain incorrect tags that are completely irrelevant to the photo. By using the most frequent tags gathered from

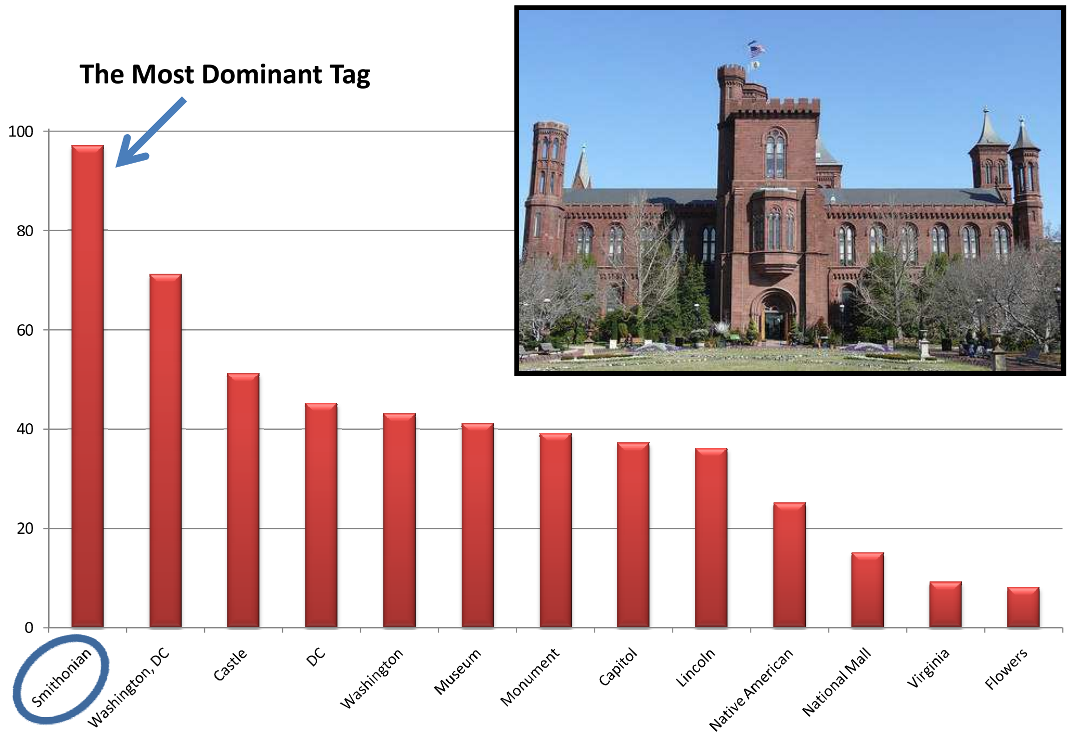


Fig. 5.9. The most frequent tags and their corresponding frequencies for the frame on the top right corner. These tags are extracted from 200 similar images obtained from Flickr.

several users, the probability of one of these tags appearing in the final set of tags decreases. Another misleading case is when a user uploads several photos at a time and associate one set of tags to all the photos. In this case not only, he contributes to the wrong tags for a number of images but also he populates these tags. This results in having incorrect tags with high frequency that can decrease the accuracy of the results substantially. One way to address this problem is to allow each user to contribute only one photo to the list.

5.4 Experimental Results

We have processed several videos recorded by different individuals in the range of 30s to 3min. All videos were segmented into camera views and summarized in a set of keyframes according to the methods described in Chapter 2. As for the image database, there are two databases that we are currently using:

Purdue University Campus Database

We have gathered approximately 250 images from the Purdue University campus and nearby areas that are all geo-tagged using a handheld navigator device. The resolution of these images is 1024×768 pixels. This database is not annotated with other tags and is used as a testing framework for our image retrieval methods. A sample of this image database is depicted in Figure 5.10.

Figure 5.11 shows two examples of the result of image retrieval using the approach described in Section 5.2.1. The images on left are used as a query in a set of 55 images taken in the same proximity. The 10 most visually similar images returned by the algorithm are shown on the right side of the figure.

Flickr Image Database

The system is also capable of searching Flickr database and downloading a specified number of images and their associated tags using the Flickr API. As the number of images used increases, the annotation results become more accurate since it reduces the effect of noisy tags. However, more computations need to be done in the kyeframe retrieval step. We currently use 200 images for each set of GPS coordinates. Due to the fact that online geo-tagged image databases are not highly populated at this time, the system may not find the specified number of images. In that case, it converts the coordinates into the street address and place name, and searches the database based on these tags. This problem will eventually disappear since the number of images equipped with location information is growing rapidly.

Figure 5.12 illustrates keyframes extracted from two different videos. Video (a) (left keyframe) is taken in front of the Smithonian Castle in Washington DC. Video (b) (right keyframe) is recorded in Sausalito, CA that has a view of downtown San Francisco. The dominant tags found by our algorithm for the first video using Flickr database include: “Smithsonian, Washington DC, castle, museum, monument, capitol, Lincoln, national mall, garden and flowers.” For the second video, the dominant

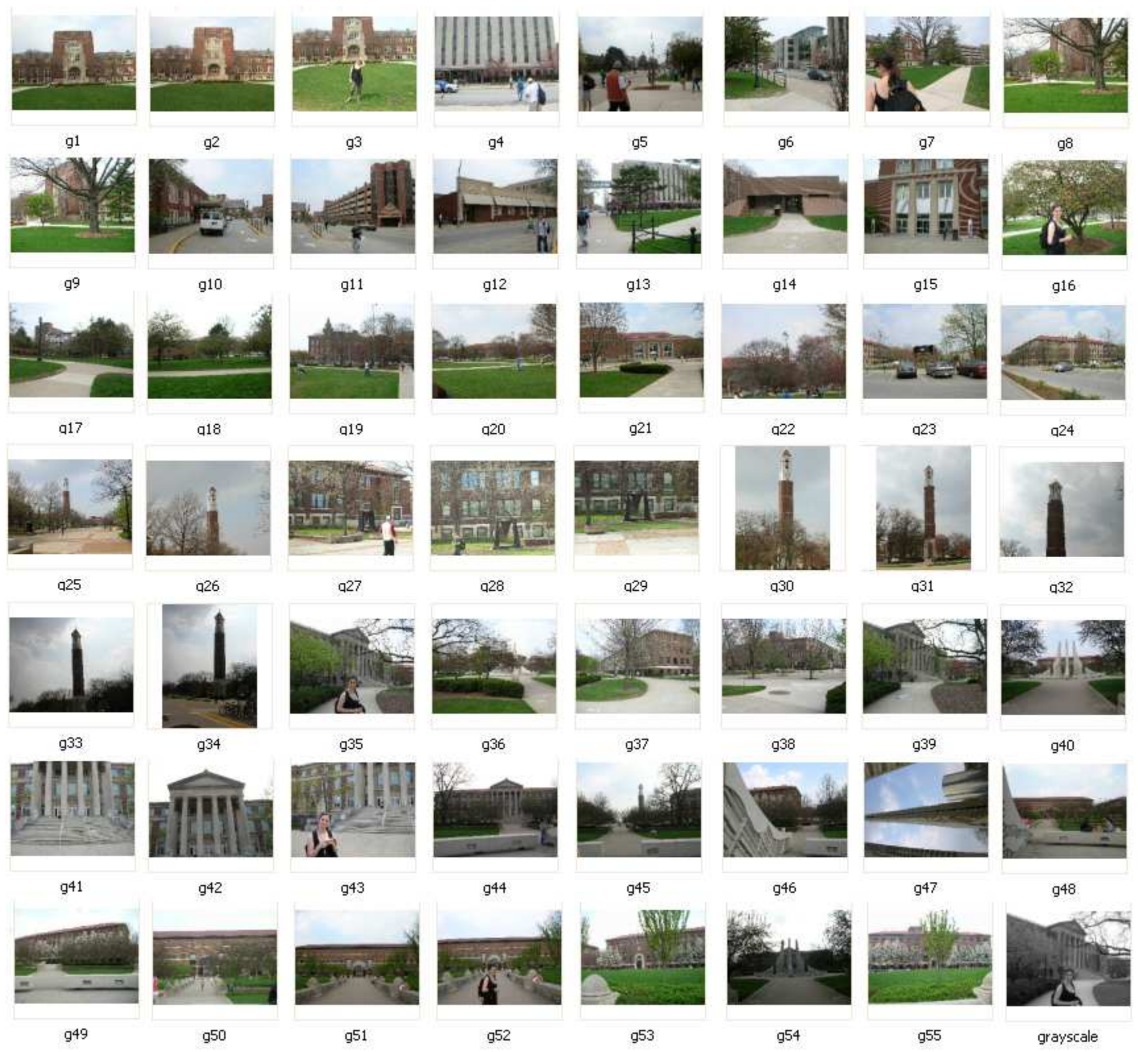


Fig. 5.10. A sample of Purdue campus image database. This database contains 250 images are of size 1024×768 pixels.

tags are: "Sausalito, San Francisco, bay area, California, marina, coast, travel, family and ferry."

We observe that the best results were obtained for tags related to the static objects in the background such as buildings, bridges, statues, towers and even for some time-related tags for example "sunset sky" or "spring blossoms."

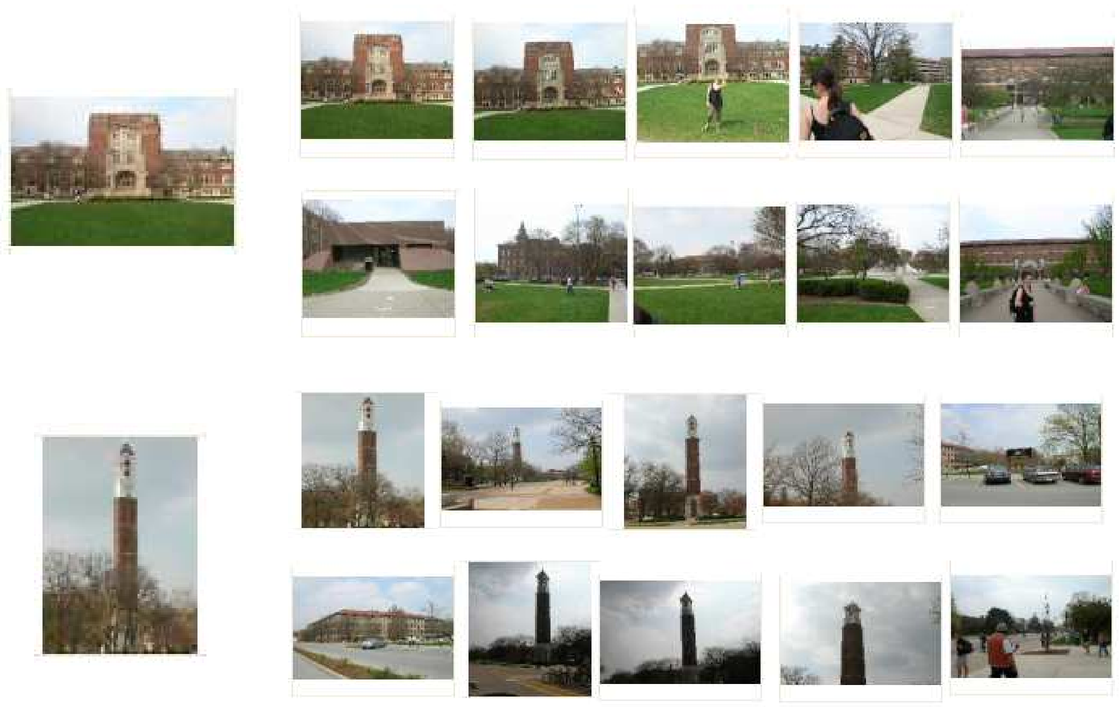


Fig. 5.11. Retrieval results from the Purdue University Campus database using low-level visual features. Images on the left are used as query examples. The 10 most visually similar images found by the system are shown on the right.

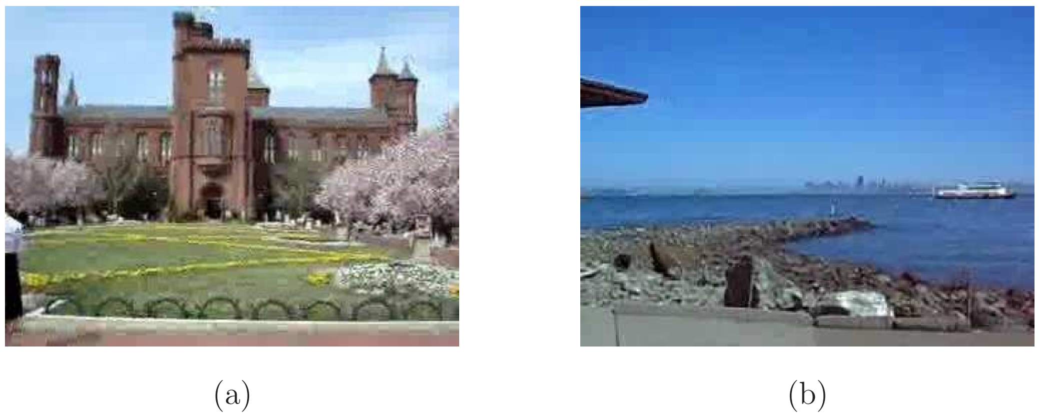


Fig. 5.12. Keyframes extracted from two different videos.

5.4.1 Experimental Results of Image Matching Using SIFT

SIFT descriptors are very successful in matching the images of the same scene that are taken in almost the same lighting conditions and when the motion is limited to rotation, zoom and translation. For this reason, SIFT is an excellent choice for applications such as registration and panorama stitching. Figure 5.13 shows two examples of successful image matching using SIFT. In the top row, the camera has panned to the left and in the bottom row there is a camera zoom-in from one picture to the other. As the images in the right column show, SIFT finds several correct matches between the images on the left.

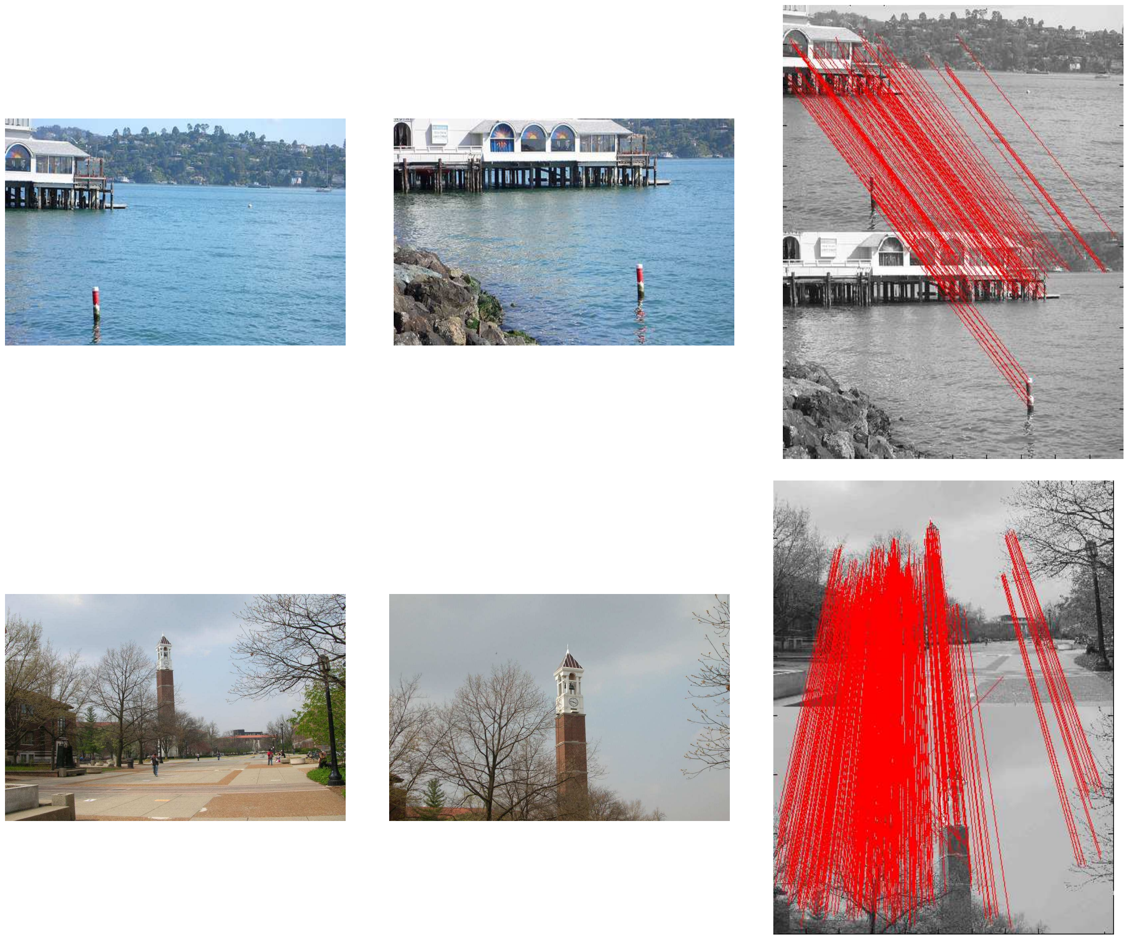


Fig. 5.13. Successful cases of image matching using SIFT.

However, for the purpose of image matching in the video annotation application, SIFT can fail in a number of circumstances. Since we compare video frames to images taken by different cameras with different resolutions and in various times of the day or year, there is a large variety among the images being compared. This can lead to poorly matched images taken from very similar scenes. Another disadvantage of this method is the heavy computational complexity that it adds to the system. Below, we illustrate examples of SIFT failure cases and discuss them in more details. We also propose a method that can reduce the complexly and improve the results.

An example of a failure case for the SIFT image matching approach is shown in Figure 5.14. The images shown on the top row of this Figure are taken from the Bell Tower on the Purdue campus from two different views. The set of keypoints obtained using SIFT algorithm are shown for each image in the second row. The number of keypoints for the image on the left is 3691 and is 2136 for the one on the right. SIFT matching technique finds only three matches between the two image which are shown on the bottom row of Figure 5.14. None of these three matches are correct. The majority of the detected keypoints fall on the tree branches. These keypoints are not very useful for the matching process. Nevertheless, extraction of them and their descriptors consumes a large amount of processing power. The keypoints that are located on the Bell Tower have also failed to match, which demonstrates a shortcoming of SIFT in cases with significant changes in view angle or lighting condition. In [72], the author argues that because an illumination change adds a constant to the image, and since SIFT descriptors are generated using the gradient of the image, these descriptors are robust to illumination changes. However, in most outdoor images taken at different times of the day or in different seasons, the lighting in the image does not change just by a constant. For example, cloud patterns have different illumination effects in different parts of the scene. SIFT is not very robust to this type of illumination changes.

In the example in Figure 5.15, the two images being compared are taken from the same fountain on the Purdue campus. These images are taken from two different angles, at two different times of the day. The corresponding keypoints are illustrated in the second row. SIFT locates 6735 keypoints for the image on left and 6388 keypoints for the one on right. Out of these keypoints, SIFT find only 13 matches with 6 of them being correct although the two images are visually very similar. Again,

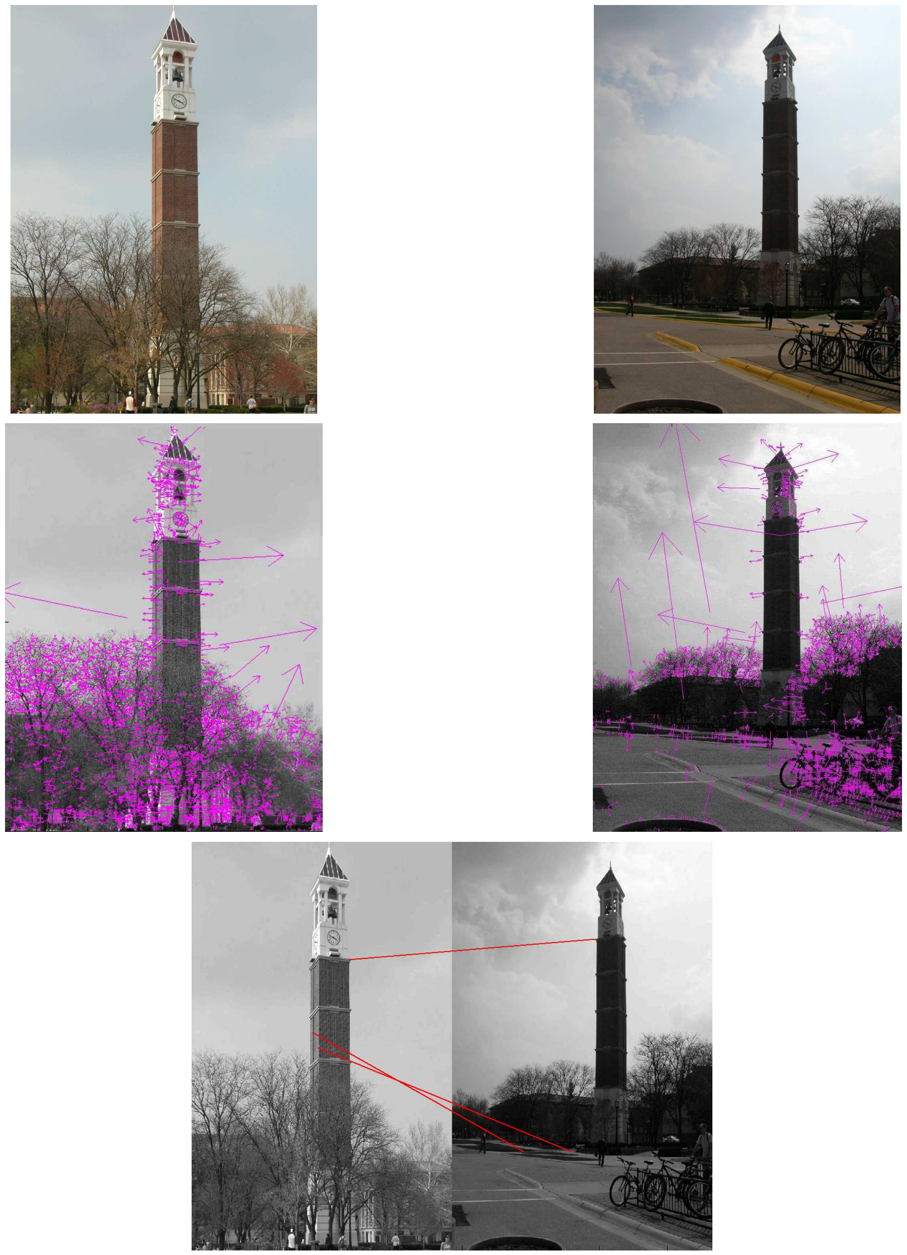


Fig. 5.14. An example of unsuccessful match by SIFT. A lot of keypoints are located on the tree branches and are not useful in matching. Other keypoints have failed to match due to changes in the view angle and lighting condition.

many of the detected keypoints are located on the textured area of the tree branches. Moreover, the change of lighting affects the descriptors considerably.

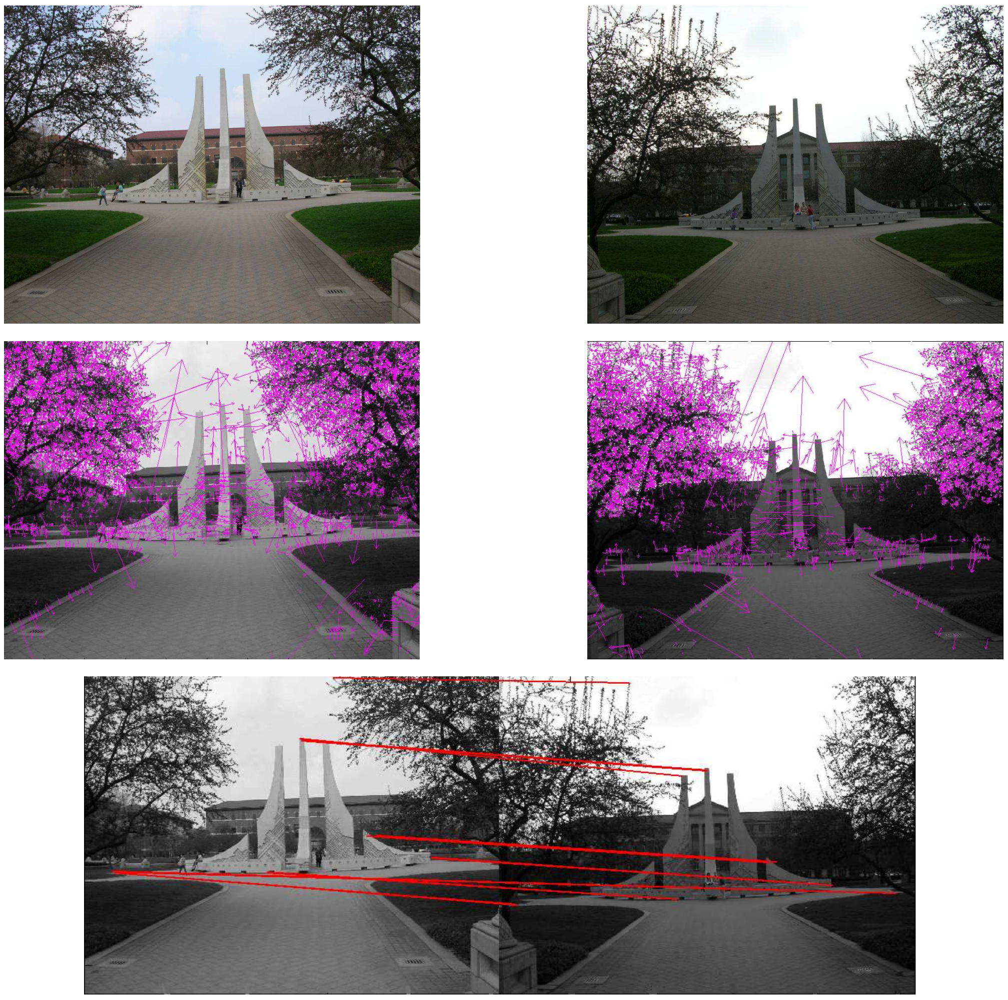


Fig. 5.15. An example of image matching using SIFT. Although the images are visually very similar, the number of matches found by SIFT is relatively small.

Another important case that shows a deficiency of SIFT is illustrated in Figure 5.16, the two images being compared are taken from the same scene with slightly different distances of camera with the respect to the building in the background. The corresponding keypoints are illustrated in the second row. SIFT locates 3525 keypoints for the image on left and 2472 keypoints for the one on right. Out of these keypoints, SIFT finds 27 matches. Many of these matches are not correct. The problem that causes many of these mismatches is the fact that the building and also the side stones have self similar or repetitive patterns. Therefore, keypoints from one object can match the keypoints in another similar object.

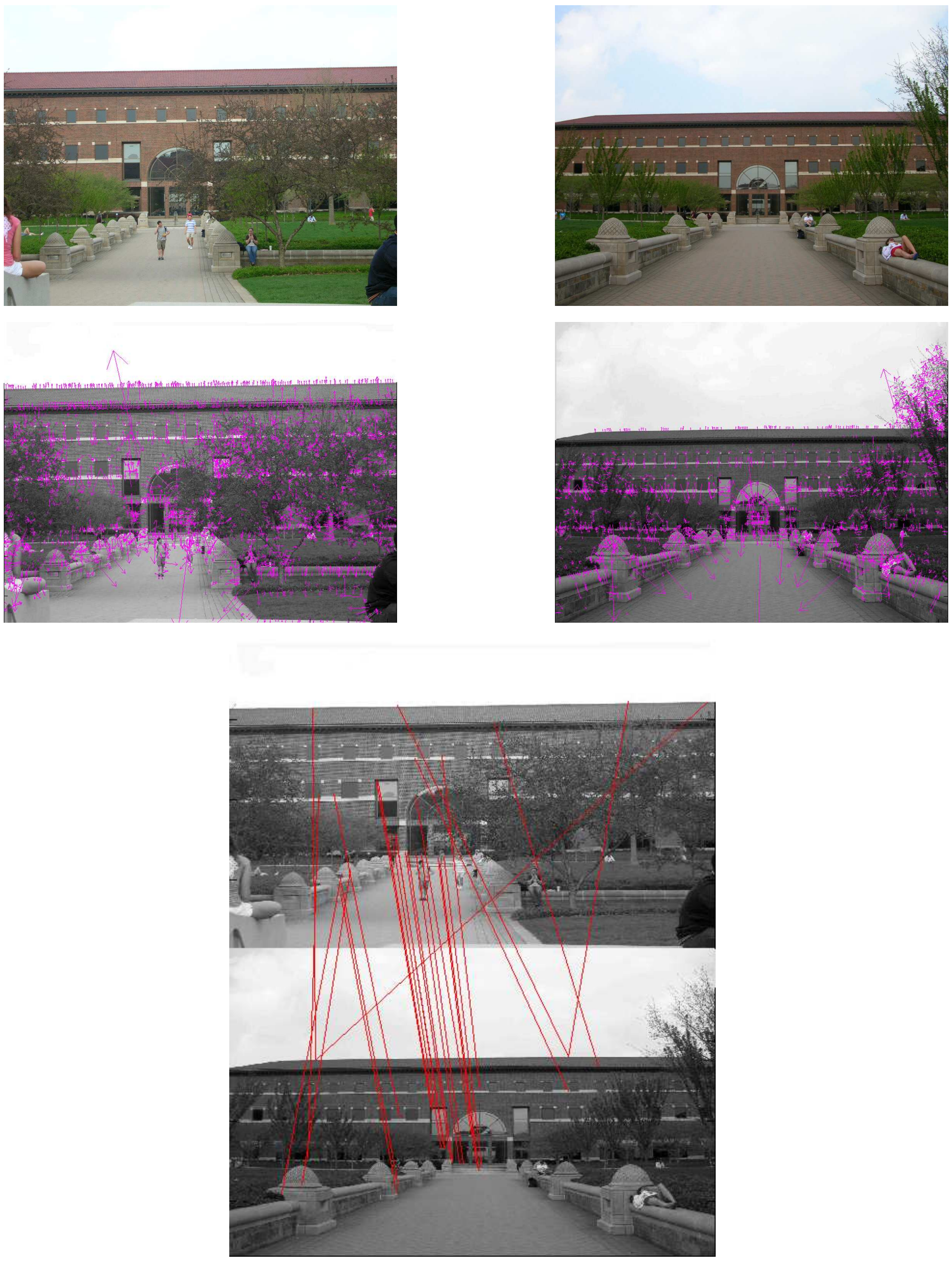


Fig. 5.16. An example of image matching by SIFT where there are a lot of mismatches due to the presence of several repetitive patterns.

5.5 Combining Color and Texture Features With The SIFT Descriptors

Various examples shown in Section 5.4 demonstrate the fact that using either of the low-level visual features or the local descriptors does not provide a universal solution for the problem of keyframe retrieval. While local descriptors are useful in matching the static background objects such as buildings and statues, color and texture can be important to measure the global similarity between images. For example, in Figure 5.17 the two image are from the same scene, one with portrait view and one with landscape view, which is taken slightly closer to the scene. These images “look” very similar and the distance measure based on color and texture is small. However, they do not contain many “keypoint-like” features and SIFT locates 294 keypoints in the left image and 215 keypoints in the image on the right. There are 9 total matches found by SIFT between these two images. Six out of these nine matches are correct. Thus, using SIFT for matching is not an appropriate choice for this case.

Moreover, using only low-level feature is not very accurate and sometimes the results are affected by a large homogeneous texture area such as grass, sky or street asphalt. This can be seen in some of the results shown in Figure 5.11. Some texture areas can be important for verifying the existence of some tags. For instance, a water area can be used to verify the tag “ocean;” a sky area can verify certain time or weather related tags such as “sunny day” and “sunset;” and snow can verify some winter related keywords. Nonetheless, the texture area itself is often not the focus of the camera person. Moreover, if the texture area is large, it can dominate the global similarity distance between images. In addition, some texture areas such as tree branches or bricks can decrease the performance of the SIFT matching method. For example, in Figures 5.14 and 5.15 the majority of keypoints are located on the tree branches. These keypoints are of no value in the process of matching but they significantly increase the computational complexity of the SIFT algorithm.

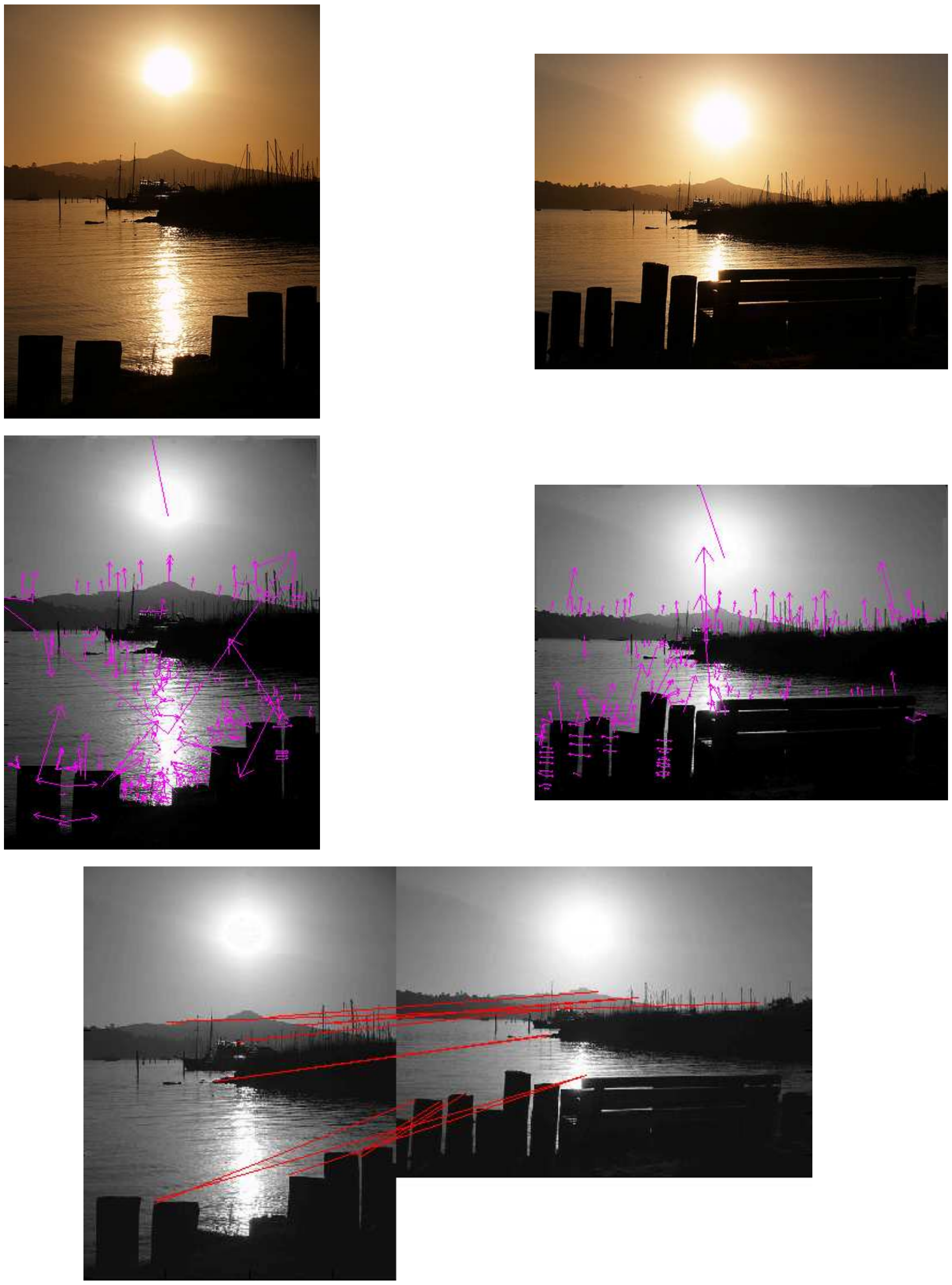


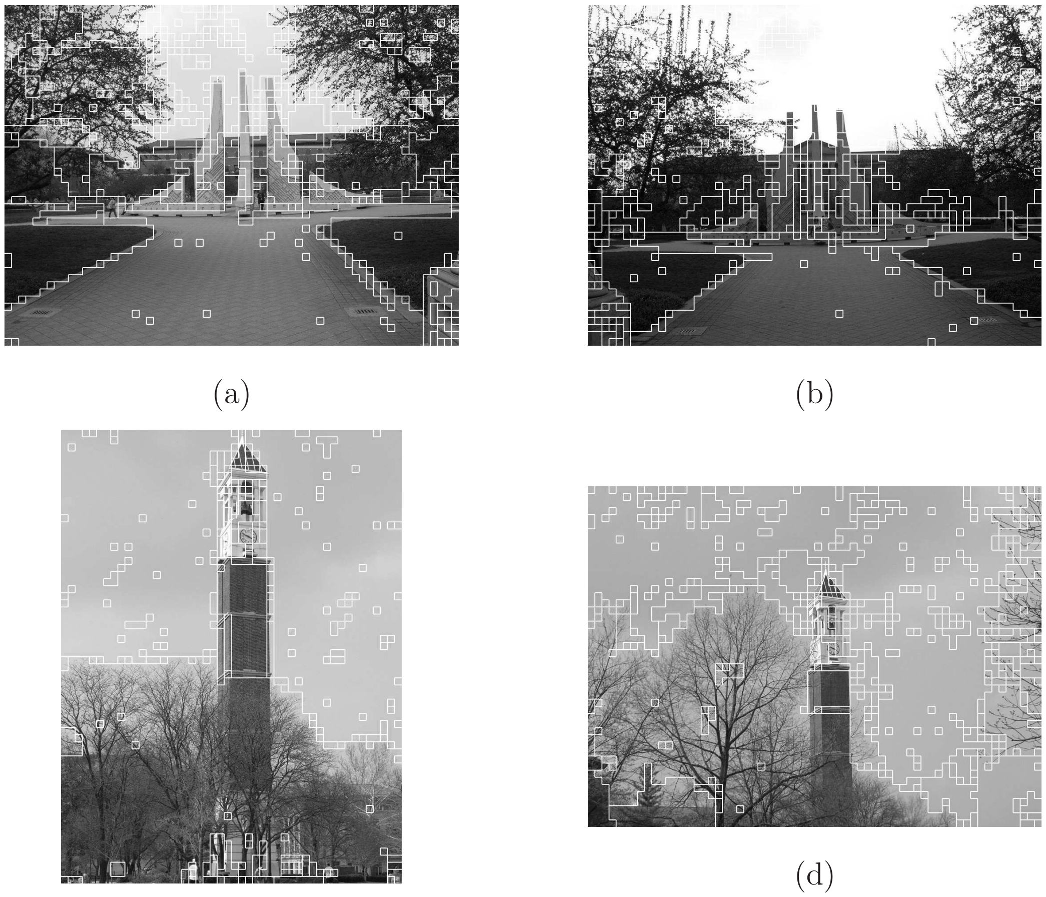
Fig. 5.17. An example of image matching where SIFT is not the appropriate choice.

5.5.1 Texture-Based SIFT Matching

In this section we describe a new method to combine color and texture features with SIFT descriptors. Large homogeneous areas are detected as texture areas, which will be excluded from SIFT as we describe below. In order to detect the texture regions, a method proposed in [74], which is a combination of quadtree segmentation and region growing algorithm, is used. In this method the image is divided into 16x16 non-overlapping macroblocks. Similar homogeneous macroblocks are then merged together based on color and edge histogram distances. In order to determine homogeneity, each macroblock is further decomposed into four non-overlapping regions. The macroblock is considered to be homogeneous if its four regions have pairwise similar statistical properties. Similar homogeneous macroblocks are then merged using a region growing algorithm. The results of this method for a number of images in our database is shown in Figure 5.18.

As Figure 5.18 shows, the results contain over-segmentation in the flat regions that belong to the sky area. To address this issue, a threshold is used on the standard deviation of the pixels in a macroblock. If this value is too small, the MB is considered as “flat” and is merged to the neighboring blocks. Each connected region that covers at least 1% of the image area is considered as a “texture area.” The similarity between texture areas in two images are obtained using the distance measure in Section 5.2.1. The identified texture regions for images in Figure 5.18 are shown as an overlaid mask on the original image in Figure 5.19.

The next step is to select the non-texture regions to determine the keypoints using SIFT. One solution would be to obtain all the keypoints and discard the ones located on the texture areas. However, the unused keypoints add unnecessary burden to the system and as we will show, by avoiding the extra computation, the method runs much faster for images that contain large detailed texture areas. Another approach is to use the masked images (Figure 5.19) as the input to SIFT. That way, we can save on the time for matching the keypoints that are located on the texture regions. There are two disadvantages for this approach. First, the time for calculation of the keypoints for the entire image is still significant. Secondly, since the mask borders align with the macroblock grid, the mask adds extra corners to the image that can



(a) (b) (c) (d)

Fig. 5.18. The result of the split and merge algorithm for identifying the homogeneous regions.

be mistaken as keypoints by SIFT. Thus, we use the following approach to overcome the mentioned problems.

The non-texture areas of the image are divided into rectangular patches that are used as individual inputs to the keypoint detection algorithm. In order to extract the patches, first, the texture mask is downsampled by a factor of 16 in each direction. Then a morphological opening step is done on the downsampled mask using a simple square kernel of size 3×3. This step discards the detected blocks that belong to the borders of two regions, as well as the isolated blocks. As a result the mask is segmented into separate regions. Figure 5.20 illustrate this process for an image

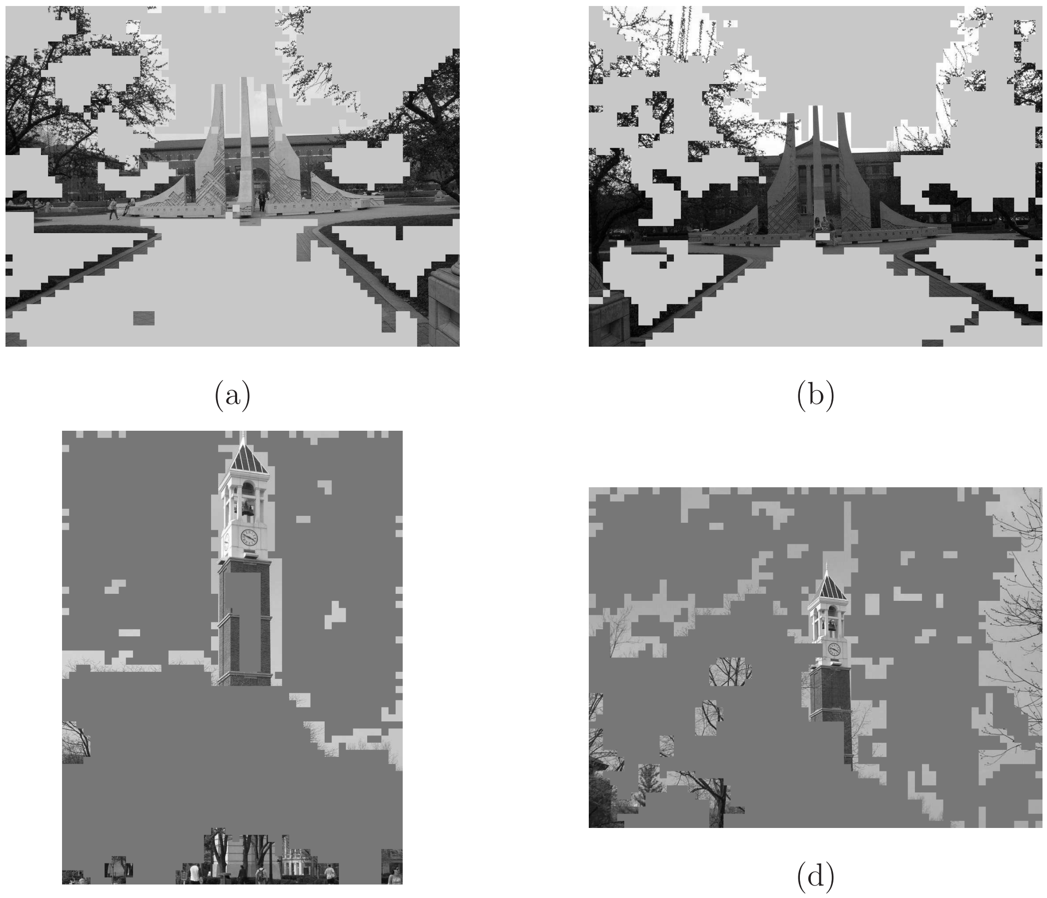

(a) (b) (c) (d)

Fig. 5.19. The extracted texture regions using the results from Figure 5.18.

shown in part (a). The initial downsampled mask is shown in (b), where the white pixels represent the non-texture blocks in the original image. Part (c) shows the mask after the morphological opening.

The bounding box for each connected white area is then obtained. Since the non-texture areas are usually not in rectangular shape, it is inevitable for the bounding boxes to contain a number of texture blocks. Our goal is to avoid the detailed texture areas as much as possible since these textures contain "keypoint-like" shapes that add to the complexity of SIFT. Therefore, we construct a cost function, C, for the image blocks that is based on the variance of pixels in the block. In other words,

Table 5.1
Statistics for the execution of SIFT on different versions (original version, extracted patches, and downsampled version) of images (a) and (b) in Figure 5.18.

Image	Original (a)	Original (b)	Non-Texture Area (a)	Non-Texture Area (b)	Downsampled by a factor of 4 (a)	Downsampled by a factor of 4 (b)
Number of Keypoints	6390	6060	3322	2593	450	367
Average time of matching (s)	52.3484		17.4423		2.3433	

$C(i,j) = \sigma_{ij}$, where σ_{ij} is the standard deviation of the macroblock MB_{ij}. An example of the cost function is shown in Figure 5.20 part (d). The cost of a bounding box, B, is as follows.

$$C_B = \frac{\sum_{(i,j)\in W_B} C(i,j)}{A_B} \tag{5.15}$$

where W_B is the set of white or non-texture blocks in the bounding box B, and A_B is the area of the box. If the cost of a bounding box is larger than a predefined cost threshold, the white pixels are split into 4 subsets using k means clustering. The bounding boxes of each cluster is then obtained as a new patch. Once the mask is split into separate rectangles, the adjacent rectangles are considered for merging into a larger region using the cost function, C. Two adjacent rectangles merge into one rectangle if the cost of the combined region does not exceed the cost threshold. The results for the extracted rectangles from images in Figure 5.18 is shown in Figure 5.21.

Each rectangular patch is then processed by SIFT, as an individual image, in order to detect the keypoints. These keypoints are placed in their actual location in the original image (See Figure 5.22). The SIFT keypoints that belong to lower scales, are detected at the same location from the patches compared to the identified keypoints when using the entire image. By segmenting the image into smaller regions, we will loose the large-scale keypoints that characterize the global structure of the image.

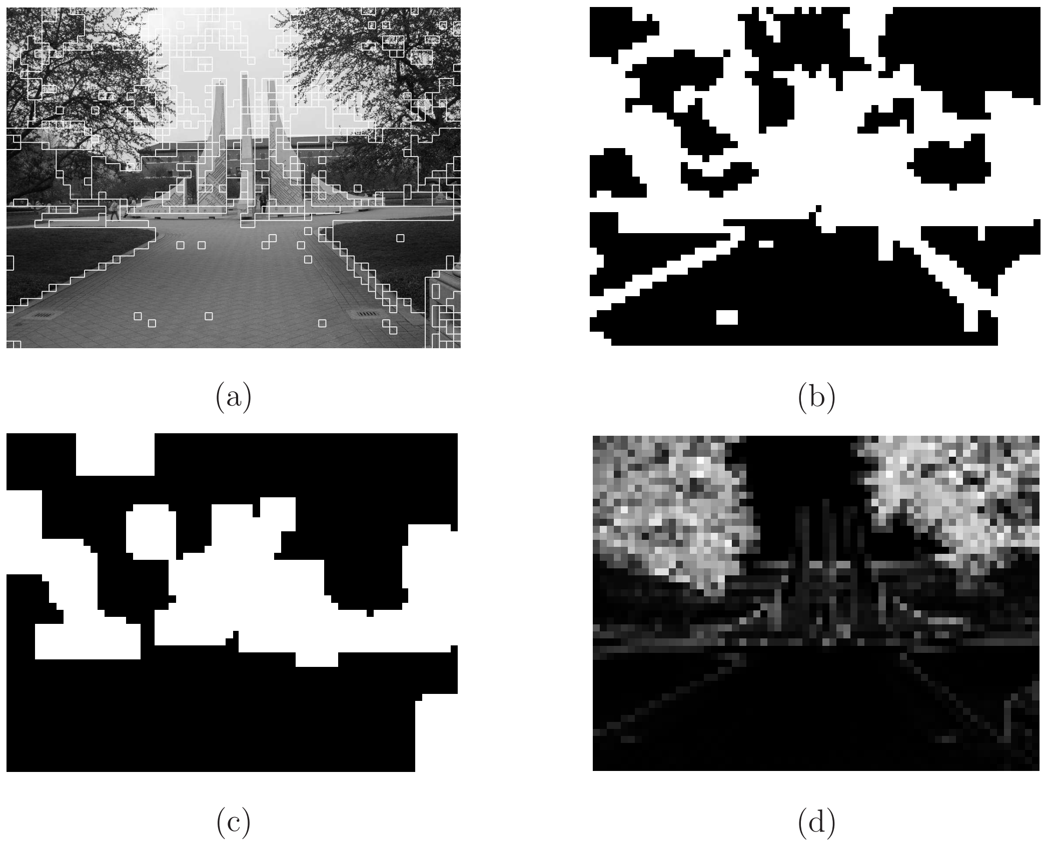


Fig. 5.20. The steps for dividing the non-texture areas into rectangular patches. (a) Results from growing the texture regions, (b) downsampled mask of the non-texture areas, (c) the mask (b) after applying morphological opening, and (d) the cost function used for split and merging the rectangles.

In order to retrieve these keypoints, we use SIFT on a downsampled version of the image. Figure 5.23 shows the results of matching using only the images downsampled with different factors. These matches are combined with the matches obtained from using the non-texture rectangular areas to complete the matching process. Figure 5.24 shows the results of the combined matched points and also the results from using SIFT on the original images. The average execution time of matching on a PC (CPU 1.66GHz, 1GB RAM), and the number of keypoints obtained by SIFT on different versions of the images in Figure 5.18 are presented in Tables 5.1 and 5.2. It

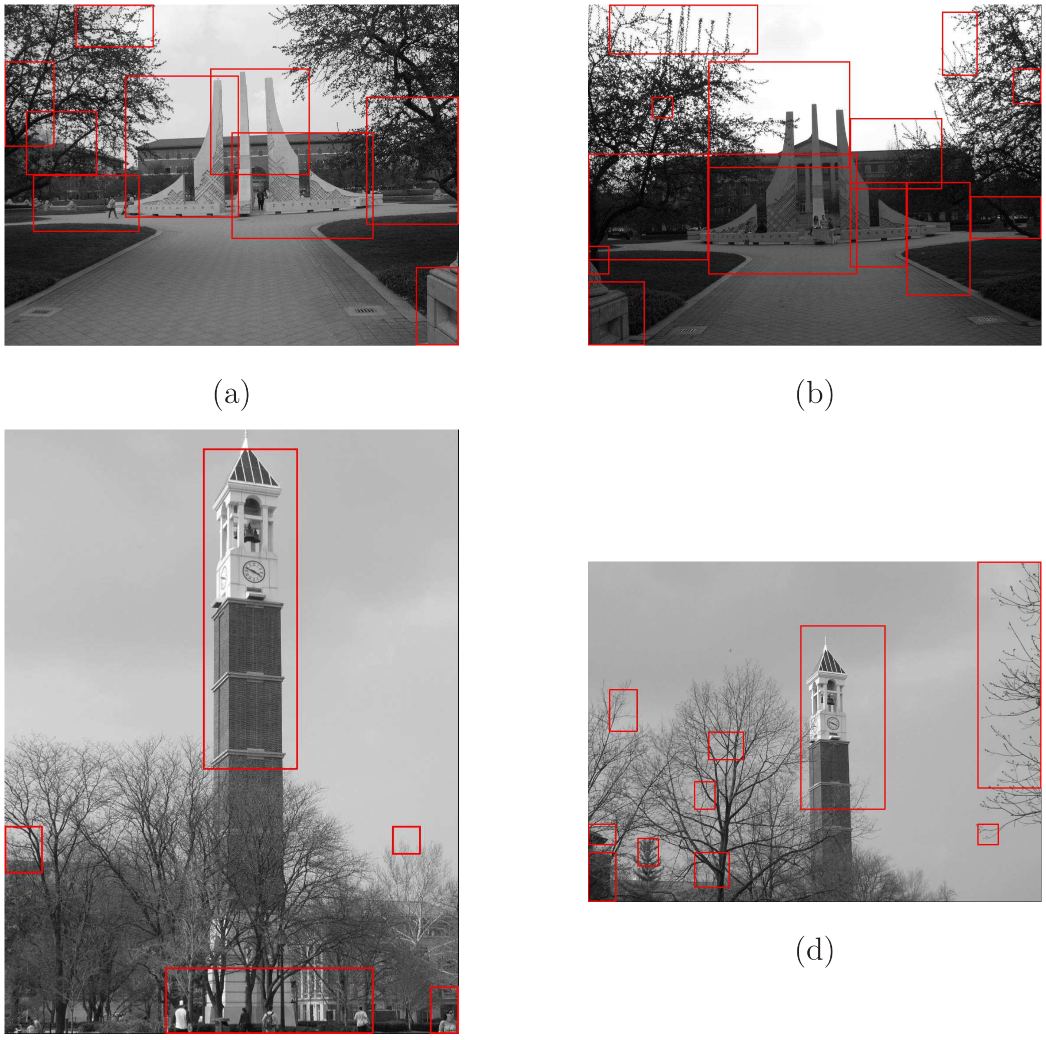

(a) (b) (c) (d)

Fig. 5.21. Examples of the non-texture patches.

is noticeable that by excluding the texture areas, the processing time for the matching algorithm decreases significantly.

Table 5.2
Statistics for the execution of SIFT on different versions (original version, extracted patches, and downsampled version) of images (c) and (d) in Figure 5.18.

Image	Original (c)	Original (d)	Non-Texture Area (c)	Non-Texture Area (d)	Downsampled by a factor of 4 (c)	Downsampled by a factor of 4 (d)
Number of Keypoints	3580	3348	1084	765	182	208
Average time of matching (s)	24.3410		6.3956		1.9649	

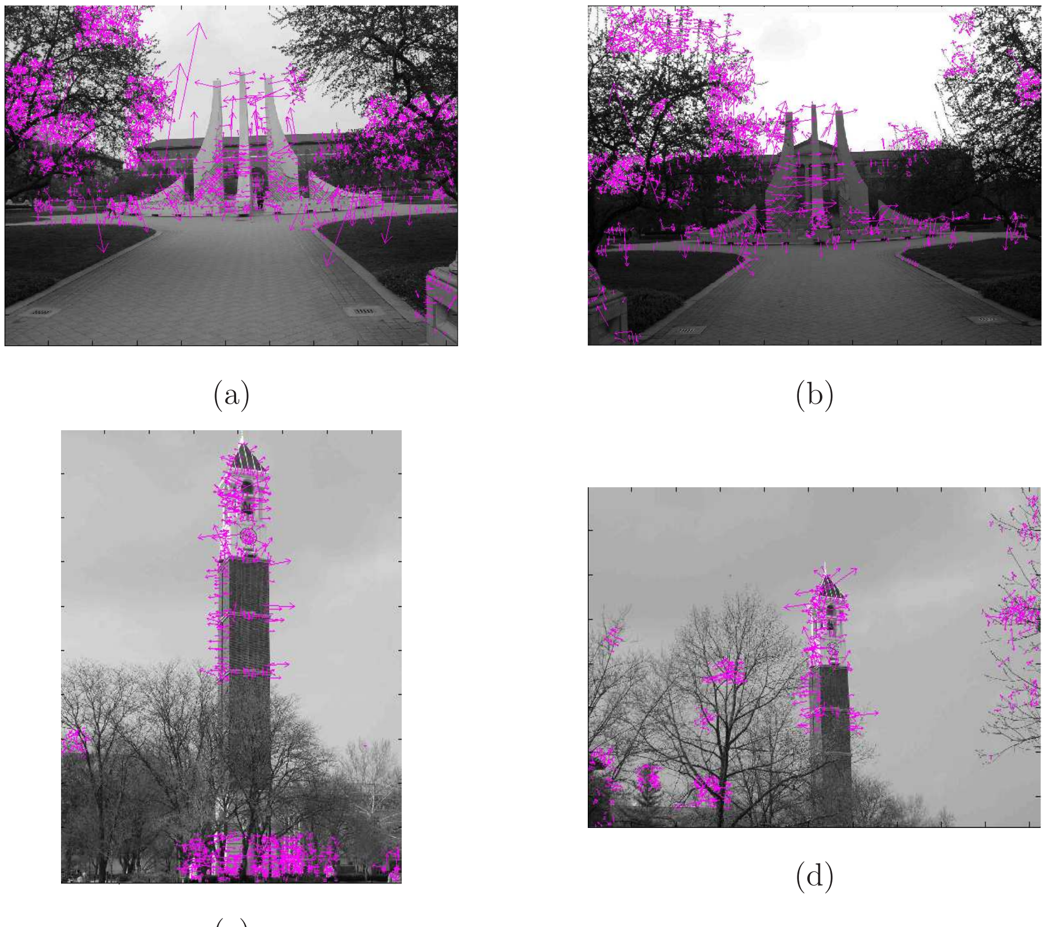
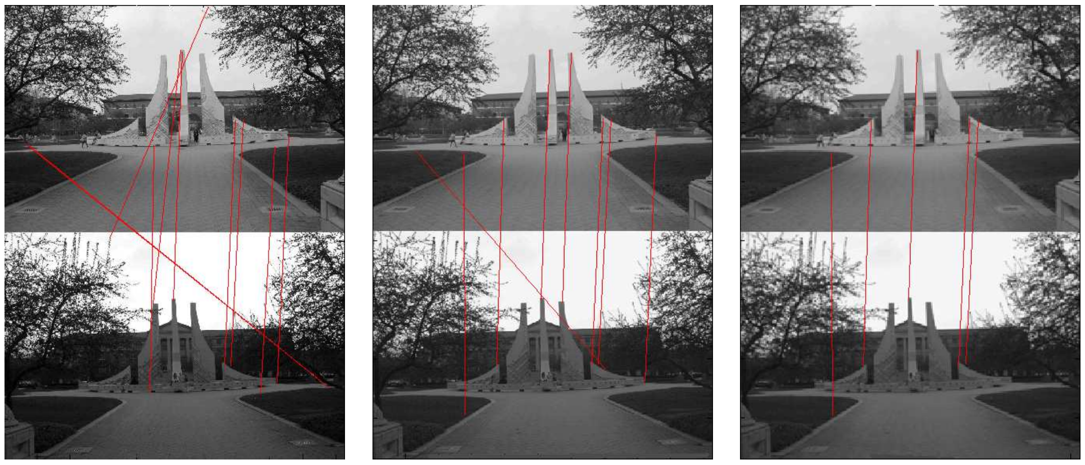
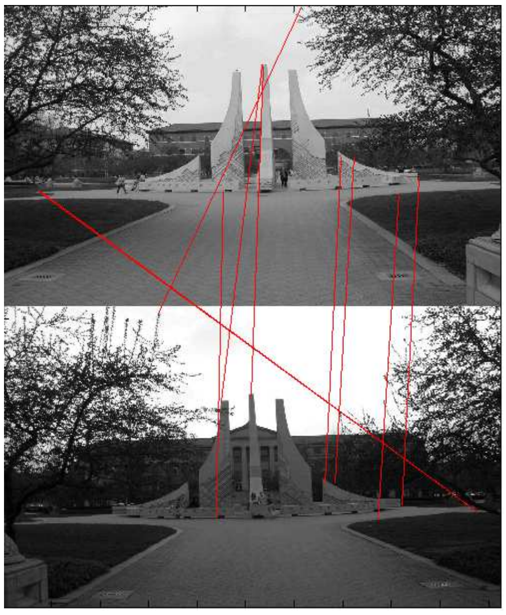


Fig. 5.22. Keypoints obtained from using the rectangular regions, overlaid on the original image.

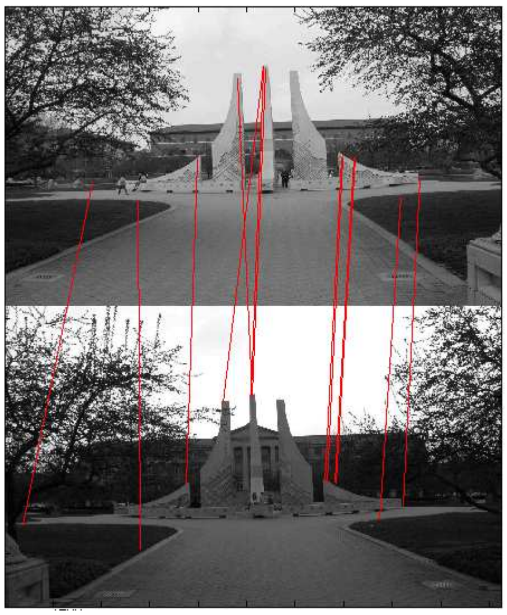


(a) (b) (c)

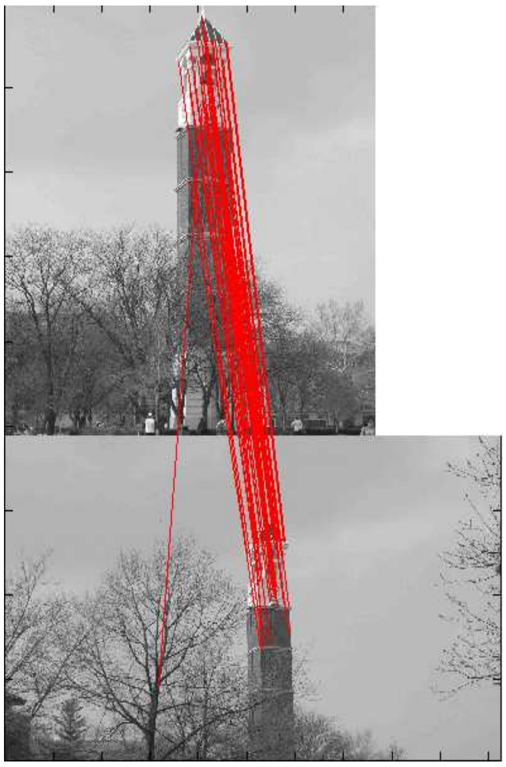
Fig. 5.23. Result of the SIFT matching method on (a) the original images, (b) downsampled images by a factor of 2, and (c) downsampled images by a factor of 4.



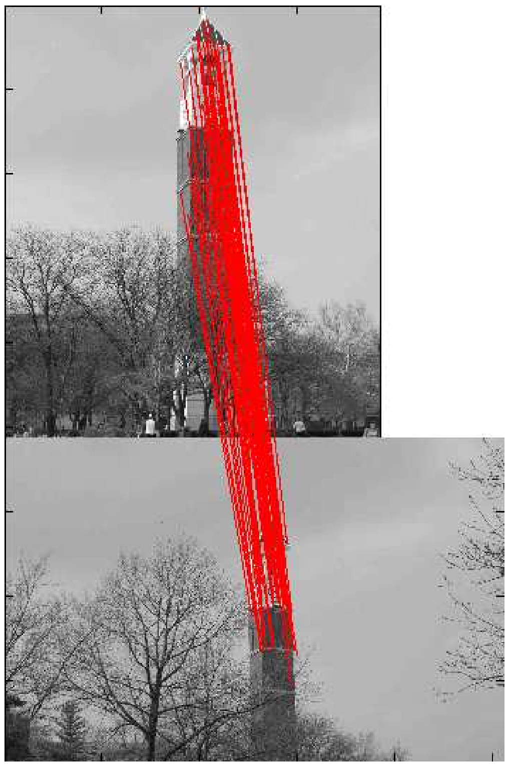
(a)



(b)



(c)



(d)

Fig. 5.24. Comparing the matching results between the original SIFT and our new method. (a) and (c) are the results of applying SIFT on the original images. (b) and (d) are the combined matches from the downsampled images and the non-texture areas.

6. SUMMARY AND FUTURE WORK

In this thesis we developed a system for video content analysis with a focus on user generated video (UGV). Compared to produced video, the special characteristics of UGV introduces uniques challenges for the content analysis. Some of these characteristics are the lack of a scene-shot-frame syntactic structure, presence of noise and abnormal camera motion (e.g. blurry and shaky motions), lack of a clear storyline and editing that is usual for produced video, and diversity in the subject. Due to these properties, many of the previously proposed methods in the literature for video content analysis, especially the ones that are developed for produced video are inadequate for UGV. For example, shot boundary detection, which has been popular for temporal segmentation of produced video, is not helpful to segment UGV into meaningful building blocks. Recent UGVs, in particular, are very different from produced video in terms of structure. These videos are often short, in the order of a few minutes or second, and include only one shot defined by the camera start and stop button. By focusing only on UGV, our system generates results that are consistent with viewers' preference as well as the camera person's interests. In this chapter we briefly summarize the main contributions of this dissertation and propose some directions for future developments.

- **General properties of the system -** Since the majority of UGVs these days are recorded, shared and watched on Mobile devices, our goal was to design our system so that it can be used with this type of devices. This factor implies two constraint on the problem; the methods used need to be computationally efficient and the results must be easy to visualize on a small-sized screen. Thus, the techniques and features used in each step of the analysis are light-weight and fast, but at the same produce good quality results that satisfy users.

Moreover, identifying ROIs in the keyframes enables the user to view only the important regions of the keyframes on their devices instead of the entire keyframe.

- **Use of camera motion -** Camera motion is a useful feature that can be extracted from a video sequence fairly easily. Several methods have been proposed in the literature to model the camera motion. This feature is intuitive for UGV content analysis since UGV often has a rich camera motion structure used by the camera person to "edit" the video while shooting it. We also conducted an eye tracking experiment on the effect of camera motion on human visual attention. This study showed that camera motion attracts viewers attention to a specific part of the frame based on the pattern of the motion. Therefore, we used this feature throughout the system. A new video segmentation method was proposed that is based on camera displacement in a video. We also proposed a new location based saliency map that is generated using camera motion parameters. This saliency map accounts for the effect of camera motion on attracting viewers' attention when ROIs in video frames are identified. Camera motion has also been used in the keyframe selection step in our system as an indicator of camera person's interests.

- **Temporal segmentation of UGV -** The new temporal concept of *camera view* was introduced that can be considered as the building block of a UGV. A video is segmented into camera views by identifying the view transitions based on the camera displacement. Each segment is devoted to one of the views that the camera has captured. Due to the fact that at least one keyframe is included in the video summary to represent each camera view, the video summary covers all the scenes viewed by the camera. Almost any other frame of the video has overlap with at least one of our keyframes. In a scene, the set of keyframes extracted by our system expands the entire video.

Another advantage of this approach for segmentation is that we can use the video keyframes instead of the entire sequence for annotation since they have the notion of all the scenes. This is particularly useful for our annotation method since we use image databases for this purpose and it is easier to compare individual frames with the images in the database.

- **User Studies -** In addition to the eye tracking experiment to study the effect of camera motion on the human visual system, we conducted several other user studies to evaluate the summarization and ROI detection results. We carefully designed and conducted the experiments so that the results were conclusive and statistically significant.

- **UGV annotation -** Due to the diversity in the content of UGV, video annotation methods that need supervised learning and are limited to a predefined number of concepts do not have the sufficient vocabulary to tag this type of video. Therefore, our method uses image databases that contain images uploaded and tagged by several users for this purpose. This leads to a rich and diverse collection of tags that can be used. In the process of annotation, we exploited the geo-location information of video to first gather images taken in the same location as the video from an image database. Two image matching approaches were then examined in order to find the most visually similar images within the collected images to the video keyframes. Tags from the final set of images were ranked based on their frequencies and the most dominant tags were associated to the video.

6.1 Future Work

The proposed systems can be improved in different aspects. Some of them are listed below.

- **Evaluation of the video annotation method -** The annotation method is based on finding tags that have higher probability of being related to video among a set of available tags in the database. Therefore, it is normal to have irrelevant tags in the results. However, validation of the annotation results is not straightforward. For example, if the system selects the tag "travel" for videos taken in a tourist location, it might be correct for many videos while incorrect for others. The reason why "travel" appears in the results is because most people who took pictures at that particular location and uploaded them on the photo sharing website were tourists who traveled to that location. If the video is also taken by a traveler, this tag is relevant to the video. However, it is also possible that this may not be the case. Therefore, an proper evaluation approach is needed that can validate annotation methods proposed for UGV.

- **Post-processing of tags and refining the results -** The tag processing step can be improved by further processing the collected tags from the database. For example, the collected tags can be classified into different categories, e.g. time-related, weather-related, object and people. The results can then be refined by choosing the feature and similarity measures that are appropriate for that particular class of tags. For instance, for verifying whether the tags, "snow," "sunset" and "flowers," are relevant to a keyframe, color and texture are useful features. But if we are to decide whether the frame belongs to the front view of Purdue Memorial Union building, one has to match the keypoints between available images of this building to the ones in the frame.

- **Local descriptors -** Although SIFT is a powerful method that has outperformed a number of other proposed approaches in the literature for image matching, there is still room for improving the descriptors and the matching process.

  One problem that we mentioned in Section 5.4.1 about SIFT features, which can be true about any local descriptor, is that self similar or repetitive patterns in the image can be misleading in the process of matching. This is due to the fact that these descriptors are generated locally and matched independently, without considering the global structure of the objects. Examples of two similar images containing self similar patterns are shown in Figure 6.1. In both examples of this figure there are several windows of the same shape in the buildings. Thus, in the matching process, a keypoint from one window may be matched to a keypoint of a similar window but not the "correct" one. This error can not be considered a regular matching error.

One way to address this problem is to obtain the mutual information between the keypoints in the same image or "intra matching." The keypoints in an image can be clustered in order to find the location of similar patterns in the image. The inter matching pairs that fall into one of these identified patterns can be corrected based on the global structure of the object and the motion between the two images.

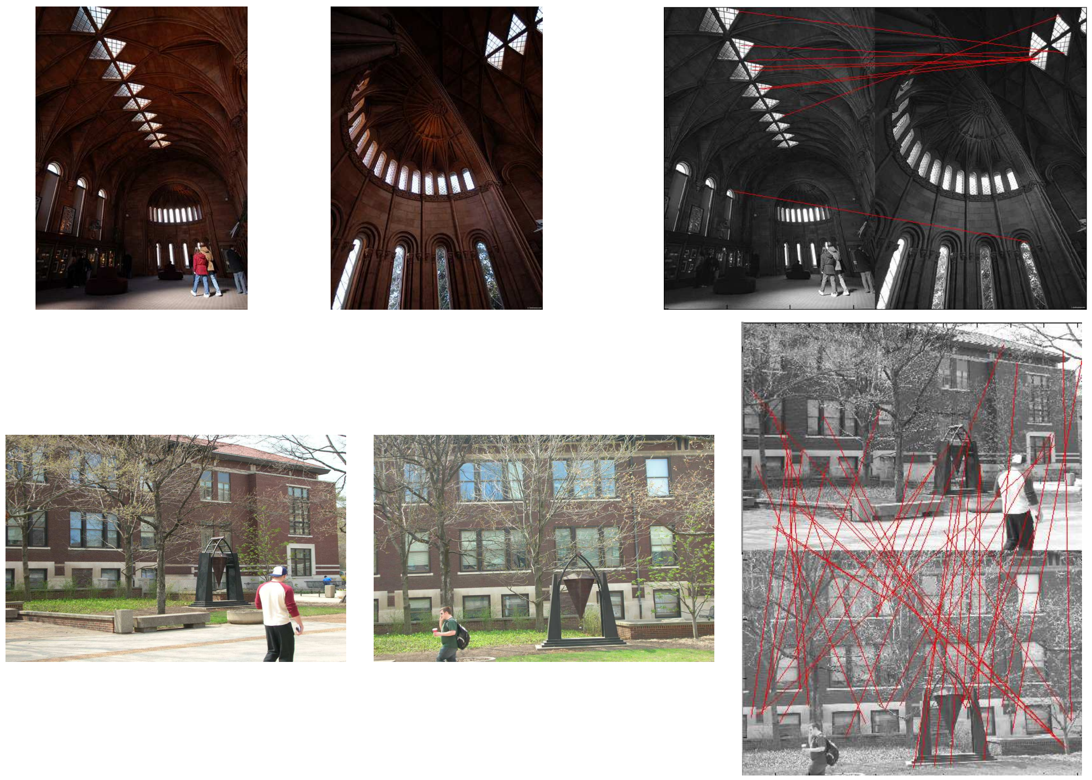


Fig. 6.1. Examples where SIFT fails due to the presence of repetitive patterns.

LIST OF REFERENCES

LIST OF REFERENCES

[1] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon, “I tube, you tube, everybody tubes: analyzing the world’s largest user generated content video system,” *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, October 2007, pp. 1–14.

[2] C.-W. Ngo, T.-C. Pong, and H.-J. Zhang, “Recent advances in content-based video analysis,” *International Journal of Image and Graphics*, vol. 1, no. 3, pp. 445–468, 2001.

[3] C. Taskiran, J. Chen, A. Albiol, L. Torres, , C. Bouman, and E. Delp, “ViBE: a compressed video database structured for active browsing and search,” *IEEE Transactions on Mutimedia*, vol. 6, no. 1, pp. 103–118, February 2004.

[4] B. Yeo and B. Liu, “Rapid scene analysis on compressed video,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 533–544, December 1995.

[5] P. Wu, “A semi-automatic approach to detect highlights for home video annotation,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, May 2004, pp. 957–960.

[6] D. Gatica-Perez, A. Loui, and M. T. Sun, “Finding structure in home videos by probabilistic hierarchical clustering,” *IEEE Transactions on Circuit and System for Video Technology*, vol. 13, no. 6, pp. 539–548, June 2003.

[7] M. Yeung, B. L. Yeo, and B. Liu, “Extracting story units from long programs for video browsing and navigation,” *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, June 1996, pp. 296–305.

[8] R. Dony, J. Mateer, and J. Robinson, “Techniques for automated reverse storyboarding,” *IEE Proceedings of Vision, Image and Signal Processing*, vol. 152, no. 4, August 2005, pp. 425–436.

[9] A. F. Smeaton, P. Over, and W. Kraaij, “Evaluation campaigns and trecvid,” *MIR ’06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, 2006, pp. 321–330.

[10] C. G. Snoek and M. Worring, “Multimodal video indexing: A review of the state-of-the-art,” *Multimedia Tools and Applications*, vol. 25, no. 1, pp. 5–35, January 3005.

[11] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma, “Annosearch: Image auto-annotation by search,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1483–1490.

[12] E. Moxley, T. Mei, X.-S. Hua, W.-Y. Ma, and B. Manjunath, "Automatic video annotation through search and mining," *Proceedings of the IEEE Conference on Multimedia and Expo*, April 2008, pp. 685–688.

[13] C. Taskiran, "Automatic methods for content-based access and summarization of video sequences," Ph.D. dissertation, Purdue University, December 2004.

[14] C. Taskiran, Z. Pizlo, A. Amir, D. Ponceleon, and E. Delp, "Automated video program summarization using speech transcripts," *IEEE Transactions on Mutimedia*, vol. 8, no. 4, pp. 775–791, August 2006.

[15] S.-H. Huang, Q.-J. Wu, K. y. Chang, H.-C. Lin, S.-H. Lai, W.-H. Wang, Y.-S. Tsai, C.-L. Chen, and G.-R. Chen, "Intelligent home video management system," *Proceedings of the 3rd International Conference on Information Technology, Research and Education (ITRE 2005)*, June 2005, pp. 176–180.

[16] M. Yeung and B. Liu, "Efficient matching and clustering of video shots," *Proceedings of the IEEE International Conference on Image Processing*, vol. 1, October 1995, pp. 338–341.

[17] J.-G. Kim, H. S. Chang, J. Kim, and H.-M. Kim, "Efficient camera motion characterization for MPEG video indexing," *Proceedings of the IEEE International Conference on Multimedia and Expo*, vol. 2, 2000, pp. 1171–1174.

[18] R. Lienhart, S. Pfeiffer, and W. Effelsberg, "Scene determination based on video and audio features," *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, June 1999, pp. 685–690.

[19] A. Hanjalic and H. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1280–1289, December 1999.

[20] P. Over, A. Smeaton, and G. Awad, *TRECVID Rushes Summarization Workshop*, 2008. `http://www-nlpir.nist.gov/projects/tvpubs/tv8.slides/tvs08.slides.pdf`

[21] H. Sawhney, S. Ayer, and M. Gorkani, "Model-based 2d and 3d dominant motion estimation for mosaicking and video representation," *Proceedings of the Fifth international conference on Computer Vision*, Los Alamitos, CA, June 1995, pp. 583–590.

[22] D. Gatica-Perez and M.-T. Sun, "Linking objects in videos by importance sampling," *Proceedings of IEEE International Conference on Multimedia and Expo*, vol. 2, 2002, pp. 525–528.

[23] A. Albiol, L. Torres, and E. Delp, "The indexing of persons in news sequences using audio-visual data," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Hong Kong, April 2003, pp. 137–140.

[24] J. Choi and D. Jeong, "Story board construction using segmentation of MPEG encoded news video," *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems*, vol. 2, 2000, pp. 758–761.

[25] Y.-P. Tan, D. Saur, S. Kulkami, and P. Ramadge, “Rapid estimation of camera motion from compressed video with application to video annotation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 1, pp. 133–146, February 2000.

[26] Y. Rui, A. Gupta, and A. Acero, “Automatically extracting highlights for TV baseball programs,” *Proceedings of the 8th ACM international conference on Multimedia*, 2000, pp. 105–115.

[27] N. Babaguchi, Y. Kawai, Y. Yasugi, and T. Kitahashi, “Linking live and replay scenes in broadcasted sports video,” *Proceedings of the 2000 ACM workshops on Multimedia*, November 2000, pp. 205–208.

[28] A. Girgensohn, J. Boreczky, P. Chiu, J. Doherty, J. Foote, G. Golovchinsky, S. Uchihashi, and L. Wilcox, “A semi-automatic approach to home video editing,” *Proceedings of the ACM Symposium on User Interface Software and Technology*, vol. 2, San Diego,CA, November 2000, pp. 81–89.

[29] L. Itti, C. Koch, and E. Niebur, “A model of saliency-based visual attention for rapid scene analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[30] Y.-F. Ma, L. Lu, H.-J. Zhang, and M. Li, “A user attention model for video summarization,” *Proceedings of The Tenth ACM International Conference on Multimedia*, 2002, pp. 533–542.

[31] O. L. Meur, D. Thoreau, P. L. Callet, and D. Barba, “A spatio-temporal model of the selective human visual attention,” *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, 2005, pp. 1188–1191.

[32] X. Xie, H. Liu, W. Ma, and H. Zhang, “Browsing large pictures under limited display sizes,” *IEEE Transactions on Multimedia*, vol. 8, no. 4, pp. 707–715, August 2006.

[33] L.-Q. Chen, X. Xie, X. Fan, W.-Y. Ma, H.-J. Zhang, and H.-Q. Zhou, “A visual attention model for adapting images on small displays,” *Multimedia Systems*, vol. 9, pp. 353–364, October 2003.

[34] W. Osberger and A. Rohaly, “Automatic detection of regions of interest in complex video sequences,” *Proceedings of the SPIE, Human Vision and Electronic Imaging VI*, vol. 4299, Bellingham, USA, 2001, pp. 361–372.

[35] Y.-F. Ma and H.-J. Zhang, “Contrast-based image attention analysis by using fuzzy growing,” *Proceedings of the eleventh ACM International Conference on Multimedia*, 2003, pp. 374–381.

[36] F. Liu and M. Gleicher, “Automatic image retargeting with fisheye-view warping,” *Proceedings of the 18th annual ACM symposium on User Interface Software and Technology*, October 2005, pp. 153–162.

[37] Y. Hu, L.-T. Chia, and D. Rajan, “Region-of-interest based image resolution adaptation for MPEG-21 digital item,” *Proceedings of ACM Multimedia*, New York, USA, October 2004, pp. 340–343.

[38] G. Abdollahian and E. J. Delp, "Analysis of unstructured video based on camera motion," *Proceedings of the SPIE International Conference on Multimedia Content Access: Algorithms and Systems*, vol. 6506, San Jose, California, January 2007, p. 65060J.

[39] J. R. Kender and B. L. Yeo, "On the structure and analysis of home videos," *Proceedings of Asian Conference on Computer Vision*, Taipei, Taiwan, January 2000.

[40] G. Abdollahian, C. Taskiran, Z. Pizlo, and E. J. Delp, "Motion driven content analysis of user generated video," *Accepted for Publication in IEEE Transactions on Multimedia.*

[41] G. Abdollahian and E. J. Delp, "User generated video annotation using geotagged image databases," *Proceedings of the IEEE International Conference on Multimedia and Expo*, July 2009, pp. 610–613.

[42] G. Abdollahian, Z. Pizlo, and E. J. Delp, "A study on the effect of camera motion on human visual attention," *Proceedings of the IEEE International Conference on Image Processing*, October 2008, pp. 693–696.

[43] G. Abdollahian and E. J. Delp, "Finding regions of interest in home videos based on camera motion," *Proceedings of the IEEE International Conference on Image Processing*, vol. 4, September 2007, pp. IV–545 – IV–548.

[44] W. Kraaij and T. Ianeva, *TREC Video Low-level Feature (Camera Motion) Task Overview*, 2005. `http://www-nlpir.nist.gov/projects/tvpubs/tv5.papers/tv5.llf.slides.final.pdf`

[45] L.-Y. Duan, M. Xu, Q. Tian, and C.-S. Xu, "Nonparametric motion model with applications to camera motion pattern classification," *Proceedings of The 12th Annual ACM International Conference on Multimedia (MULTIMEDIA '04)*, 2004, pp. 328–331.

[46] F. Coudert, J. Benois-Pineau, and D. Barba, "Dominant motion estimation and video partitioning with a 1D signal approach," *Proceedings of the SPIE Conference on Multimedia Storage and Archiving Systems III*, vol. 3527, Boston, MA, 1998, pp. 283–294.

[47] D. Lan, Y. Ma, and H. Zhang, "A novel motion-based representation for video mining," *Proceedings of IEEE International Conference on Multimedia and Expo*, Baltimore,Maryland, July 2003, pp. 469–472.

[48] K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 5, pp. 513–518, 1996.

[49] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[50] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[51] S. Wu, Y. Ma, and H. Zhang, "Video quality classification based home video segmentation," *Proceedings of IEEE International Conference on Multimedia and Expo*, July 2005, pp. 217–220.

[52] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, no. 2, pp. 4–29, April 1984.

[53] Y. Deng, M. M. C. Kenney, and B. Manjunath, "Peer group filtering and perceptual color image quantization," *IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 21–24, 1999.

[54] Y.-F. Ma and H.-J. Zhang, "A model of motion attention for video skimming," *Proceedings of the IEEE International Conference on Image Processing*, vol. 1, 2002, pp. 129–132.

[55] Pittsburgh pattern recognition, demonstration: Face detection in photographs. `http://demo.pittpatt.com/`

[56] P. K. Sahoo, D. W. Slaaf, and T. A. Albert, "Threshold selection using a minimal histogram entropy difference," *Optical Engineering*, vol. 36, no. 7, pp. 1976–1981, 1997.

[57] D. D. Salvucci and J. H. Goldberg, "Identifying fixations and saccades in eye-tracking protocols," *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications (ETRA '00)*. ACM, 2000, pp. 71–78.

[58] R. C. Sprinthall, *Basic Statistical Analysis*. Allyn & Bacon, 2002.

[59] G. E. P. Box, "A general distribution theory for a class of likelihood criteria," *Biometrika*, vol. 36, no. 3/4, pp. 317–346, December 1949.

[60] L. Itti, "The iLab neuromorphic vision C++ toolkit: Free tools for the next generation of vision algorithms," *The Neuromorphic Engineer*, vol. 1, no. 1, p. 10, 2004.

[61] C. J. V. Rijsbergen, *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 1979.

[62] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[63] B. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, December 1996.

[64] R. Haralick, K. Shanmugan, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, November 1973.

[65] C. Harris and M. Stephens, "A combined corner and edge detector," *Alvey Vision Conference*, 1988, pp. 147–151.

[66] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[67] D. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the IEEE International Conference on Computer Vision*, 1999, pp. 381–395.

[68] J. Morel and G. Yu, “ASIFT: A new framework for fully affine invariant image comparison,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, 2009.

[69] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[70] H. Bay, T. Tuytelaars, and L. V. Gool, “Surf: Speeded up robust features,” *Lecture Notes in Computer Science - ECCV 2006*, vol. 3951, pp. 404–417, 2006.

[71] Y. Ke and R. Sukthankar, “PCA-SIFT: A more distinctive representation for local image descriptors,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. 506–513.

[72] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[73] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[74] F. Zhu, K. Ng, G. Abdollahian, and E. J. Delp, “Spatial and temporal models for texture-based video coding,” *Proceedings of the SPIE International Conference on Video Communications and Image Processing (VCIP 2007)*, January 2007, p. 650806.

VITA

VITA

Golnaz Abdollahian was born in Shahrood, Iran. She received her B.S. degree in Electrical Engineering from Sharif University of Technology, Tehran, Iran in 2004. Her research interests include multimedia content analysis, image and video summarization, indexing and retrieval.

Since 2004, Golnaz has studied in the direct Ph.D. program in the area of Communications, Networking, Signal and Image Processing at Purdue University, West Lafayette, IN. In the Fall 2005, she joined the Video and Image Processing Laboratory (VIPER) as a research assistant under the supervision of Professor Edward Delp, the Charles William Harrison Distinguished Professor of Electrical and Computer Engineering. From 2006 - 2009, he was supported by SABA fellowship funded by Motorola Research Laboratories. During this time, she developed signal processing tools for video content analysis. During the summer of 2007, she was a Student Intern at the Eastman Kodak Company, Rochester, NY.

Golnaz Abdollahian is a student member of the IEEE professional society.

Golnaz Abdollahian's publications from this research work include:

Journal papers:

1. **Golnaz Abdollahian**, Cuneyt Taskiran, Zygmunt Pizlo, and Edward J. Delp, "Motion Driven Content Analysis of User Generated Video," *Accepted for publication in IEEE Transaction on Multimedia*

Conference Papers:

1. **Golnaz Abdollahian** and Edward J. Delp, User Generated Video Annotation Using Geo-Tagged Image Databases," *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, NY, July 2009
2. **Golnaz Abdollahian**, Zygmunt Pizlo and Edward J. Delp, A study on the effect of camera motion on human visual attention," *Proceedings of IEEE International Conference on Image Processing*, San Diego, CA, October 2008
3. **Golnaz Abdollahian** and Edward J. Delp, Finding regions of interest in home videos based on camera motion," *Proceedings of IEEE International Conference on Image Processing*, San Antonio, Texas, September 2007
4. **Golnaz Abdollahian** and Edward J. Delp, Analysis of unstructured video based on camera motion," *Proceedings of The SPIE International Conference on Multimedia Content Access: Algorithms and Systems*, San Jose, CA, January 2007
5. F. Zhu, K. Ng, **Golnaz Abdollahian** and Edward J. Delp, Spatial and Temporal Models for Texture-Based Video Coding," *Proceedings of the SPIE International Conference on Video Communications and Image Processing (VCIP 2007)*, San Jose, CA, January 28 - February 1, 2007