Graduate School Form 30
(Revised 08/14)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Saeid Bagheri

Entitled
TEMPORAL PROFILE SUMMARIZATION AND INDEXING FOR SURVEILLANCE VIDEOS

For the degree of Master of Science

Is approved by the final examining committee:

Jiangyu Zheng

Mihran Tuceryan

Shiaofen Fang

To the best of my knowledge and as understood by the student in the Thesis/Dissertation Agreement, Publication Delay, and Certification/Disclaimer (Graduate School Form 32), this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Jiangyu Zheng

Approved by Major Professor(s): _____

_____

Approved by: Shiaofen Fang 11/18/2014

Head of the Department Graduate Program Date

TEMPORAL PROFILE SUMMARIZATION AND INDEXING FOR SURVEILLANCE VIDEOS

A Thesis

Submitted to the Faculty

of

Purdue University

by

Saeid Bagheri

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

December 2014

Purdue University

Indianapolis, Indiana

ACKNOWLEDGMENTS

I would like to extend my most sincere gratitude towards all who have supported and helped me through this work. I especially want to thank Dr. Jiangyu Zheng for his endless academic support during my research and study, and provisions above the academic contexts. I would also like to thank Dr. Shiaofen Fang and Dr. Mihran Tuceryan for their assistance and supervision in preparation of this thesis and sharing their valuable knowledge with me.

I would also like to thank the department of Computer and Information Science for the opportunities and the support throughout my master's program. Finally, I reserve special thanks for my family, for their boundless mental, emotional, and financial support during my hard work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABSTRACT

Bagheri, Saeid M.S., Purdue University, December 2014. Temporal Profile Summarization and Indexing for Surveillance Videos. Major Professor: Jiangyu Zheng.

Surveillance videos are recorded continually and the retrieval of such videos currently still relies on human operators. Automatic retrieval has not reached a satisfactory accuracy. As an intermediate representation, this work develops multiple original temporal profiles of video to convey accurate temporal information in the video while keeping certain spatial characteristics. These are effective methods to visualizes surveillance video contents efficiently in a 2D temporal image, suitable for indexing and retrieving a large video database. We are aiming to provide a compact index that is intuitive and preserves most of the information in the video in order to avoid browsing extensive video clips frame by frame.

By considering some of the properties of static surveillance videos, we aim at accentuating the temporal dimension in our visualization. We have introduced our framework as three unique methods that visualize different aspects of a surveillance video, plus an extension to non-static surveillance videos.

In our first method "Localized Temporal Profile", by knowing that most surveillance videos are monitoring specific locations, we try to emphasize the other dimension, time, in our solution. we focus on describing all the events only in critical locations of the video. In our next method "Multi-Position Temporal Profile", we generate an all-inclusive profile that covers all the events in the video field of view. In our last method "Motion Temporal Profile" we perform in-depth analysis of scene motion and try to handle targets with non-uniform, non-translational motion in our temporal profile. We then further extend our framework by loosening the constraint that the video is static and including cameras with smooth panning motion as such

videos are widely used in practice. By performing motion analysis on the camera, we stabilize the camera to create a panorama-like effect for the video, allowing us to utilize all of the aforementioned methods. The resulting profiles allows temporal indexing to each video frame, and contains all spatial information in a continuous manner. It also shows the actions and progress of events in the temporal profile. Flexible browsing and effective manipulation of videos can be achieved using the resulting video profiles.

1 INTRODUCTION

The amount of recorded surveillance video is growing at a very fast pace. Viewing and analyzing such footage become labor-intensive and time consuming tasks as massive amounts of videos are obtained on a daily basis. Problems with storage, indexing and retrieval of such video databases arise when the data growth rate is high. Videos are also far more difficult to browse and search as compared to images due to its large data size and sequential data structure. Thus, it becomes an important topic to summarize the videos in compact files that are intuitive and preserves most of the information in the video.

Current video indexing is mainly based on key frames [1] from coarse thumbnails and fine storyboards [2] to tapestries [3] for searching and editing. The spatial mosaicing is the extension of key frames to a larger field of view covered by the video movement [4]. They can index to a clip but not a frame. For a large volume of surveillance video from static cameras, several works have removed segments without events and thus shorten the video length. Video synopsis [5], [6], [7], [8] based on spatial mosaicing methods [9], [10] compose different actions of targets at different time instances in a single key frame. However, such effort has limitations on representing temporal changes for long videos because it becomes cluttered and confusing as the video length and number of targets increases. It also fails to present the time instance of the actions, thus it will not serve very well for indexing purposes [11].

In contrast to such spatial indexing composed of multiple frames, a compact temporal profile has been sought [12], [13]. In this work, we proposed a thorough framework that summarizes a video for static surveillance cameras, providing a high resolution profile of video that preserves the temporal order of frames. We then extend our work to a create a permeating framework that is applicable to a vast majority of problems concerning surveillance videos. Such extension was realized to deal with

four major problems that are introduced in surveillance videos. We will proceed by identifying these problems, analyzing the constraints and finally introducing our solution to the problem.

1.1 An Overview of Our Framework

The amount of recorded surveillance video is growing at a tremendously fast pace. Current video retrieval still relies on human operators because automatic search has not reached a satisfactory accuracy. Viewing and analyzing such footage has become a labor-intensive and time consuming task. Captured by a static camera, surveillance videos are usually aimed at monitoring specific areas and observing critical locations where targets pass through. Hence the background is static throughout the entire video and the flow of background is along the time axis in the video volume. However, a moving object leaves a trace non-parallel to the time axis. Therefor, we are aiming to provide a compact index that is intuitive and preserves most of the information in the video in order to avoid browsing extensive video clips frame by frame. By taking into consideration that the video is static, and it is aimed at monitoring specific locations, we try to emphasize the other dimension, time, in our solution. The next three frameworks we introduced are based on these constraints and all aim at accentuating the temporal dimension in our visualization. We introduced *temporal slices* that are generated by sampling a 2D plane in the 3D volume of surveillance video. Temporal slices are an important building block of this study and will be used throughout this work as base for our solutions. Chapter 2 explains the theory behind temporal slices in detail, describes the implementation and expands on some characteristics of the temporal slice. This temporal slice, however, is subject to to shape deformation as it is a reduction from the 3D space of the video volume to a compact 2D image. In addition to that, the temporal slice alone fails to include some important information.

As mentioned earlier, the temporal slice is sampled on a single plane in the video volume. This means that we are reducing one of the spatial spaces by holding it

constant. Thus, while the temporal slice shows that the target has crossed a certain location at a certain time, it does not show the direction in which the target crossed the location (i.e. from left to right or from right to left). To solve this problem, the "Localized Temporal Profile" was introduced that samples three planes in the field of view at positions that are sufficiently far apart, but still fall into the same monitored location [14]. We blend all the slices at different locations into a single temporal profile and use transparency to indicate spatial positions of slices. The localized temporal profile is a very intuitive and efficient profile that describes part of the field of view and it has been described in fine detail in chapter 3.

Localized Temporal Profile, as the name suggests, is designed to represent parts of the video volume. This means that information in other parts of the field of view may be missed in this form of representation. This was our motivation in extending our work to a profile that would include more global information while preserving the accuracy on the temporal axis, as well as the space efficiency. In chapter 4, we will introduce the "Multi-position Temporal Profile" that intents to capture all motion information in either horizontal or vertical direction. We will then continue by investigating the information represented by this profile and how to interpret it.

Nonetheless, preserving the temporal quality of the profiles comes at a cost. The spatio-temporal sampling in the video volume will introduce some shape deformations to the shapes that the targets leave in the profiles. For instance, take the example of a person walking through a doorway. If the person smoothly, walks past the doorway, the methods introduced above will capture all the necessary motion information successfully. However, if the person decides to stay in the doorway before passing the door completely, or if he walks through with non-uniform translational motion aforementioned methods might not work. For non-translational motions such as a person dancing in a monitored area, or any arbitrarily motion such as waving of the leaves in a scene, the traces will not leave very helpful information in the temporal slice. This motivated us to introduce a profile that takes the velocity of objects into account, called "Velocity Based Temporal Profile". The motion temporal profile gives

more visual weight to objects with higher image velocity. This means that motions such as slow waving or swaying from left to right in the same position will not clutter the profile and leave room for more important information such as a person running away. The visual weights of targets are assigned by measure their image velocity and assigning a transparency value to the accordingly. Chapter 5 will start by an in-depth description of the profile and further explain the implementation process.

We then further extend our framework by loosening the constraint that the video is static and including cameras with smooth panning motion as such videos are widely used in practice. Chapter 6 describes a simple and robust method to generate a stabilized, panorama-like video using a "motion-condensed image" (1.2). All of the methods described above can then be utilized on such video, as it is most similar to a static video. All these profiles described in this framework extremely space efficient for storing large surveillance video. Furthermore, they are computationally in-expensive and can be performed on the fly on a video stream. These methods can also be used in combination to increase the robustness of presented information, while measuring significantly smaller in size than the video and with very little performance overhead.

1.2 Condensed Images

As we stated earlier, motion analysis of videos is a key element of our work and optical flow is an important characteristic that helps us analyze videos. The cost of optical flow computation for large video database is high and the results are unstable for scenes with deformation, arbitrary motion and scenes without many features. As introduced in [15], *condensed images* are an efficient and robust way to represent important motion in the video. Two condensed images were employed to reflect the motion of objects in the video in horizontal and vertical direction. The vertical condensed image is an average of intensity values along the y axis, $C(x,t)$, and the

Table 1.1.
Terminology and symbols used

Name	Symbol used	Description
Temporal Slice	$I(t, l)$	Sampled in the video volume along arbitrary line
Vertical Temporal Slice	$I_i(t, y)$	Sampled in the video volume along y axis
Horizontal Temporal Slice	$I_j(t, x)$	Sampled in the video volume along x axis
Horizontally Condensed Slice	$\bar{I_x}(t, y)$	Average pixel intensities along x axis for N columns
Vertically Condensed Slice	$\bar{I_y}(t, x)$	Average pixel intensities along y axis for N columns
Horizontally Condensed Image	$C(y, t)$	Average pixel intensities along x axis for all columns
Vertically Condensed Image	$C(x, t)$	Average pixel intensities along y axis for all rows
Temporal Profile	$P_i(t, y)$	Result of blending temporal slices together

horizontal condensed image is obtained through averaging pixel intensities along the x direction, $C(y,t)$:

$$C(x,t) = \frac{1}{Y}\sum_{y=0}^{Y} I(x,y,t)$$
$$C(y,t) = \frac{1}{X}\sum_{x=0}^{X} I(x,y,t) \tag{1.1}$$

where X and Y are the width and height of video frame respectively, as depicted in figure 1.1.

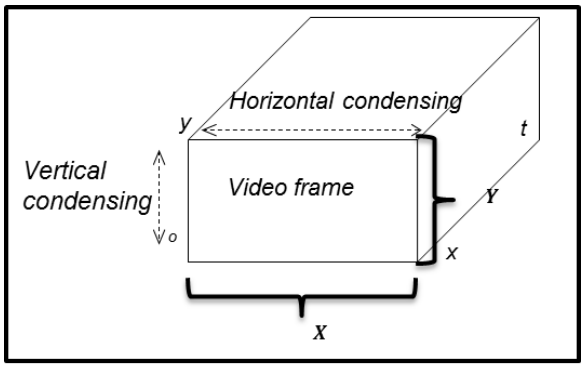
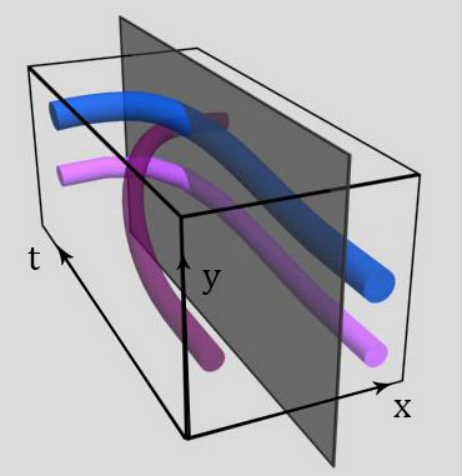


Figure 1.1. Generating condensed images in the x and y direction

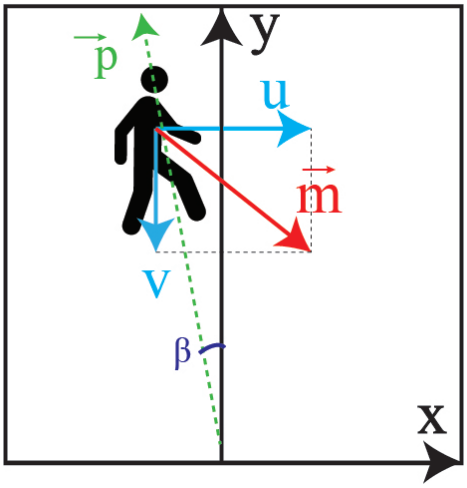
In case of static cameras, condensed images will consist of two major part. The background will appear as parallel lines stretched along the time axis and the foreground motion will appear as traces non-parallel to the time axis. In this study, we will use vertically condensed images to quantify the motion of foreground objects. In case of cameras with smooth panning motion, the vertically condensed image is used to determine the major flow in the video and determine the direction and velocity of camera motion. This information is the used to stabilize the video to prepare it for the preceding steps of generating the temporal profile.

2 TEMPORAL SLICE

Captured by a static camera, surveillance videos are usually aimed at monitoring specific areas and observing critical locations where targets pass through. Hence the background is static throughout the entire video and the flow of background is along the time axis in the video volume. However, a moving object leaves a trace non-parallel to the time axis as illustrated in figure 2.1(a).



(a) An illustration of video volume with foreground flow (colored tubes). The plane intersecting the volume is the temporal slice obtained from sampling a pixel line continuously over time.



(b) A foreground object passing the sampling line with the moving direction.

Figure 2.1. Sampling foreground motion with a line.

2.1 Implementation

If we set a line in the video frame and sample the pixel data on it over consecutive frames, obtaining a temporal slice, I(t,y), in the volume. If the line orientation is set non-parallel to the motion direction of foreground flow in the image, the foreground will leave some shapes in the temporal slice, otherwise known as flow traces. As a real example, figure 2.2 shows several frames and an obtained temporal slice in the video volume. This gives the first criterion to set the sampling line.

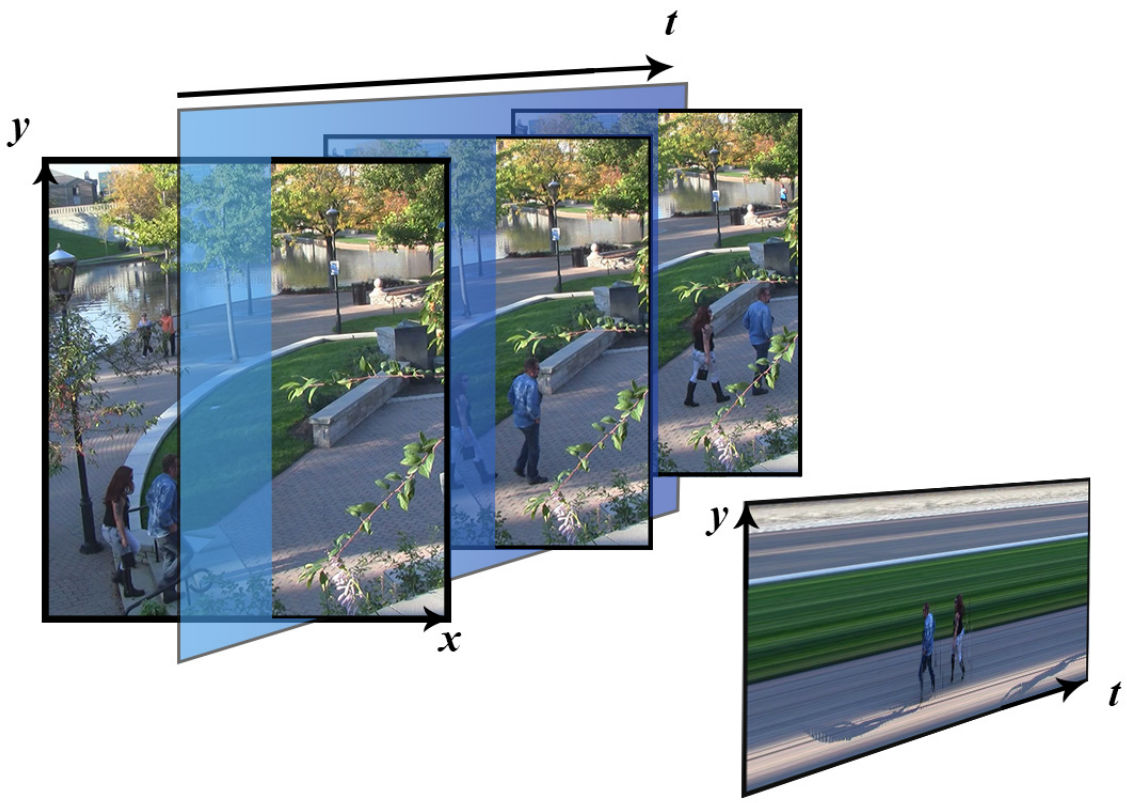


Figure 2.2. Cutting a temporal slice along the time axis at a location in the video with apparent motion. As you can see shape and motion information is revealed in the temporal slice *I(t,y)*

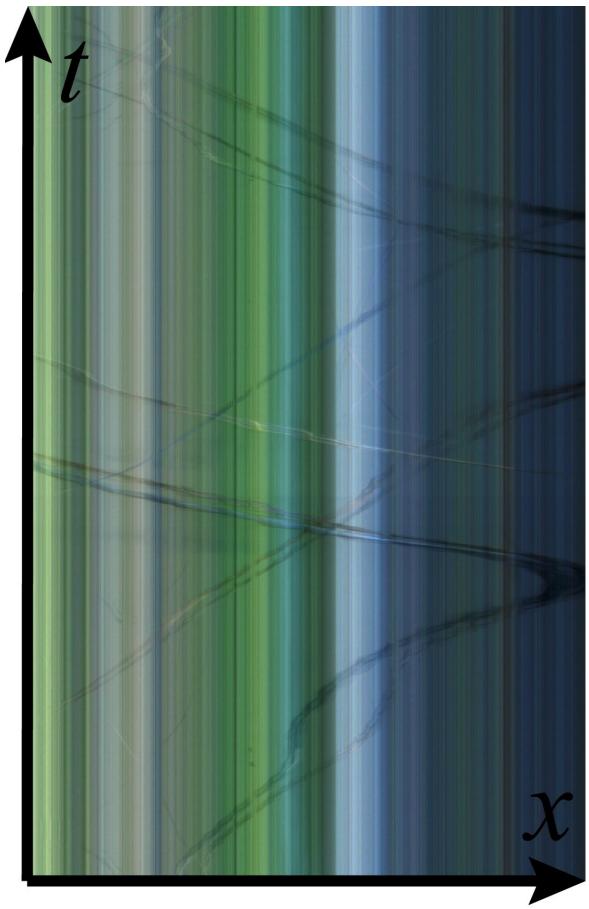


Figure 2.3. A condensed image of video obtained from averaging the pixel values in y direction in the volume to observe the flow. The traces of people are visible in the condensed image.

Criterion 1*: The fixed sampling line to obtain a temporal slice in the video should not be set parallel to the flow direction.*

Now, what direction is better to fix the sampling line in the frame after excluding the flow direction? Assume a path exists in the camera view that causes a major foreground flow with a certain variation for individual targets as in figure 2.1(b). The path direction for targets to path is **m**=(u, v) possible to be considered as speed vector. We also consider the basic poses of targets, **p**, in their up-right direction in the image for improving the generated shapes.

Criterion 2*: A sampling line is set to cross the foreground flow either in vertical or in horizontal direction so that the line is more orthogonal to the major flow and more aligned with the normal target pose.*

Although a slanted sampling line is possible to be set orthogonal to the major flow in the view, we select a horizontal or vertical line here to maintain the quality and Sampling cost of temporal slice. A coordinate system O-xy is set on the sampling line. The flow vector passing the line for an individual target is denoted as vector in figure 2.1(b), where u is the component orthogonal to the line (in x direction) for revealing shapes, and v is the component parallel to the line that leaves skewed shape in the slice. A condensed image of video [15] accumulated along y direction indicates the global motion trajectories in the video. In Figure 2.3, we can observe traces of background parallel to the time axis and the foreground flow moving in different directions.

We sample the pixels on the selected line at each frame to obtain an array of pixels, and the arrays from consecutive frames are connected along time axis. This results in a *temporal slice* in the spatial-temporal volume as shown in figure 2.3. The slice thus shows very accurate temporal information, and is able to preserve certain characteristics of target shape and environment as shown in figure 2.4. From the slice, we can index to a frame t at the precision of 1/60 second, if the interlace format is used in the sampling.

This is much more accurate than the indexed resolution of a clip by mosaicing [5], [6], [7], [8] or tapestry [3]. Sometimes slicing at an obscured location can even reveal acute and deliberate details in the video, often unnoticed by the human. In figure 2.5, the visual attention may not notice an object (marked in green) sneaking past a certain point, while it appears clearly in the temporal slice.

Similar to the aforementioned method, we can sample pixels on a line parallel to the x axis to construct a horizontal temporal slice. Figure 2.6 shows an example where a horizontal temporal slice reveal the vertical motion of objects in a video.

Figure 2.4. Original resolution of temporal slice *I(t,y)* where people are detailed enough for identification. It is compatible to the original resolution in the frame in the y direction.

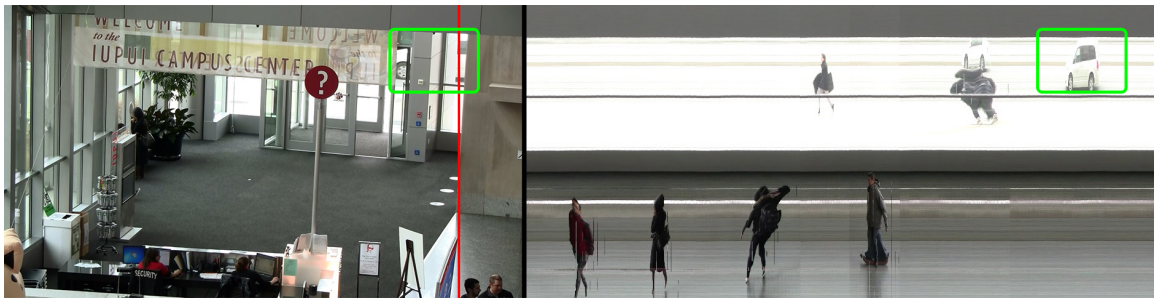


Figure 2.5. A temporal slice that captures a passing car in complete shape (right), which is unnoticeable as a complete shape in the video frame (left). The red line is sampled for the temporal slice.

2.2 Characteristics of Temporal Slice

Subject to certain shape deformations, the temporal slice reveals all the passing targets in the video that pass through the sampled space.In case of uniform, translational motion, these deformations are negligible and the shapes revealed in the temporal slice can be used for retrieval purposes. It is worth mentioning that the deformation also reflects some valuable information about the foreground targets.

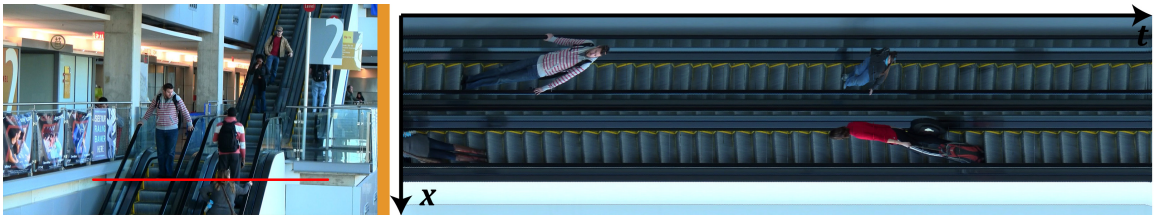


Figure 2.6. A horizontal line samples apparent vertical motion to obtain a temporal slice. The targets are skewed due to slanted motion vector with respect to the sampling line (non-zero v component). The high image quality allows capturing deliberate details.

- The length of a target in the temporal slice is related to its image velocity, i.e., its length is inversely proportional to u at the sampling line. If the target crosses the sampling line at a high velocity, its shape appears narrow in the slice. On the contrary, if its velocity is low, its shape appears stretched in the slice.

- If the slanted axis or pose **p** of a target passes the sampling line with an angle β in the image, the projected shape is skewed along the t direction in the slice. This may happen when the camera overlooks a site from a high position under perspective projection.

- The v component of image flow extends the height of the targets in the temporal slice, which can be restored by an inverse-skew operation after detecting the dynamic targets, as will be described later in chapter 3.

- All targets face the same direction in a single temporal slice; it lacks the spatial location and moving direction of foreground, because targets are scanned at their front when they move forward. This can be improved by multi-line sampling in the following.

2.3 Skewed Traces in the Temporal Slice

When a frame is covered by denser sampling lines, it is possible that a target has both vertical and horizontal motions at a sampling location. If velocity component v parallel to the sampling line is large, its shape will be stretched along the spatial axis in the temporal slice. Hence, the traces left in the temporal slice will have a skew distortion factor. If the passing velocity component $u \neq 0$, we have an opportunity to skew the foreground target in y direction to improve the shape. Before blending slices into the temporal profile, we skew the bounding boxes enclosing the targets in the temporal slice according to the direction of $\mathbf{m}$. Figure 2.7 shows such skewed results, where local shapes of targets are improved and the moving direction in the temporal profile (moving up or down in the view) is preserved. The condition $u \neq 0$ can be examined as traces in the condensed image are non-parallel to the time axis. If those targets have u=0, additional horizontal sampling lines can be set to obtain another temporal profiles (e.g., figure 2.6).

Figure 2.7. Inverse-skew operation for shape improvement in the temporal profile. (left) An indoor area where passing flows (yellow) are very slanted in the camera view. (middle) Original temporal profile. (right) Inverse-skewed profile in dynamic foreground regions.

3 LOCALIZED TEMPORAL PROFILE

Because the space a surveillance camera focuses on is fixed, we focus on representing the precise time progress of dynamic events in the video happening at critical positions. We introduce an original method that will create a temporally accurate profile of a video in a 2D image while preserving some shape and spatial information. The most important information in a video recorded with a static camera is the foreground motion. Therefore, we develop a technique to present the motion intuitively and preserve the temporal context in video. Previous research by Zheng and Sinha [16] has realized a line sensor to capture dynamic targets through a monitored line for temporal video profiling. However, the spatial information is lost when a temporal slice is cut from the spatial-temporal volume of video. To solve this problem, this work samples multiple lines in the field of view at critical positions with major motion flow in the video to capture major motion in the entire volume. In addition, we blend all the slices at different locations into a single temporal profile and use transparency to indicate spatial positions of slices. A diagram of the approach is shown in figure 3.1. The resulting profile is easy for measuring the time instance and duration of actions. Multiple targets can be compared in the time domain as well. Some other temporal methods that visualize brief summaries of video along the time axis [2], [3], the time instance of actions and events are not accurate.

3.1 Foreground Extraction in Temporal Slice

In order to overcome the problem of a temporal slice lacking spatial information, we set multiple parallel slices at critical locations in the video to sample dynamic events. If these lines are spatially apart from each other with the distances in between wider than target widths, a target will not pass them simultaneously. Thus the

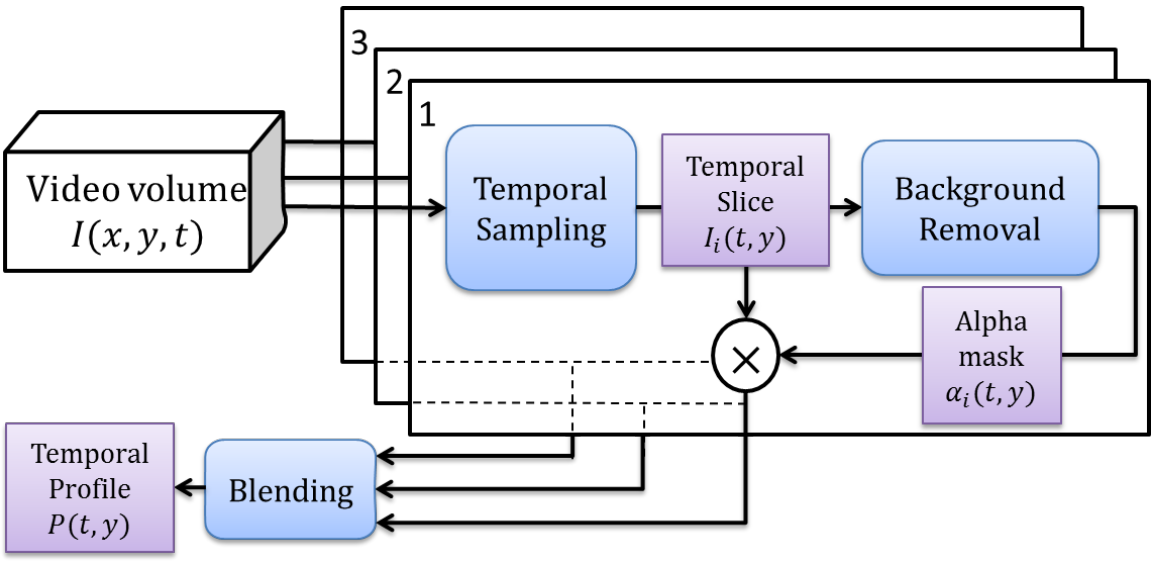


Figure 3.1. Framework of generating temporal profile from video.

foreground flow crossing these slices will have delays, and the shapes appearing in the temporal slices will not overlap exactly. This temporal order in the slices helps to determine the motion direction of target. Therefore, we blend these slices together according to their spatial locations to create a *temporal profile* of video that shows the dynamic flow of foreground clearly.

In each temporal slice, background is visible as parallel stripes along time axis as depicted in figure 2.2, and it will occlude other slices in the blending. Analysis of the temporal slice alone can yield sufficient information for background removal, which deals a smaller data set than analyzing the video volume [17], [18], and thus we perform a series of steps to remove the background in each slice, as depicted in the diagram in figure 3.2.

For temporal slice $I_i(t, y)$ sampled at position i, we define the temporal derivative of the slice as:

$$I_i'(t, y) = \frac{\partial I_i(t, y)}{\partial t} \tag{3.1}$$

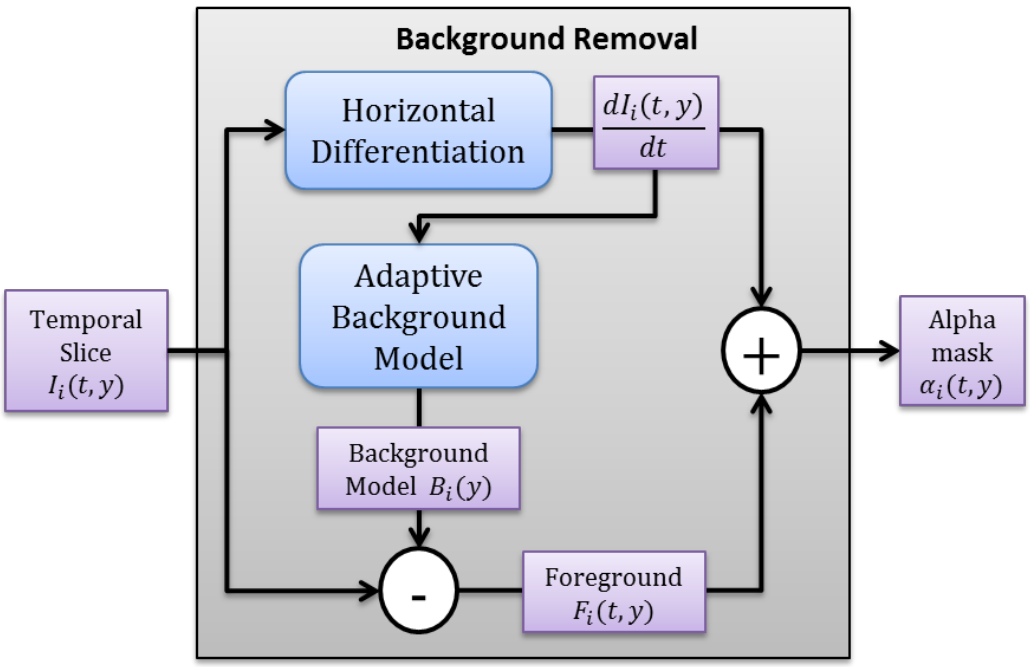


Figure 3.2. Foreground detection from temporal differentiation and background subtraction in each temporal slice.

which is implemented by Sobel operator and the output is mapped to range [0, 1]. Since each background pixel leaves a uniform trace parallel to the time axis, it will be removed after temporal differentiation with a threshold τ_1. The foreground may include some slow movements such as the waving of leaves, flag, and water as dynamic scenes.

We use this differential image to produce and update a background map, $B_i(y)$, which will be further used for foreground extraction. For time instances $t_j (j < N)$ without foreground activities, i.e., $|I_i'(t_i, y)| < \tau_1$, the background map is estimated as the average of those columns.

$$B_i(y) = \frac{1}{N} \sum_{j=0}^{N} I_i(t_j, y) \tag{3.2}$$

We then subtract the temporal slice from this background array to get foreground regions; the resulting value of subtraction is thresholded by a threshold value τ_2.

$$F_i(y) = \begin{cases} 0, & |B_i(y) - I_i(t,y)| < \tau_2 \\ 1, & otherwise \end{cases} \tag{3.3}$$

This output is further combined with the result from equation 3.1 to generate a mask of dynamic foreground as the maximum of background subtraction and temporal differentiation.

$$mask_i(y) = \max \left\{ F_i(t,y), \frac{dI_i(t,y)}{dt} \right\} \tag{3.4}$$

It is worth mentioning that the differences are all performed in three color channels, and the result is converted to an 8-bit gray scale image. Figure 3.3 is an example of detected foreground from temporally sampled slice. The resulting masks are used for blending multiple slices.

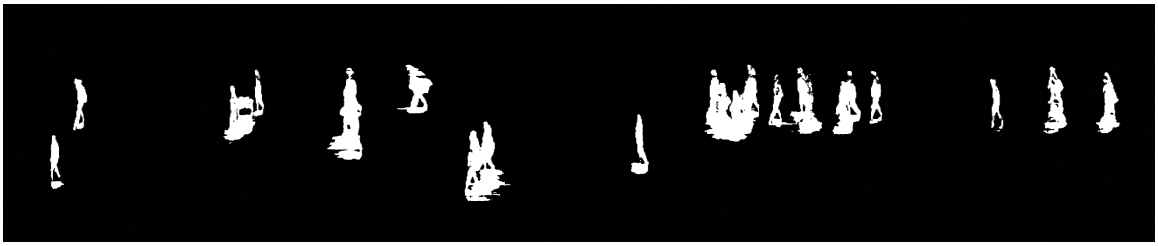


Figure 3.3. Mask generated for foreground objects from temporal slice.

3.2 Integrating Slices to Temporal Profile of Video

At a critical location, a temporal profile is integrated from three different temporal slices sampled on parallel planes. We blend slices with different transparencies according to their spatial locations in the video frame. Denote $I_0(t,y)$, $I_1(t,y)$ and $I_2(t,y)$ as the slices on locations from left to right in the video frame respectively, each slice has a blending coefficient α_i that determines its contribution to the final

temporal profile. In general, assume the video is sampled at n different locations, where $n \geq 3$, e.g., as depicted in figure 3.4, we blend slices as follows:

$$\begin{aligned} P_i(t,y) &= [1 - \alpha_i mask_i(t,y)]P_{i-1}(t,y) + \alpha_i mask_i(t,y)I_i(t,y) \\ P_0(t,y) &= I_0(t,y), \qquad\qquad i = 1, ..., n \end{aligned} \tag{3.5}$$

Figure 3.4. Temporal profile of a video recording an indoor environment by sampling three slices (red lines) with the transparency of slices increasing from left to right. The arrows on top of targets show the motion directions of passing people in left or right direction.

where P_0 is the slice at location 0. It contains background to provide the profile with a context and it is not influenced by background removal. If the value of $mask_i$ at a position is zero, the color value in P_{i-1} is used. As long as the slices are blended in such spatial order, the motion direction will be clear in the resulting profile. An example is shown in figure 3.4, where α_1 and α_2 for slices $I_1(t,y)$ and $I_2(t,y)$ are set as 0.7 and 0.5, respectively. Therefore, one can observe that, if a target moves to the right in the frame, the shapes will be more and more transparent in the profile. Inversely, a leftward motion generates shapes more and more opaque in the profile. This solves the moving direction problem in a single profile.

4 MULTI-POSITION TEMPORAL PROFILE

The proposed idea in this work is to watch the video volume sideways in the time domain (figure 4.1), which yields a 2D temporal profile with one axis as the time and the other as an axis in the space. For static camera field of view (FOV), we create this continuous visual index for precise time progress of dynamic events at all positions, while preserving shape for target identification. Also, the temporal profile includes an embedded background to provide space context. Thus, it can present motion targets intuitively in the time domain with the spatial information embedded.

To reflect target motions in the entire view as complete as possible, we consider sampling all positions in the frame to generate the temporal profile. If the sampling lines are vertical, they will capture the horizontal motions, while if we choose the sampling lines horizontally, the vertical motion in the video will appear in the profile. The sampling locations are chosen at evenly distributed locations, and the slices are integrated to show a variety of motion in the entire video. Although the interval of sampling lines can be arbitrary small, the integrated profile may become cluttered. We then blend all the slices into a single temporal profile and adjust the slice transparencies according to their image positions to show a *haze* effect. For the static background and less dynamic targets, another slice is further cut diagonally across background flow in the volume to reflect the background in the profile and provide spatial context. In addition, a strategy is designed to align a sampling line with a principal pose axis of passing object to improve the target shapes in the resulting temporal profile.

As depicted in figures 4.1 and 4.2, for a video volume *I(x,y,t)* a temporal slice i is a 2D image $I_i(t, l)$, such that for any pixel $p \in I_i(t, l)$, we have $p \in I(ax, by, t)$. The t axis indicates the time and the other axis l is a linear combination of x and y. Time stamp t provides the precise frame number of a target for further investigation

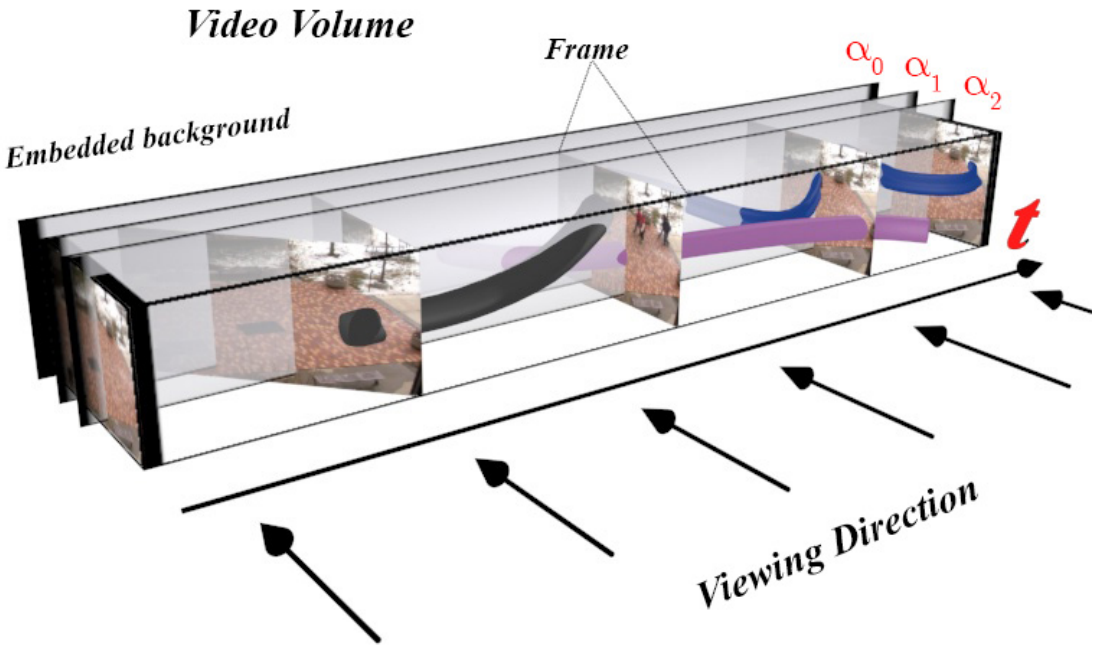


Figure 4.1. Observing a video volume from the side for precise time profile. Three temporal slices are cut and blended together with transparency values α_0, α_1 and α_2. A background slice is also cut (diagonally) to provide spatial context.

around after its brief identification in the profile; the profile thus serves as a video index. The mapping from the volume to the 2D profile reduces the data dimension but keeps the time information. Our goal is to expand spatial probing across the field of view in order to include more targets in the temporal profile. In addition to the time, shape and location should be preserved for target identification and action understanding.

Assuming targets such as humans and vehicles have an obvious translation direction $\mathbf{M}$ on the 3D world, then there is a principal pose direction $\mathbf{H}$ associated

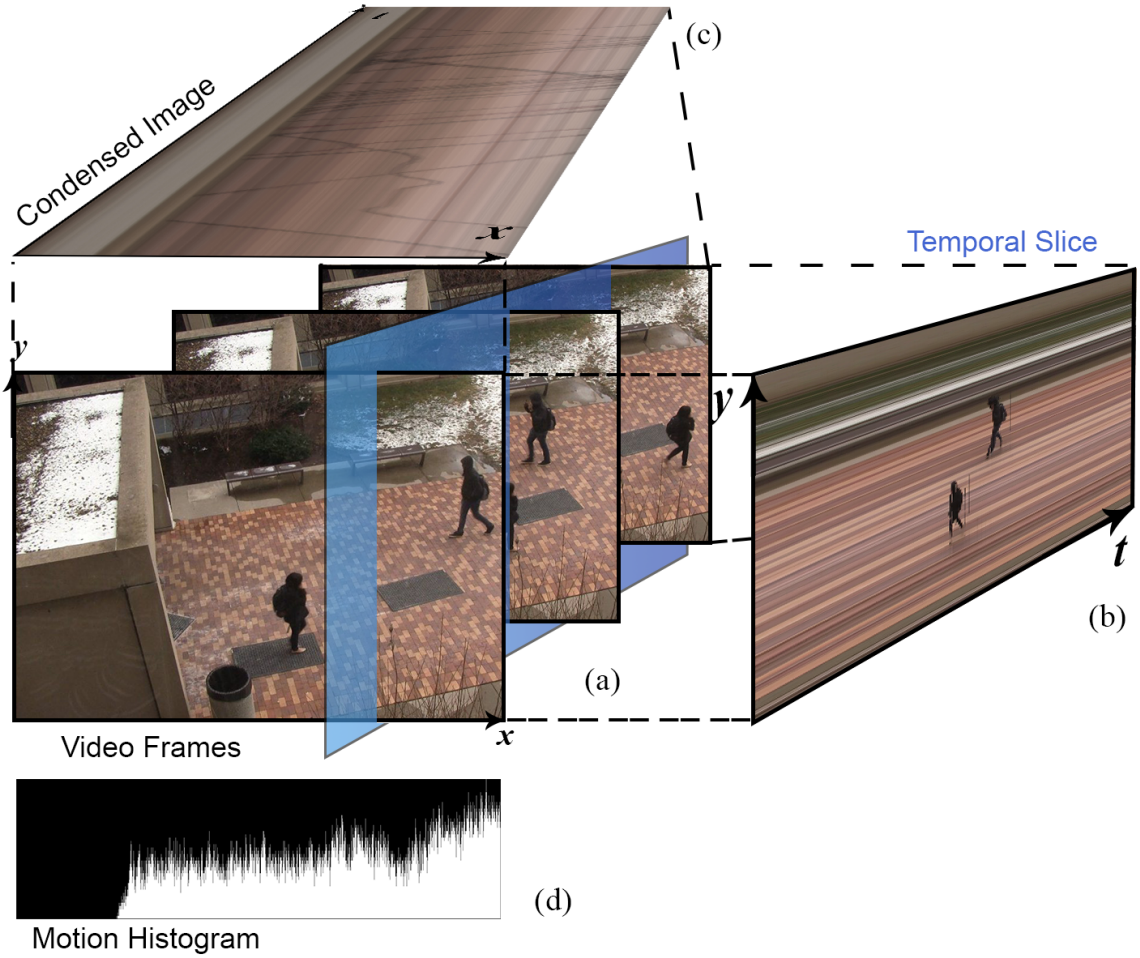
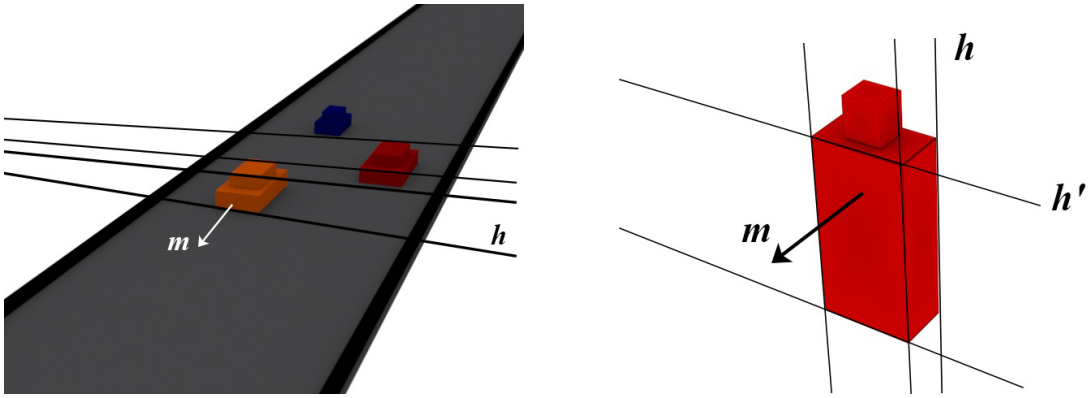


Figure 4.2. Cutting a temporal slice along the time axis in the video at a location where apparent motion exists. (a) Video frames, (b) Temporal slice, (c) A condensed image of video obtained by averaging pixel values in the volume in y direction for observing flow. The traces of people are visible in it. (d) A histogram of flows in the condensed image.

with target objects. The projection of the object motion in the image is **m**=(u,v) known as image velocity, and the projection of principal pose direction in the image is $\mathbf{h} = (h_x, h_y)$. E.g., **H** can be the standing direction of humans in case of pedestrians, or a horizontal direction on a vehicle orthogonal to its moving direction. Figures 4.3(a) and 4.3(b) is an illustration of objects with principal pose directions in the im-

age and in general, $\mathbf{M} \perp \mathbf{H}$. Other 3D motion of targets such as rotation, translation along a camera ray, deformable and articulate body motion, and shaking or waving of natural objects are projected to the camera as minor motion.



(a) (b)

Figure 4.3. Sampling foreground motion with a line in the video frame. A foreground object with a principle pose direction passes a sampling line with the flow vector $\boldsymbol{m}$ in the view. Such a setting preserves the shape of object with minor deformations.

4.1 Directional Flow for Locating Sampling Lines

Because the background is static in a video, $\mathbf{m}$=0, and its trace is along the time axis throughout the video volume. However, a translating object as foreground leaves a trace non-parallel to the time axis, i.e., $\mathbf{m} \neq 0$. We obtain a *temporal slice* $I_i(t, l)$, as described in chapter 2, by sampling the pixel data on a line l over consecutive frames, where l satisfies the equation $(a, b) \cdot (x, y) = c$. In case of the temporal profile described in chapters 2 and 3, l satisfies the equation $x = c$. So for simplicity, l can be parameterized by y (or x) only so that the temporal slice becomes $I_i(t, y)$ (or $I_i(t, x)$) with y mapped from l. If the orientation of l is set non-parallel to local motion direction $\mathbf{m}$ in the video, the foreground will leave shapes in the temporal

slice. Hence, we choose the orientation of line to sample apparent translational flow as follows:

Condition 1: *The fixed sampling line l should be set to intersect a translational flow to capture meaningful shape in the temporal slice, rather than aligning it with the flow, note* $\boldsymbol{l} \times \boldsymbol{m} \neq 0$.

Under this condition, a sampling line is as orthogonal to flow **m** as possible, subject to another condition next for generating better shapes of targets in the temporal slice.

Condition 2: *Sampling line* $\boldsymbol{l}$ *should be aligned with one of principal poses* $\boldsymbol{h}$ *that is more orthogonal to motion* $\boldsymbol{m}$ *in the view. Thus, a target moving across* $\boldsymbol{l}$ *will not leave skewed shape in t direction in the slice.*

Many surveillance cameras have a tilt angle overlooking a large area from a high position (bridge or building). Vertical direction **H** determines poses **h** to be slightly slanted at different locations in the video frame. The **h** directions pass through a vanishing point far below the frame, because vertical vectors are parallel to **H** in 3D space. If a vertical pixel line is used to sample the motion in the frames, the shape of object will be skewed along t axis in the profile, which may increase the difficulty in recognizing the object appearance. We avoid this skew of upright poses by following condition 2. By selecting several building rims or poles in the view, the vanishing point can be computed at the crossing of their extensions. The pose **h** at each image position is the vector connecting the vanishing point and that position. Such a sampling line captures close-to-horizontal **m** in the frame. If a close-to-vertical motion appears in the video, the sampling line will be chosen to align with the second principal direction more orthogonal to **m**.

4.2 Spatially enhancing the profile by blending temporal slices

As discussed above, we set multiple lines in the field of view along the principal pose of targets to sample the video volume. These lines are set at equidistant intervals that are wider than the object size, in order to avoid a target crossing two sampling

lines simultaneously and cluttering the profile. We blend these slices together according to their spatial locations to create a multi-position temporal profile of video that shows the foreground flow clearly. The algorithm for blending is similar to that mentioned in chapter 3 and depicted in figure 3.1. The only difference is that there is more than 3 slices involved in the blending process. A foreground extraction scheme described in 3.1, was used so that a slice without target passing will not contribute to the final profile. A target may pass one or more sampling lines and leave multiple copies in the profile. Although the interval of sampling lines can be set small, the integrated profile may become dense with targets. We thus keep a limited number of sampling lines to avoid cluttering the profile. In order to probe locations with apparent motion flow, we measure the flow amount in the FOV. A condensed image as described in chapter 1.2 was used to build a motion histogram. As visible in figure 4.2 (c), motion condensed image indicates the global motion trajectories in the video. Background traces are parallel to the t axis and it can be inferred that edges non-parallel to the t axis present horizontal motion in video. By differentiating the condensed image along t direction, we remove the background and find foreground traces. Then we accumulate the number of foreground traces in t direction, and find the spots with large values in the histogram (figure 4.2(d)). Locations for line cutting, though not necessary to be unique, are obtained.

Condition 3*: The sampling locations of lines are obtained at the places where the amount of translational flow are high.*

This condensing process is also implemented horizontally for selecting close-to-horizontal lines to sample the camera view. The lines can cover a large span in the view for finding flows in vertical direction.

To include all the moving targets from the entire FOV as complete as possible, we blend vertically orientated temporal slices together for the final 2D temporal profile. To reflect more spatial information in this dimension-reduced profile, we use different transparencies for slices according to their spatial positions in the video frame. This

creates a haze effect for visualizing the distance of slices from one margin of the video volume, either left or right depending on the 3D scenes.

Similar to what was discussed in chapter 3, each temporal slice $I_i(t, l)$ has a blending coefficient α_i that determines its contribution to the final temporal profile. The profiles are blended using equation 3.5. In other words, we blend consecutive slices of extracted targets onto a background slice to provide a context independent of background removal. If the value of $mask_i$ is zero at a position, the color value in P_{i-1} is used. The coefficients α_i decrease for slices away from a frame margin such that targets fade away in depth. As long as the slices are blended in such a spatial order, the motion direction can be recognized in the resulting profile. Figure 4.4 shows the final results of multi-position temporal profile for observing opacity changes on moving targets.

Figures 4.5, 4.6 show examples where pedestrians walk on campus and vehicles move at an intersection. Moreover, through a series of copies of a target, its image velocity can be estimated by matching them in consecutive foreground slices (figure 4.5), under the condition that the different targets are not occluding each other severely.

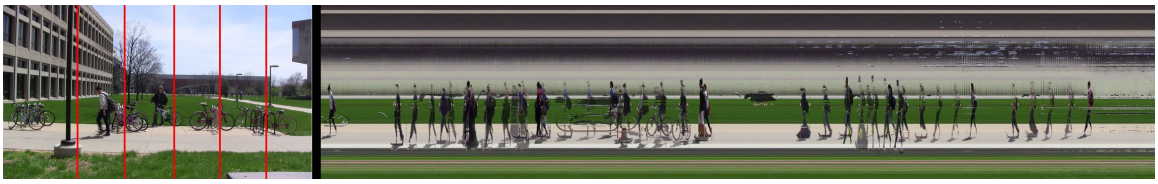


Figure 4.4. Temporal profile of a video recording a path by sampling slices (red lines) with the opacity reduction from left margin.

4.3 Haze Effect and Background Embedding

The order of transparency changes from opaque to transparent to show an increasing depth in the scene. Now let us determine the order to fade temporal slices.

Figure 4.5. Setting sampling lines evenly at all positions for a profile. (left) An interface for specifying sampling locations (red lines). (right) Temporal profile of the entire video. The passing directions of people can be figured out from the change of transparency. The higher the transparency, the more the target appears on right in the space.

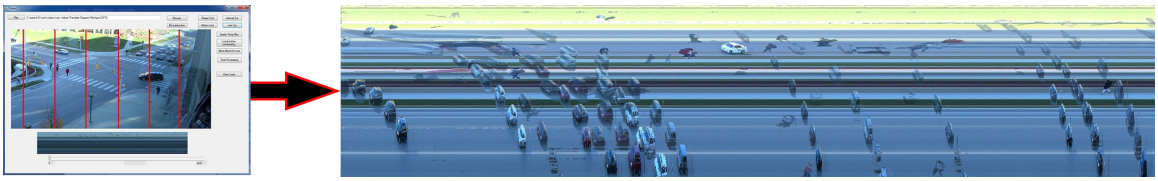


Figure 4.6. Global motion in temporal profile generated at an intersection. Series of copies of cars are extracted when they move from right to left in the view (from transparent to opaque in profile). The vertical displacement in the profile is due to image velocity component v than in the y direction. The time delay is shorter for cars than for pedestrians because of a faster speed of cars in the view.

Taking the example of close-to-vertical slice for near-horizontal motion/path, scene layout can be close at either margin of the frame as shown in figure 4.7. This is determined by the camera orientation: (a) right facing, (b) orthogonal, or (c) left facing with respect to a path regardless of the camera tilt. Here we denote left-close scene (a) and right-close scene (c) as LC and RC layouts, respectively. In addition, a lower position in the frame normally has a closer depth, if the camera viewpoint is set high. Overall, the transparency assigned to different slices should reflect the target depth.

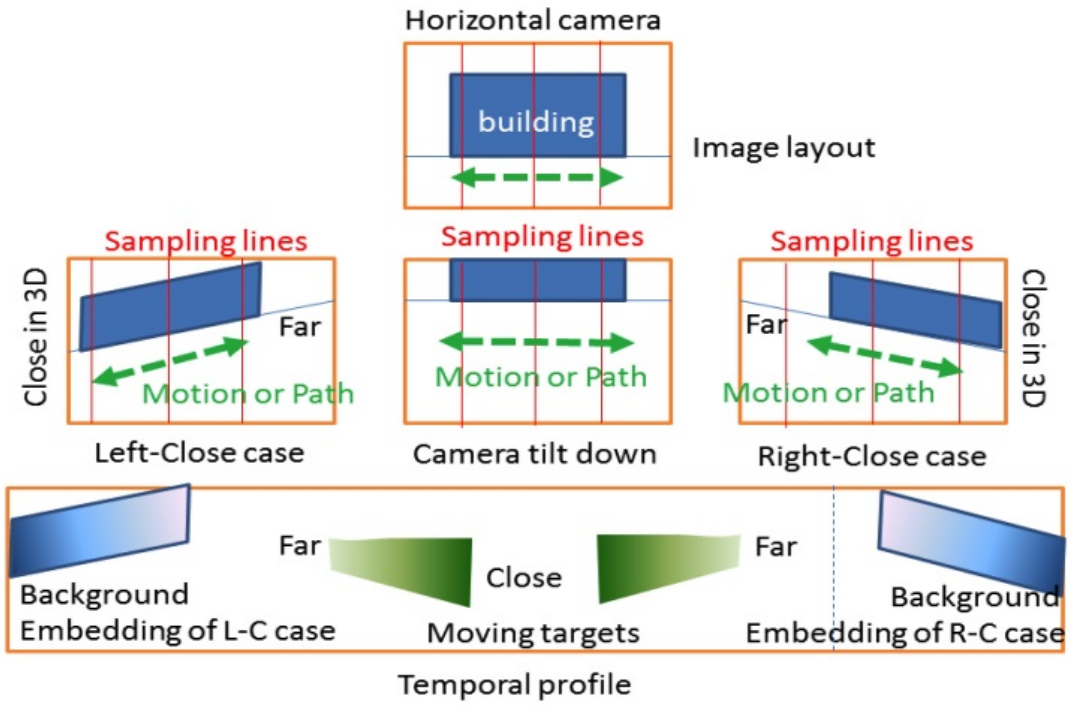


Figure 4.7. Determining the transparency in blending slices to reflect depth in 3D scene layout. (Top) Camera orientations w.r.t. major motion. Green dash lines show major two-way path. (Bottom) Temporal profile with opacity changes on moving targets and background.

We select the left margin to be opaque for LC layout, and right margin to be opaque for RC layout. For orthogonal case (b) or multiple paths without a dominant one, either side can be selected as the opaque margin. The transparency then increases (α decreases) towards the other frame margin as described equation 3.5. Using this strategy, the generated temporal profile shows a close target at a low position with high opacity. If the target is away at a far depth, it becomes small at a higher position in the temporal profile, rendered more transparent as illustrated in figure 4.7. Based on this design, the transparent-to-opaque change of target along the temporal profile indicates a leftward motion in LC layout, or a rightward motion in

RC layout. Inversely, the moving direction presented by an opaque-to-transparent transition can also be derived for LC and RC layouts easily.

With dynamic targets listed along the temporal profile, one still cannot grasp the scene layout. Targets with less movement are also missed in temporal slices. Our next effort is to embed the entire background view into the temporal profile to remind the spatial locations and target existence. This embedding is implemented in the period without foreground to avoid overlapping with dynamic targets and yielding incorrect spatial relation. We cut a diagonal slice in such a static period across all the background traces in video as shown in figure 4.1. The cutting direction is from left to right in the volume for all layouts. This makes background visible from sideways and keeps the definition of temporal profiling correct, i.e., for any point visible in $P_i(t, y)$, it is in frame t in the video. We also fade background slice to show the depth hazing effect. The opacity is proportional to the image position x changing from opaque to transparent for LC layout and from transparent to opaque for RC layout along the time axis, respectively. This is consistent with the order for temporal slices in varied depths. Figure 4.8 shows the results with LC background embedding. This work has also been publish by us in [19].

Figure 4.8. Results of temporal profiles from surveillance video with background embedding and foreground hazing. Original frames are also shown.

5 VELOCITY BASED TEMPORAL PROFILE

In chapters 3 and 4, we proposed solutions to generate an intuitive representation of the video. However, this framework still has shortcomings in handling special cases. For example, if the targets in the video have non-translational motion, such as a person standing in the same place while delivering a speech, or dancing the aforementioned solutions do not perform well. To handle targets with non-uniform, non-translational motion in our temporal profile we will need fine level motion analysis on targets. This means that we will need pixel level velocity information about the video volume. This is a classic problem and many works have proposed methods to measure the optical flow in the video, [20], [21]. However, in addition to being extremely expensive to compute, optical flow methods tend to estimate optical flow for regions of images. This means that an object could be estimated as multiple regions with different flow values and this would not be ideal for our work as part of the object will have "false" zero values because they fall on the boundaries of these approximated regions.

We thus take a simple, yet effective approach to estimate the image velocity of each pixel in the video. In the following sections we will give a complete overview of our approach to estimate image velocity. Once we have calculated the image velocity for each pixel, we can use this information to enhance our temporal profile and extend it one step further. We will then explain how we utilized this to form a new representation of video and construct a "Velocity Based Temporal Profile".

5.1 Object Motion and Image Velocity

A more general analysis of objects in the scene is possible, but for the purpose of this study we focus on motion analysis of objects with regards to a static camera.

In other words, the only apparent motion in the scene is from the foreground object. Assume an object is located at point $P_1 = (X_1, Y_1, Z_1)$ in the 3D world at time t_1, and at point $P_2 = (X_2, Y_2, Z_2)$ at time t_2. The projection of these points onto the video volume will be $p_1 = I(x_1, y_1, t_1)$ and $p_2 = I(x_2, y_2, t_2)$ respectively. Considering the brightness consistency constraint, we can say that the intensity value of voxel p_1 at time t_1 equals to that of point p_2 at time t_2.As mentioned in [22] this results in the following equation:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t}\frac{dt}{dt} = 0 \tag{5.1}$$

And we can equally write

$$I_x\mathbf{u} + I_y\mathbf{v} = -I_t \tag{5.2}$$

where $\mathbf{u}$ and $\mathbf{v}$ are the components of velocity along the x axis and y axis respectively in each frame of the video, and I_x, I_y and I_t are the derivatives of image at (x,y,t) in the corresponding directions. So we can re-write the equation as :

$$\nabla I \cdot \vec{\mathbf{m}} = -I_t \tag{5.3}$$

Figure 5.1 is a clear illustration of this movement in the video volume.

5.2 Our Approach to Calculating Image Velocity

In order to study the horizontal motion of objects in the video, we can decompose the video volume into horizontal temporal slices. If we set a sampling line at each row of video frames, we can get h temporal slices $I_j(t, x), j = 1, ..., h$ for a video volume, where h is the frame height. These temporal slices reveal motion traces of objects for the duration of video, and by analysis of the motion in these slices, we can obtain image velocity values for each pixel position. Id est, each one of these slices can be used to find the image velocity of one row of our temporal slice at the time targets cross it.

Our task therefore is to find the horizontal component of image velocity,**u** in each slice. Image velocity of objects on our vertical sampling slice, equals the slope of the trace it leaves horizontally at each time instance t. As depicted in figure 5.2, the velocity along x axis noted **u** captured in our horizontal slice equals the displacement along the x axis between each two frames; Hence we calculate the horizontal velocity by calculating the gradient in each horizontal slice. For objects with non-uniform motion, for parts of the object that cross our sampling line slowly, the traces look stretched and tend to clutter the temporal profile. The information that these near-static parts of video provide is not key, and it might even be undesirable as they occlude and destroy other parts of the object that would otherwise appear clearly. For example, the waving of the tree leaves in the wind might cause some of the leaves to cross the slices back and forth while others might stay static. If we analyze the motion of the leaves, we would want to see the ones crossing back and forth, but not the ones that are static.With this said, it makes sense to give more weight to objects passing the slice with higher velocity than those otherwise. We use the velocity of each pixel calculated at the time it crosses our profile to determine the contribution of that pixel to the final profile. Since the vertical velocity of the object **v** only leaves skewed shapes in the slice as mention in 2.3, we only consider **u** as a weight. Hence we will find **u** by finding the slope of the traces in the horizontal slice $I_j(t, x)$.

Due to the nature of temporal slices, once an object crosses the slice it leaves a clear trace on it. This means that we will get clear edges that helps us determine the velocity of the object when passing the slice. However, because we are cutting the object off by sampling it we are creating false edges that do not represent the motion of the object. These edges are usually because the object has some vertical motion **v** and so it does not stay at the same height for the duration of the video. They represent the boundary of object when they are crossing the horizontal slice and usually form edges along the x axis in the slice $I_j(t, x)$.

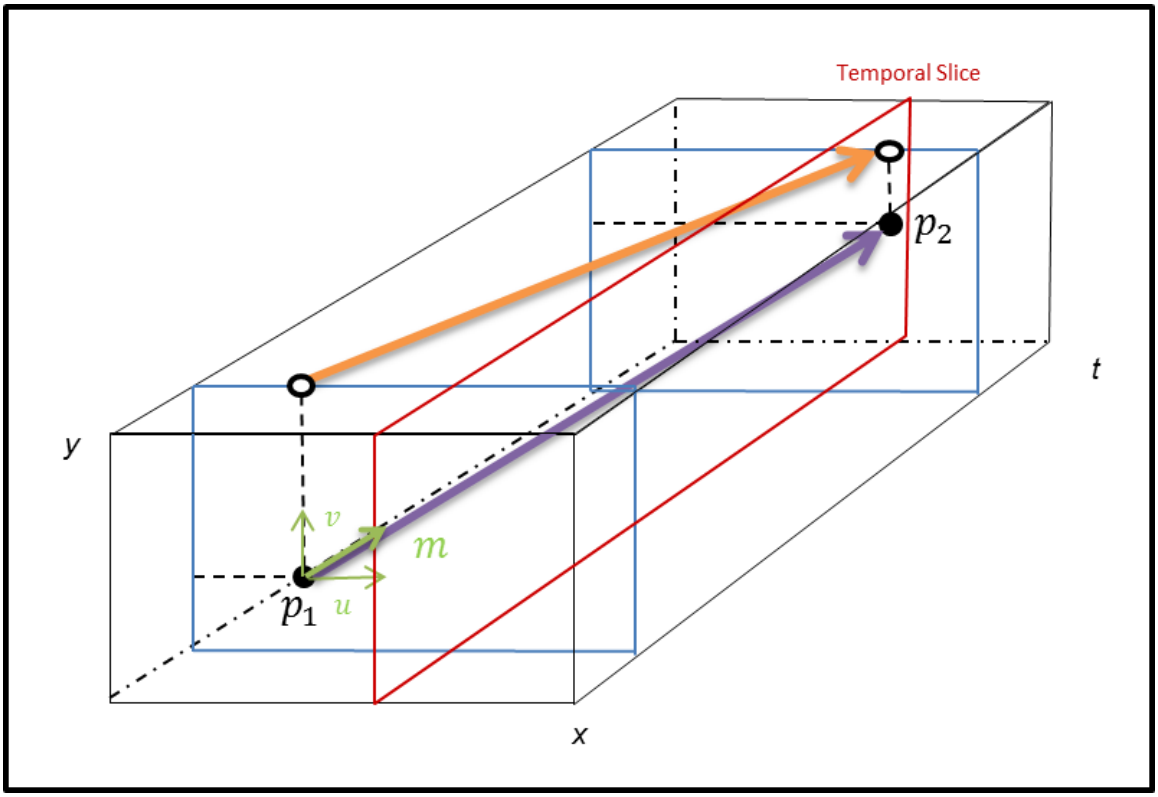


Figure 5.1. Diagram of an object moving from point p_1 to p_2 in the video volume. The object has both vertical and horizontal motion (**m**=(u,v))

We work our way around these "false edges" by averaging the horizontal slices to get a more robust slice that is representative of object motion. We utilize a horizontal condensed image, similar to 1.2 as

$$\bar{I}_y(t,x) = I(x,y,t) * g(y) \tag{5.4}$$

where

$$g(y) = \begin{cases} 1, & |y| \leq \frac{N}{2} \\ 0, & otherwise \end{cases} \tag{5.5}$$

and N is the number of frames averaged vertically. The false edges along the x axis would not appear as strong because they are not representing the motion of the

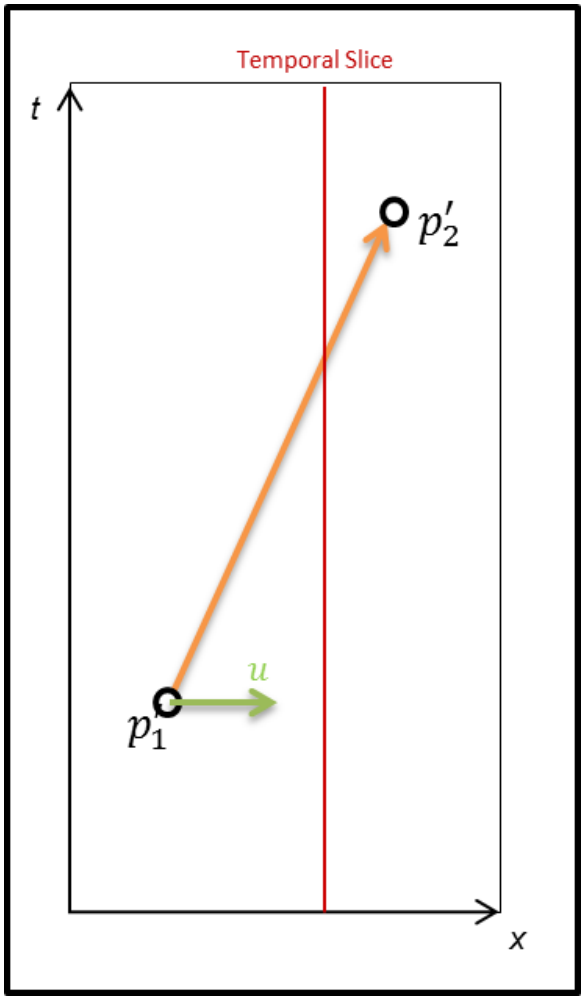


Figure 5.2. Projection of motion vector **m** unto the horizontal slice $I_j(t, x)$. Only the horizontal motion is apparent in the horizontal slice.

object, hence by averaging along the y axis they become weaker. The real edges however preserve their strength as they are representing the motion of the object. It is noteworthy that the choice of N is not very critical for serving our purpose, and it can be chosen anywhere between 5 and 10.

figure 5.3 shows A clear flow diagram of our approach to calculating the image velocity of objects for the temporal profile. After obtaining the condensed image, we calculate the gradient in the x and t directions denoted $\nabla_x \bar{I}_y(t, x)$ and $\nabla_t \bar{I}_y(t, x)$.

Based on the values of these differentiations, we can compute the orientation and magnitude of the gradients at each point as:

$$\gamma(t,x) = \begin{cases} arctan(\frac{\nabla_t \bar{I}_y(t,x)}{\nabla_x \bar{I}_y(t,x)}), & \nabla_x \bar{I}_y(t,x) \neq 0 \\ \pm\frac{\pi}{2}, & otherwise \end{cases} \tag{5.6}$$

$$\|\nabla \bar{I}_y(t,x)\| = \sqrt{(\nabla_x \bar{I}_y(t,x))^2 + (\nabla_t \bar{I}_y(t,x))^2} \tag{5.7}$$

Where $\gamma(t,x)$ is the computed gradient orientation for each point and $\|\nabla \bar{I}_y(t,x)\|$ is the magnitude for each point. However, the gradient image $\nabla \bar{I}_y(t,x)$ will contain some noise due to camera shakes and illumination variations in the scene. To handle such noise, we use the neighboring information of the pixels to determine a more robust direction for each location. We apply a median filter on the orientations (independent of magnitude), to get a more smooth orientation for each region. Our task is then, to find a major orientation for each row in the condensed image, that is for each time instance. We should explain that the magnitude $\|\nabla \bar{I}_y(t,x)\|$ is not very meaningful in the condensed image or the horizontal slice because it represents the strength of the edges found; This means that an object that has more contrast with the background will have stronger edges. However, we can still use the magnitude the determine the more important orientations in the image. For example the gradient orientation for position x_0 at time t_0 in the image may equal to zero $\gamma(x_0,t_0) = \frac{\pi}{2}$. This might mean that the gradient orientation at point (x_0,t_0) is equal to $\frac{\pi}{2}$ or that $\nabla_t \bar{I}_y(t,x) = 0, \nabla_x \bar{I}_y(t,x) = 0$. This is where we can use magnitude to rule out these unwanted cases and get more robust major orientation for each time instance of the condensed image. Here $\gamma(t,x)$ is the major orientation of the gradient. However, to find the velocity of the objects, we need to find the tangent of the traces. So we will find the tangent direction of a gradient vector denoted $\theta(t,x)$ as the vector perpendicular to $\gamma(t,x)$ such that it is pointing to the direction where t is increasing:

$$\theta(t,x) = \begin{cases} \gamma(t,x) - \frac{\pi}{2}, & -\pi \leq \gamma(t,x) \leq -\frac{\pi}{2} \\ \gamma(t,x) + \frac{\pi}{2}, & -\frac{\pi}{2} \leq \gamma(t,x) \leq \frac{\pi}{2} \\ \gamma(t,x) - \frac{\pi}{2}, & \frac{\pi}{2} \leq \gamma(t,x) \leq \pi \end{cases} \tag{5.8}$$

Where $\theta(t,x)$ is between $-\pi$ and π. We want to take the average of orientation values where the magnitude values is more than a threshold δ_m. Since $\theta(t,x)$ is a circular value, we have to take the average of each component separately. This actually simplifies our work since we are only interested in the horizontal component $\mathbf{u}$. Thus we calculate the normalized horizontal velocity denoted as:

$$\mathbf{u}(t,x) = \frac{1}{tan(\theta(t,x))}, \qquad \forall \quad \left\{ \theta(t,x) \quad \middle| \quad \|\nabla \bar{I}_y(t,x)\| > \delta_m \right\} \tag{5.9}$$

And the average of normalized horizontal velocity as:

$$\bar{\mathbf{u}(t)} = \frac{1}{N_m} \sum^{N_m} \mathbf{u}(t,x) \tag{5.10}$$

Where N_m is the number of elements that satisfy the condition in equation 5.9 Each condensed image $\bar{I}_y(t,x)$ will the contribute to calculate one row of the velocities in our motion temporal profile $I_i(t,y)$.

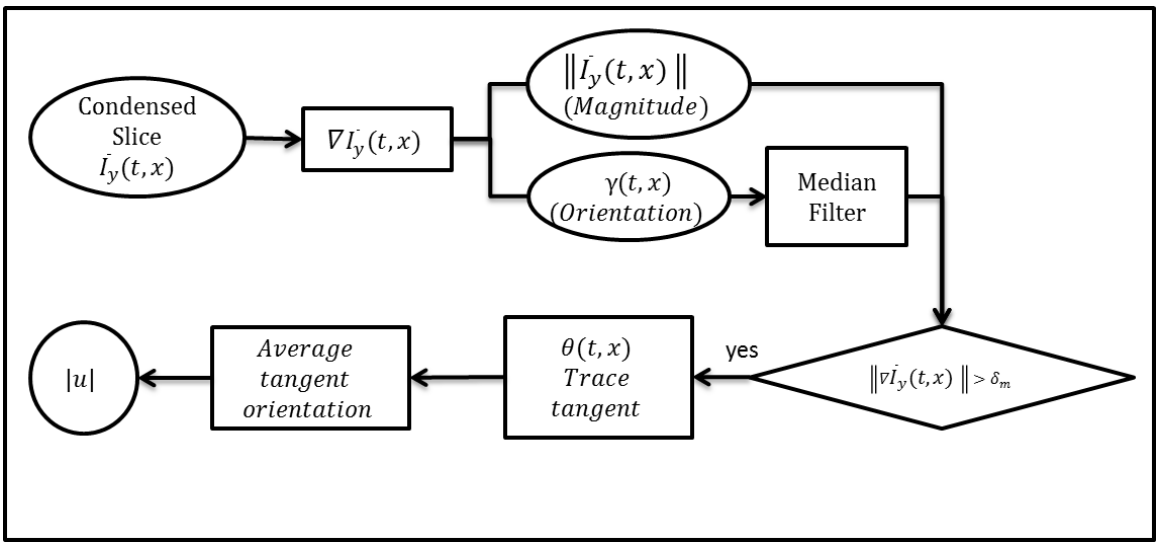


Figure 5.3. Flow diagram of determining image velocity of objects from horizontally condensed images. Each condensed image provides velocity values for one row of our velocity based temporal profile

6 PROFILES FOR PANNING CAMERAS

In this chapter we will cover an extension of our work that will make it usable for videos with smooth panning motion. Aside from static surveillance cameras, panning cameras are the most commonly obtained surveillance video cameras. Panning is usually achieved by the means of a motor rotating the camera along an axis with a constant velocity. Hence it is reasonable to assume smooth panning motion for surveillance videos. Figure 6.1 shows the panning motion of a surveillance camera.

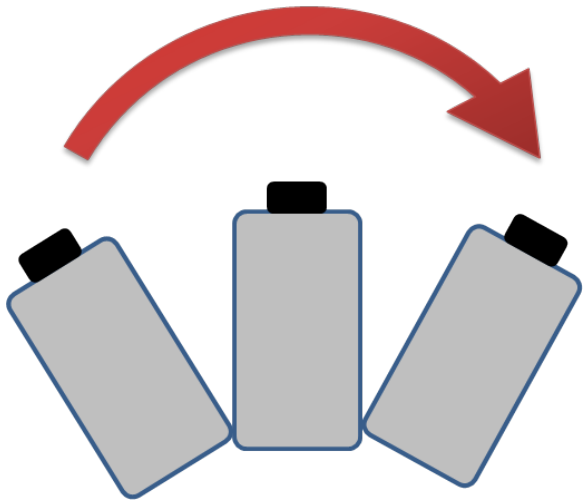


Figure 6.1. Panning motion of a camera along the x axis.

In this chapter we will analyze the motion of the camera, in order to transform the panning camera video to a panorama-like static video. We can then use any of the above mentioned techniques to obtain a temporal profile from this video sequence. For simplicity, we will focus on cameras with smooth panning motion along the x axis, but this study can easily be generalized to cameras with smooth motion along one arbitrary axis.

Let us assume that a camera overlooking a scene is smoothly panning with the constant angular velocity ω_c. So the movement of objects in the scene will be with the same velocity but in the opposite direction denoted $-\omega_c$. If we obtain a condensed image of this video, C(x,t) as described in section 1.2, all of the traces will be facing the direction opposite to that of the camera movement, as you can observe in figure 6.2.

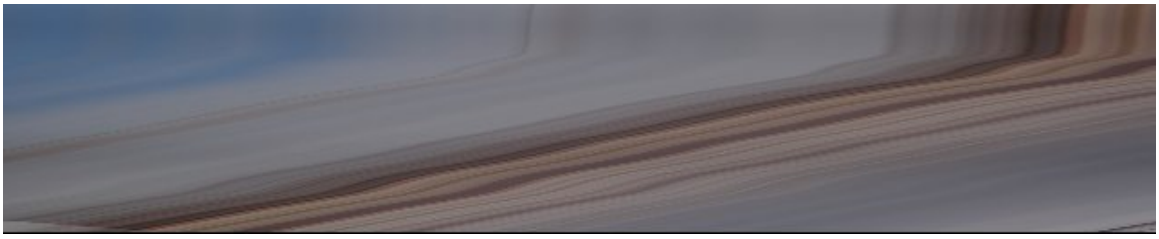


Figure 6.2. Condensed image of a camera with smooth panning motion.

According to section 5.2, we know that the velocity of the camera movement equals the slope of the trace it leaves in the condensed image. Zheng and Cai give a more detailed explanation on how the traces are connected in the velocity of the camera in [12] and [23].

Hence by performing edge detection on images, we will find a displacement between two frames and correct the position of the video frames so that the will appear as static. Figure 6.3 shows a flow diagram of detecting the camera pan speed, very similar to that of figure 5.3 with the exception that we want to find the pan direction as well as the magnitude.

Figure 6.4 is a condensed image of hand-held panning video. The video is panning from left to right and the direction and magnitude of panning is detected.

Once we have detected the motion of camera, we can compensate for it by zero padding the missing pixels in the video. This would mean we would have a smaller field of view as we are panning out of the current field of view. We translate the pixels back by the amount of motion, so that we get a stabilized video looking like

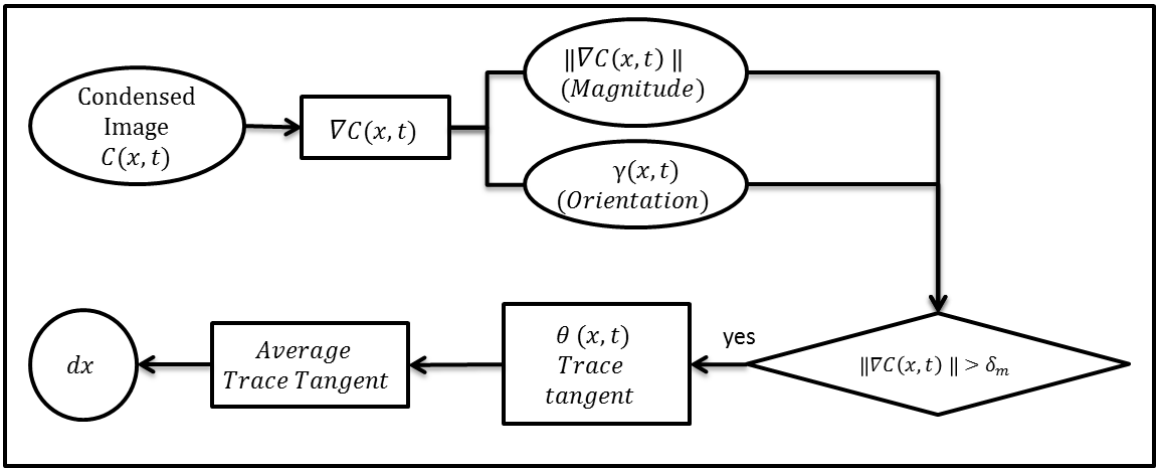


Figure 6.3. Flow diagram of determining camera motion for panning cameras.

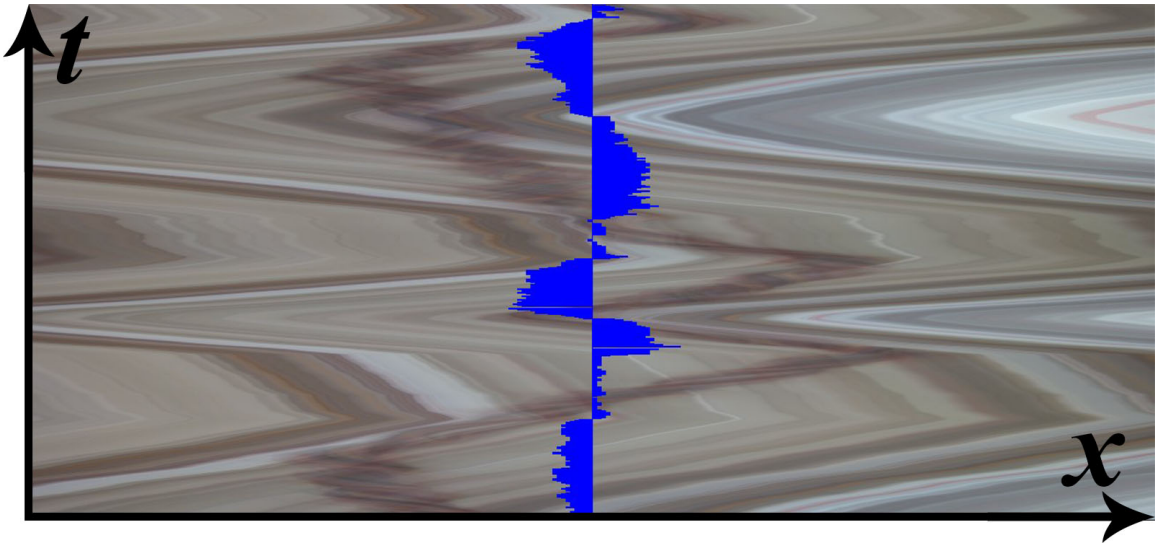
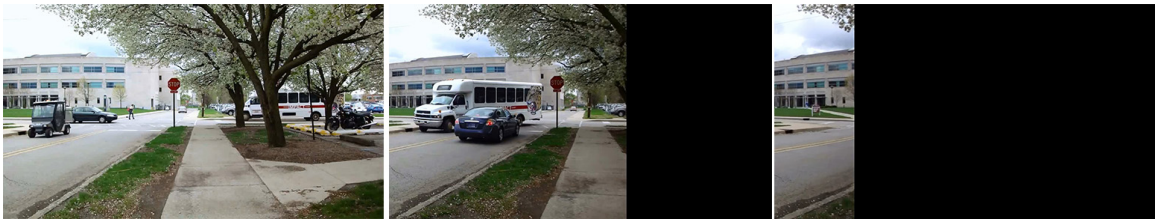


Figure 6.4. Condensed image of a video panning left and right repeatedly. Our method successfully detected the direction and magnitude of camera motion.

a static video that gets smaller. Figure 6.5 is an example of the corrected video by our method. Figure 6.5(a) is 3 frames of the original video, and the corrected video frames are visible in figure 6.5(b). Figure 6.6(a) shows the results of the camera

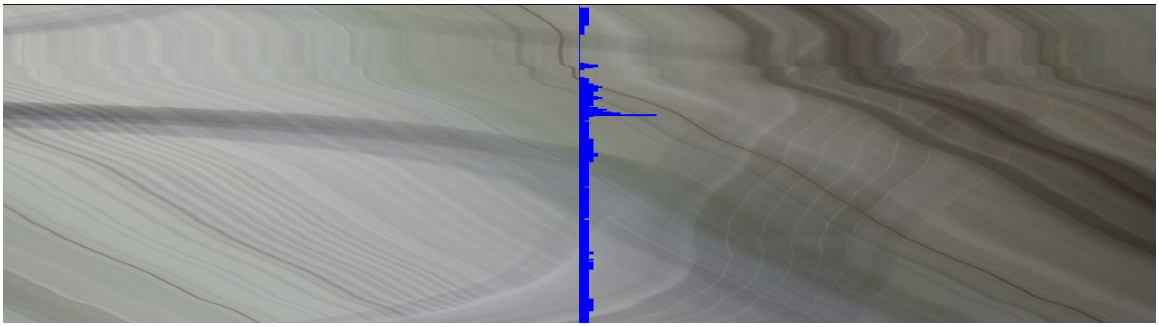
motion detection in the condensed image. After the correction we can obtain a temporal slice as described in 2, by cutting the regions that stay in the field of video. Figure 6.6(b) is an example of a temporal slice obtained after stabilizing the video.

(a) A video with smooth panning camera motion.



(b) Static video created by compensating the panning motion of the camera.

Figure 6.5. The original panning video (a) and the corrected static video (b)



(a) Condensed Image of a video with smooth panning camera motion. The direction and magnitude of the motion is shown as the blue bar chart in the middle of the image.

(b) The resulting temporal slice cut in the corrected video.

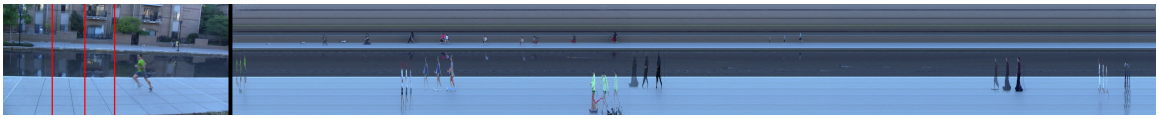
Figure 6.6. Detecting the motion of panning camera, correcting the video and cutting a temporal slice in the video.

7 RESULTS AND DISCUSSIONS

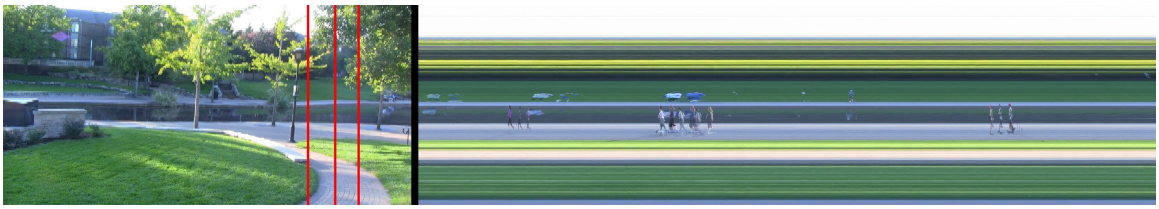
7.1 Localized Temporal Profile

We have tested various experimental videos including indoor and outdoor scenes. For each long video, sampling locations are set manually at critical locations according to the layout of the site in the field of view and motion directions there. A graphical user interface is developed to perform video profiling on PCs in real time (60 frames per second). The quality of the figures in profile is compatible to that in video frame in most cases. Figure 7.1 displays several video profiles at different locations. We use multi-lines to capture temporal slices with synchronized time stamps. Background and foreground separation is implemented stably in the slices. Blending of slices gives a natural visual effect. By incorporating the shape and temporal information acquired in the profiles, we obtain global motion directions in the video. Finding the frames with passing targets in the video becomes extremely efficient by scanning the temporal profile. Furthermore, cutting off segments without targets from profiles is straightforward.

Compared to tracking a target path in video frames, our framework yields the motion transition between critical locations as shown in figure 7.2. Distributed surveillance cameras, even with non-overlapping viewing sights can take advantage of temporal profiles. Because the spatial relation of distributed cameras is known, we can model the cameras as a graph. Once a target is spotted in a camera, we only need to search the neighboring cameras after that time instance. Determining the path of target then becomes a single source shortest path problem. With temporal profiles generated from all nodes, the search of following node becomes the search of profiles with neighboring relations.



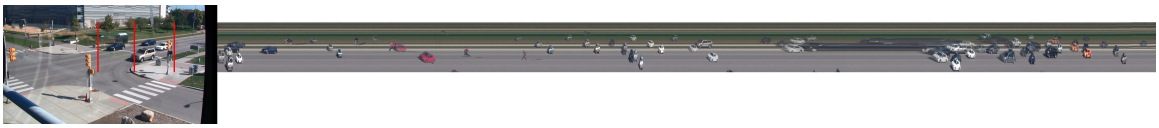
(a) Temporal profile generated for a 2:31 min long video.



(b) Profile at two paths on two sides of a river, where aspect rates are different for people walking at almost same speed at different depths.

(c) Sampling at an outdoor area with multiple paths. Leave waving is also observed in green part.

(d) Profile from a surveillance video at the entrance of a building with multiple targets crossing in different directions at the same time.



(e) Sampling at an intersection where pedestrians and vehicles appear.

Figure 7.1. Experimental results of localized temporal profiles from surveillance video, time axes are horizontal.

7.2 Multi-Position Temporal Profile

We have tested various videos including indoor and outdoor scenes. A GUI is developed to perform the localization of sampling lines through the vanishing point and profiling of entire video in real time. The spatial resolution of figures in profile is the same as in video, except the deformation in the time domain. The background

estimation on temporal slices uses a median filter on 400 lines but still maintains a low cost in sorting to generate temporal profile in real time. The quality of background removal does not affect the results very much because the extracted foregrounds are displayed in transparent regions on a sampled slice without being influenced by segmentation. Figure 7.3 shows some results of the Multi-Position Temporal Profile.

In many cases, the temporal width of a target is squeezed due to the limited frame rate (60Hz) on close objects with fast speeds (figure 4.6). This distortion can be reduced by selecting a camera direction less orthogonal to M in the 3D space (as LC or RC layouts), which results in small —m— in video. A distant target has wide shape, but pushing it far away sacrifices its resolution. The fading direction is determined from the path/scene layout by human once for a surveillance camera.

By incorporating the layout and temporal information in the profiles, we can obtain global motion directions in the video. The density or interval of target copies in the profile is controlled by the interval of cutting lines, and is influenced from target moving speed and depth. With close target intervals, temporal profile can even be utilized to show actions in the video with efficiency. Targets may have a collision in the temporal profile if they simultaneously appear on their sampling lines at the same height, which can be separated by directing the camera to a side of the path overlooking the ground (less horizontal or orthogonal to the path). Even if targets collide, further examination around nearby frames can be trigged by clicking the position in the temporal profile. The temporal profile can be viewed by human operators in scrollable windows for target identification or analyzed by algorithms in the future. The 2D profile has a much compact data size as compared to synopsis video and can be further shortened by removing long periods without target motion. It is not necessary to be the optimal video summary, but is particularly efficient for surveillance video browsing.

7.3 Velocity Based Temporal Profile

We have generated a mask using the method described in 5 and used this transparency mask to blend our foreground into the traces obtained as the background model. As you can see in figure 7.4 the velocity based temporal profile helped remove some of the unnecessary artifacts created by non-translational motion of objects on the sampling line. As you can see, the resulting profile is much more pleasant and informative. Another example is visible in figure 7.5.

However, there are short coming related to this work. Similar to the Localized Temporal Profile and Multi-Position Temporal Profile, the resolution of the profile is dependent on the temporal resolution of the video. The cut position and angle is chosen manually and greatly affects the spatial quality of the profile. The calculation method is based on extracting a major gradient direction from the horizontal slice, which means it could be subject to noise. As the major gradient direction is obtained from averaging a neighborhood around the temporal slice cut position, objects that occlude each other may introduce some noise in the profile.

7.4 Profiles for Panning Cameras

We have included a wide range of videos in our results. Figure 7.6 shows some of such results. You can see that there is some shaking artifacts in the resulting corrected video. This is due to minor shakes in the videos. As we mentioned before, we are detecting the motion of cameras based on the fact that cameras are panning with smooth motion. In future, we might want to remove such noise before detecting the camera motion velocity.

However, simply using the condensed image might not give very robust motion estimation for the camera. If a moving target is covering most of the field of view, it will leave strong traces in the condensed image that will interfere with detecting the major flow. Because such traces will have strong edges, the gradient orientation detection will detect them as major flow. This means that they either interfere

with the correctly detected flow, or they themselves are detected as major flow. As a workaround about this problem, we can utilize the horizontally condensed slices again.The idea is that objects that leave traces in parts of the video, will not be apparent in all of th video height at all times. We can divide the video volume into multiple non-overlapping sub-volumes, such that each sub-volume will have the same width as the original video, but only a fraction of the original video height. A condensed image is the calculated for each of these sub-volumes and the camera motion is estimated for each. The major flow of the camera motion is the mode of the detected motion in each slice.

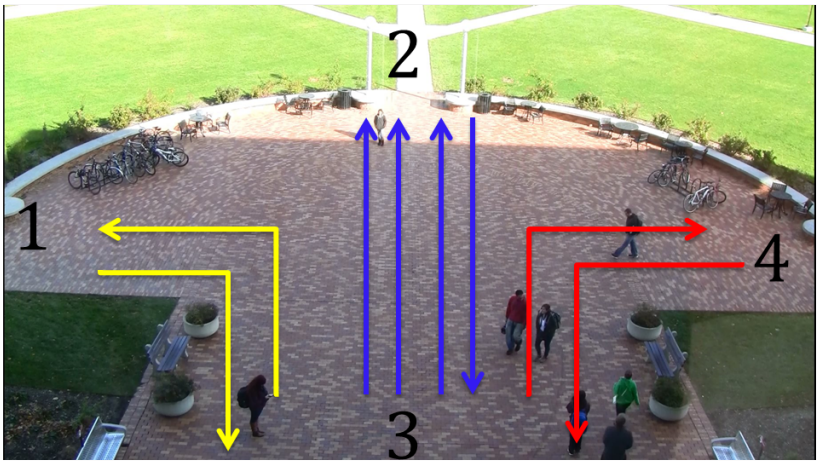
7.5 Discussion

For the purpose of this study, we have built a database of more that 120 video clips, all shot by us using an HD video recorder. These clips include videos of indoor scenes as well as outdoor, with static shots and panning camera shots. Targets with different motion characteristics such as humans, bicycles and cars were some of the subject recorded for our study.

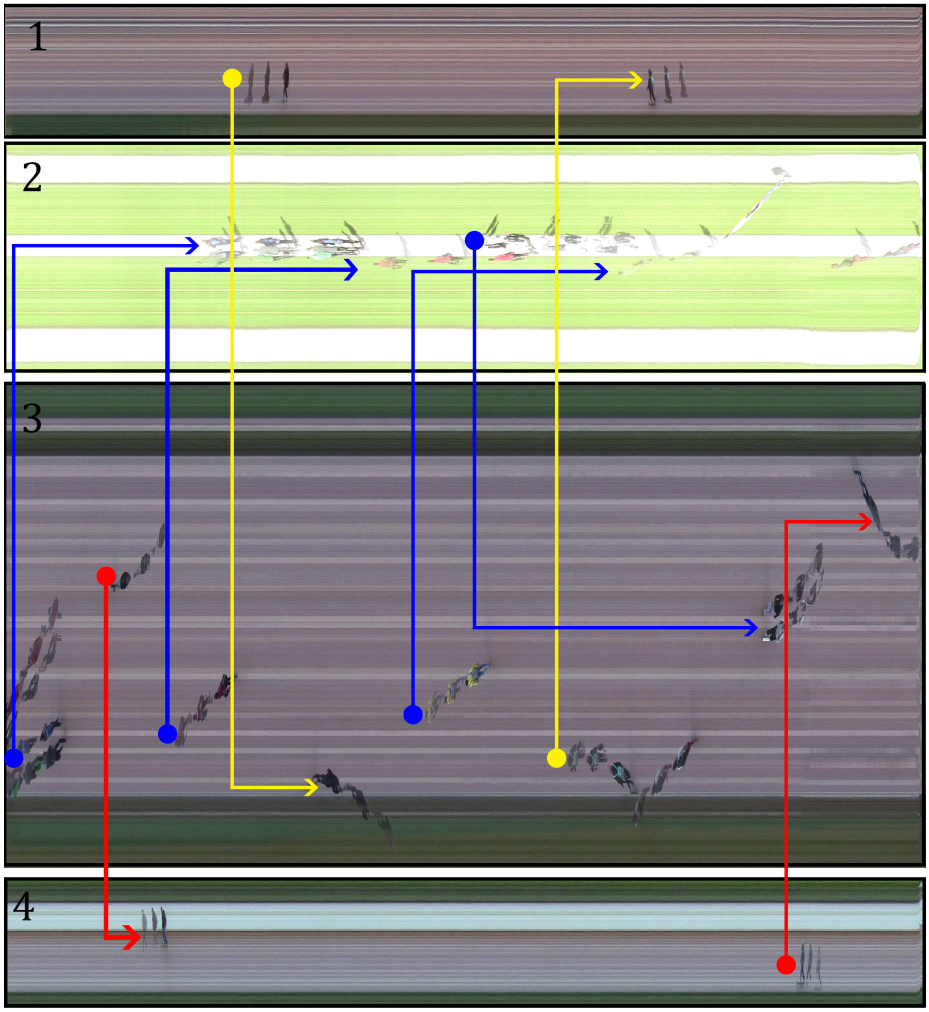
Furthermore, we developed a GUI to facilitate and advance our tests. Windows was used as the main platform for our tests and all the code was written in C# in a windows forms framework. Emgu CV library was utilized in most of our code to accelerate the process and help with our proof of concepts. Our tests were run on a Dell XPS desktop with 16 GB of RAM and i7-3770 3.40 GHz.

For all of the algorithms above, a non-parallel implementation has been provided that leads to linear run time with regards to the video length. For a 2 minute video, any of the above mentioned profiles will take less than 10 seconds to generate. The velocity based temporal profile is has a preprocessing step that will sample the video at all of the rows, which might take longer, but unfortunately a more detail run-time analysis has not been done for this work.

The current framework provides a linear mapping between the time in the video and the spatial resolution in the image. As a future work, it is possible to generate a non-linear profile that accentuates the events rather than the time. Such profile will be extended longer for more eventful durations and will provide a more concise representation of the non-eventful parts of the video. So a dynamic sampling rate algorithm has to be implemented to sample more frames for the eventful parts of video (maximum 60 fps) and less frames for the other segments. As another future development direction, we could increase the density of sampling in the FOV as opposed to a fixed set of sampling lines to capture events in periods when the field of view is more busy to reflect on busy frames rather than frames with scarce motion. This requires some analysis of the video frames and there would always be a trade-off between the computational complexity and the amount of concise information provided in each case.



(a) Critical locations and their spatial relations,colors indicate transitions.



(b) Profiles showing transitions from one location to another.

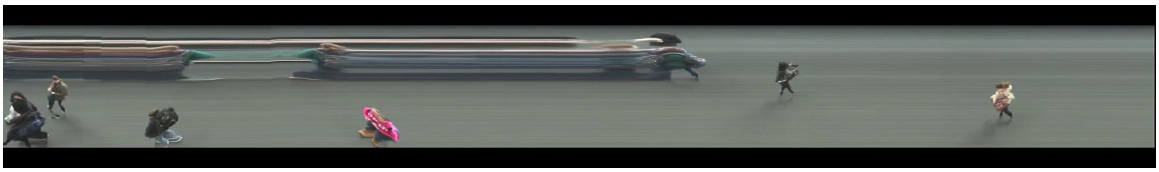
Figure 7.2. Localized temporal profiles from four different locations for finding transitions of people between locations.

(a) A global temporal profile of scene with a path orthogonal to a horizontal camera direction. Tree waving on a slice is included in foreground.

(b) Sampling at an outdoor path orthogonal to the camera with tilt down. Some targets only pass three lines when the clip starts.

(c) Profile from a surveillance video at a building entrance with multiple targets passing in two directions.

Figure 7.3. Experimental results of multi-position temporal profiles from surveillance videos with background embedding and foreground hazing. Original frames are also shown.

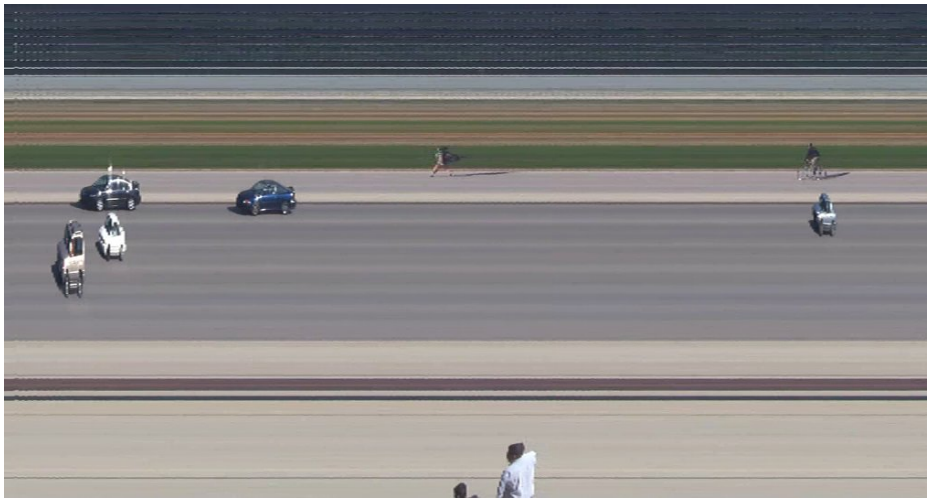


(a) The initial temporal slice

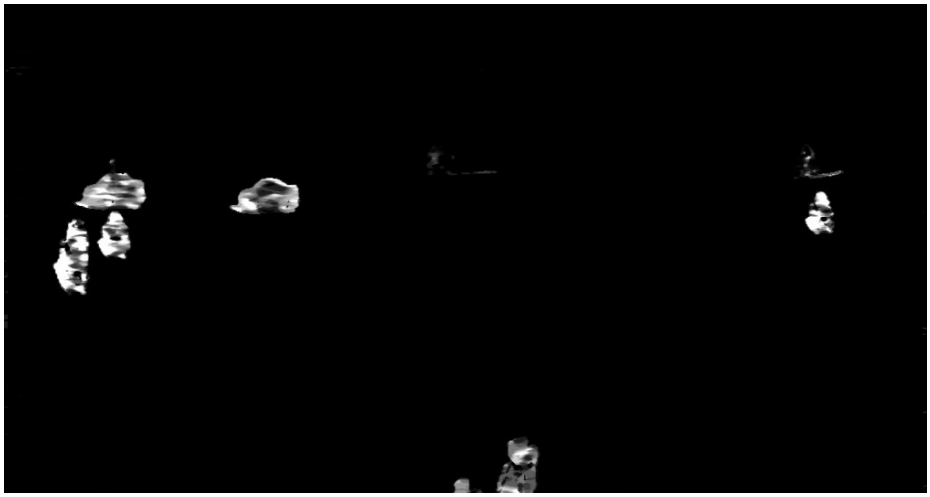
(b) The mask used to generate the velocity based temporal profile. As you can see, areas with faster motion have higher alpha values

(c) The velocity based temporal profile

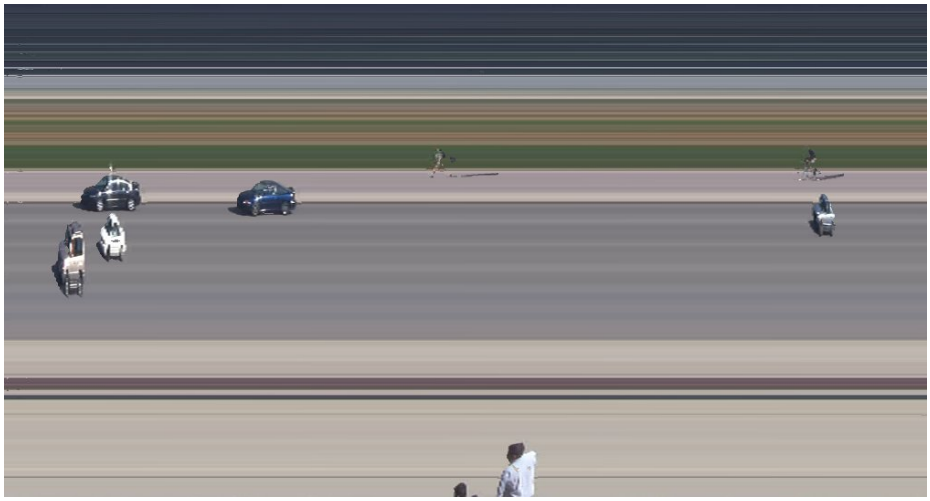
Figure 7.4. Result of our Velocity Based Temporal Profile



(a) The initial temporal slice



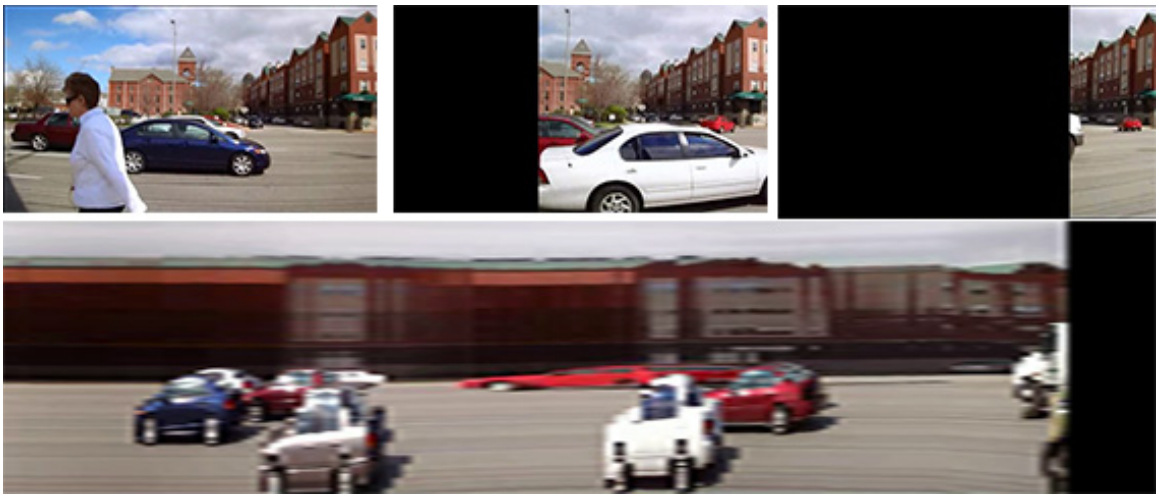
(b) The alpha mask used to generate the velocity based temporal profile. It is apparent that the vehicles have higher velocity than bicyles, and bicycles are faster than pedestrians.



(c) The velocity based temporal profile

Figure 7.5. Another example of Velocity Based Temporal Profile

(a) A panning video monitoring an intersection.



(b) A video panning a busy street.

Figure 7.6. Generating a temporal slice after correcting a panning video.

8 CONCLUSION

This study proposed three original methods to generate a temporal profile of surveillance video without time limit. The temporal profiles will not only provide an intuitive summary of surveillance video, but also have accurate time-stamp on the moving targets. Moreover, they also provide certain spatial and shape information of targets for further examination in the video frames. The image quality is compatible to the video itself. It can greatly facilitate the fast searching and retrieval targets in large databases of surveillance video. Different from spatial indexing methods, the proposed profiles are continuous in time domain without length limitation, and displays shape and spatial relationship in an inexpensive way.

REFERENCES

REFERENCES

[1] Bruno Janvier, Eric Bruno, Thierry Pun, and Stéphane Marchand-Maillet. Information-theoretic temporal segmentation of video and applications: multiscale keyframes selection and shot boundaries detection. *Multimedia Tools and Applications*, 30(3):273–288, 2006.

[2] Dan B Goldman, Brian Curless, David Salesin, and Steven M Seitz. Schematic storyboarding for video visualization and editing. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 862–871. ACM, 2006.

[3] Connelly Barnes, Dan B Goldman, Eli Shechtman, and Adam Finkelstein. Video tapestries with continuous temporal zoom. *ACM Transactions on Graphics (TOG)*, 29(4):89, 2010.

[4] Richard Szeliski. Video mosaics for virtual environments. *Computer Graphics and Applications, IEEE*, 16(2):22–30, 1996.

[5] Alex Rav-Acha, Yael Pritch, and Shmuel Peleg. Making a long video short: Dynamic video synopsis. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 435–441. IEEE, 2006.

[6] Yael Pritch, Sarit Ratovitch, Avishai Hendel, and Shmuel Peleg. Clustered synopsis of surveillance video. In *Advanced Video and Signal Based Surveillance, 2009. AVSS'09. Sixth IEEE International Conference on*, pages 195–200. IEEE, 2009.

[7] Yael Pritch, Alex Rav-Acha, Avital Gutman, and Shmuel Peleg. Webcam synopsis: Peeking around the world. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[8] Yael Pritch, Alex Rav-Acha, and Shmuel Peleg. Nonchronological video synopsis and indexing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(11):1971–1984, 2008.

[9] Alex Rav-Acha, Yael Pritch, Dani Lischinski, and Shmuel Peleg. Dynamosaicing: Mosaicing of dynamic scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(10):1789–1801, 2007.

[10] Carlos D Correa and Kwan-Liu Ma. Dynamic video narratives. *ACM Transactions on Graphics (TOG)*, 29(4):88, 2010.

[11] Jagannadan Varadarajan, Rémi Emonet, and Jean-Marc Odobez. A sequential topic model for mining recurrent activities from long term video logs. *International journal of computer vision*, 103(1):100–126, 2013.

[12] Hongyuan Cai and Jiang Yu Zheng. Video anatomy: cutting video volume for profile. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1065–1068. ACM, 2011.

[13] Jiang Yu Zheng, Hongyuan Cai, and Karthik Prabhakar. Profiling video to visual track for preview. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011.

[14] Saeid Bagheri and Jiang Yu Zheng. Localized temporal profile of surveillance video. In *Multimedia and Expo (ICME), 2014 IEEE International Conference on*, pages 1–6. IEEE, 2014.

[15] Jiang Yu Zheng, Yasaswy Bhupalam, and Hiromi T Tanaka. Understanding vehicle motion via spatial integration of intensities. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–5. IEEE, 2008.

[16] Jiang Yu Zheng and Shivank Sinha. Line cameras for monitoring and surveillance sensor networks. In *Proceedings of the 15th international conference on Multimedia*, pages 433–442. ACM, 2007.

[17] Jian Yao and Jean-Marc Odobez. Multi-layer background subtraction based on color and texture. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[18] Chris Stauffer and W. Eric L. Grimson. Learning patterns of activity using real-time tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):747–757, 2000.

[19] Saeid Bagheri and Jiang Yu Zheng. Temporal mapping of surveillance video. In *Pattern Recognition (ICPR), 2010 22nd International Conference on*, pages 1–6. IEEE, 2014.

[20] Berthold K Horn and Brian G Schunck. Determining optical flow. In *1981 Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, 1981.

[21] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis*, pages 363–370. Springer, 2003.

[22] Steven S. Beauchemin and John L. Barron. The computation of optical flow. *ACM Computing Surveys (CSUR)*, 27(3):433–466, 1995.

[23] Hongyuan Cai and Jiang Yu Zheng. Automatic heterogeneous video summarization in temporal profile. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2796–2800. IEEE, 2012.