GENERATING AND VISUALIZING SUMMARIZATIONS
OF SURVEILLANCE VIDEOS

by

Dusan Puletic
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Computer Science

Committee:

Date: 04/26/2012

Dr. Zoran Duric, Thesis Director
Dr. Jana Kosecka, Committee Member
Dr. Huzefa Rangwala, Committee Member
Dr. Sanjeev Setia, Chairman, Department of Computer Science
Dr. Lloyd J. Griffiths, Dean, Volgenau School of Engineering

Spring Semester 2012
George Mason University
Fairfax, VA
Generating and Visualizing Summarizations of Surveillance Videos

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Dusan Puletic
Engineering Diploma
University of Nis, 2009

Director: Dr. Zoran Duric, Professor
Department of Computer Science

Spring Semester 2012
George Mason University
Fairfax, VA
Dedication

This work is dedicated to my parents who instilled in me a love for learning. Also to my sister, for huge support during last two years.
Acknowledgments

Without the help of many people this thesis would not have been possible. First, I would like to express my sincere gratitude to my advisor Dr. Zoran Duric. From the beginning Dr. Duric and I had long discussions about studies, life, and work. He has helped me with advice and encouragement throughout my studies at GMU and especially in research that has led to this thesis. His guidance and support will be part of my career forever. Additionally, I would like to thank the members of my thesis committee, Dr. Jana Kosecka and Dr. Huzefa Rangwala for helping me understand and apply in my research important robotics and data mining concepts. My sincere thanks also goes to Mr. Cristopher Gettings and Mr. George DeVaux for offering me the internship opportunity at videoNEXT Network Solutions, Inc. and for giving me a chance to work on exciting projects. Finally a lifetime of gratitude to my parents Mirjana and Ivan, to my sister Nevena and brother in law Azzouz for their love and support throughout the years.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>vi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>vii</td>
</tr>
<tr>
<td>Abstract</td>
<td>viii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Related Work</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Techniques for generating summarizations</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Techniques for visualizing summarizations</td>
<td>8</td>
</tr>
<tr>
<td>3 Technical Approach</td>
<td>11</td>
</tr>
<tr>
<td>3.1 Ground Plane Calibration</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Detection and tracking</td>
<td>14</td>
</tr>
<tr>
<td>3.2.1 Foreground objects detection</td>
<td>14</td>
</tr>
<tr>
<td>3.2.2 Foreground objects tracking</td>
<td>16</td>
</tr>
<tr>
<td>3.3 Extracting information about objects in the scene</td>
<td>17</td>
</tr>
<tr>
<td>3.3.1 Calculating distances and widths of objects in the scene</td>
<td>18</td>
</tr>
<tr>
<td>3.3.2 Calculating heights of objects in the scene</td>
<td>19</td>
</tr>
<tr>
<td>3.4 Video summarization and visualization</td>
<td>23</td>
</tr>
<tr>
<td>3.5 A numerical example</td>
<td>27</td>
</tr>
<tr>
<td>4 Experiments and Discussion</td>
<td>31</td>
</tr>
<tr>
<td>5 Conclusion and Future Work</td>
<td>37</td>
</tr>
<tr>
<td>Bibliography</td>
<td>39</td>
</tr>
</tbody>
</table>
### List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>17</td>
</tr>
<tr>
<td>3.4</td>
<td>17</td>
</tr>
<tr>
<td>3.5</td>
<td>23</td>
</tr>
<tr>
<td>3.6</td>
<td>24</td>
</tr>
</tbody>
</table>
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Imaging process</td>
<td>12</td>
</tr>
<tr>
<td>3.2</td>
<td>Grid structure</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>Frame 1970</td>
<td>16</td>
</tr>
<tr>
<td>3.4</td>
<td>Frame 2000</td>
<td>16</td>
</tr>
<tr>
<td>3.5</td>
<td>Frame 2050</td>
<td>16</td>
</tr>
<tr>
<td>3.6</td>
<td>Vanishing points</td>
<td>20</td>
</tr>
<tr>
<td>3.7</td>
<td>Parallel lines in the scene</td>
<td>21</td>
</tr>
<tr>
<td>3.8</td>
<td>Summarization image for the first thousand frames</td>
<td>25</td>
</tr>
<tr>
<td>3.9</td>
<td>Summarization image with trajectories for the first thousand frames</td>
<td>25</td>
</tr>
<tr>
<td>3.10</td>
<td>Image of the scene</td>
<td>27</td>
</tr>
<tr>
<td>3.11</td>
<td>Google Earth view of the scene</td>
<td>27</td>
</tr>
<tr>
<td>3.12</td>
<td>Frame 3344 with an object shown in a red bounding box</td>
<td>28</td>
</tr>
<tr>
<td>3.13</td>
<td>Frame 3483 with an object shown in a red bounding box</td>
<td>28</td>
</tr>
<tr>
<td>3.14</td>
<td>Frame 3483 with objects shown in a red and blue bounding boxes</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Selected frames from sequence 4000-5000</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>Video synopsis of sequence 4000-5000</td>
<td>32</td>
</tr>
<tr>
<td>4.3</td>
<td>Video Synopsis for frames 1-12000</td>
<td>33</td>
</tr>
<tr>
<td>4.4</td>
<td>Video Synopsis for frames 13000-21000</td>
<td>34</td>
</tr>
<tr>
<td>4.5</td>
<td>Object classification based on speed</td>
<td>35</td>
</tr>
<tr>
<td>4.6</td>
<td>Object classification based on speed and height</td>
<td>36</td>
</tr>
</tbody>
</table>
Abstract

GENERATING AND VISUALIZING SUMMARIZATIONS OF SURVEILLANCE VIDEOS
Dusan Puletic, MS
George Mason University, 2012
Thesis Director: Dr. Zoran Duric

This thesis describes a method for efficient summarization of long surveillance videos. The method consists of four phases: ground plane calibration, detection and tracking of scene objects, extracting information about objects in the scene, generating and visualizing the summarizations. The method assumes a static camera. Both extrinsic parameters—3D position and orientation, and intrinsic parameters—focal length, principal point, lens distortion of the camera are unknown. Ground plane calibration is achieved by computing a homography [1] between the scene and corresponding location in Google Earth. Detection and tracking are based on techniques described in [2,3]. Planar homography and single view metrology [4,5] are used to calculate widths, heights, position and speed of objects in the scene. The method generates video summarization for video sequence by choosing a single image of each tracked object and overlaying it on the background image. The method chooses images of tracked objects in a way to minimize the overlap between them. For each tracked object its trajectory is shown as a sequence of vectors corresponding to object motion between successive frames. The method generates video synopsis—a summary of all activity for specified period of time. Since speeds and sizes of objects are calculated, the
method also generates sequences using various combinations of object properties.
Chapter 1: Introduction

The power of video over still images is the ability to represent movement. Deriving compact representations of video sequences that are intuitive for users and let them easily and quickly browse large collections of video data is becoming one of the most important topics in content-based video processing. Such representations—video summaries, provide the user with information about the contents of the particular sequence being examined while preserving the essential message. The need for automatic methods for generating video summaries is motivated both from the user and production viewpoints.

Video summarization is a challenging task since it requires the processing system to make decisions based on the semantic content and relative importance of the parts of the documents with respect to each other. Evaluating resulting summaries is also a problem since it is hard to derive quantitative measures of summary quality. Also, the problem is hard due to the fact that large time intervals may have no activity, or have activities that occur in a small image region. Two main approaches, proposed to address this problem are action recognition and video summarization. Action recognition methods are methods for automatic detection of activities. This kind of methods still face problems in many scenarios. The goal of video summarization is to process video sequences that contain high redundancy and make them more exciting, interesting, valuable, and useful for users. The properties of a video summary depends on the application domain, the characteristics of the sequences to be summarized, and the purpose of the summary.

This thesis describes a method for efficient summarization of surveillance videos. Millions of surveillance cameras are covering the world, capturing their field of view 24 hours a day. It is reported that in the UK alone there are 4.2 million security cameras covering city streets. Surveillance videos are largely affected with the problem of spatio-temporal redundancies. Video summarization tries to make the cameras more useful by giving a viewer the
ability to view summaries of the long video sequences, in addition to the live video stream. However, having an option to view simple summaries of the video is not sufficient in video surveillance applications.

The method described in this thesis analyzes the full motion video and detects and tracks all objects in the scene. Summarized sequences are represented by displaying active objects on the background image and marking object trajectories. Object motion is described with a vector originating at the center of the object at frame $i$ and ending at the center of the object at frame $i+1$. The method uses ground plane calibration and single view metrology [4,5], so it is possible to calculate heights, widths, areas and velocities of objects in the scene. Objects can be classified by these properties. Some examples of classification could use object size: a person, a group of people, a car, a truck; and object speed: standing, slow–moving, fast–moving, very fast–moving. Moreover, summaries can be made to show various combinations of object properties, for example slow objects of people size etc.

The remainder of this thesis is organized as follows. Chapter 2 presents a discussion on related literature. Chapter 3 outlines the technical approach used in the method. Chapter 4 presents experimental results and some analysis on these results. Finally, chapter 5 presents conclusions and future work.
Two main criteria can be used to classify research on video summarization. First, it can be classified based on techniques used for generating summarizations (see Sec. 2.1). It can also be classified based on techniques used for visualizing summarized sequences (see Sec. 2.2).

2.1 Techniques for generating summarizations

The taxonomy presented in this section was adapted from [6]. The techniques for generating summarizations can be roughly divided into following groups:

- Speedup of playback
- Techniques based on frame clustering
- Techniques based on frame clustering by dimensionality reduction
- Techniques based on domain knowledge
- Techniques based on closed-captions or speech transcripts
- Techniques based on multiple information streams

In the remainder of this section, these groups of techniques are described in some details.

**Speedup of playback** is a simple way to compactly display a video sequence. A technique known as time scale modification can be used to process the audio signal so that the speedup can be made with little distortion [7]. The compression allowed by this approach, however, is limited to a summarization ratio (SR) of 1.5 – 2.5, depending on the particular program genre [8]. However, this SR value is not adequate for most summarization applications.
Techniques based on frame clustering are recognized as one of the simplest and most compact ways of representing video sequences. These techniques generally focus on the image data stream only. Color histograms, because of their robustness, have frequently been used as features for clustering. One of the earliest work in this area is by Yeung and Yeo [9] using time-constrained clustering of shots. Each shot is then labeled according to the cluster to which it belongs, and three types of events, dialogue, action, and other, are detected based on these labels. Representative frames from each event are then selected for the summary. The Video Manga system by Uchihashi et al. [10] clusters individual video frames using YUV color histograms. Iacob, Lagendijk, and Iacob [11] propose a similar technique. However, in their approach video frames are first divided into rectangles, whose sizes depend on the local structure, and YUV histograms are extracted from these rectangles. Ratakonda, Sezan, and Crinon [12] extract key-frames for summaries based on the area under the cumulative action curve within a shot, where the action between two frames is defined to be the absolute histogram difference between color histograms of the frames. They then cluster these key-frames into a hierarchical structure to generate a summary of the program. Ferman and Tekalp [13] select key-frames from each shot using the fuzzy c-clustering algorithm, which is a variation of the k-means clustering method. Cluster validity analysis is performed to automatically determine the optimal number of key-frames from each shot to be included in the summary. This summary may then be processed based on user preferences, such as the maximum number of key-frames to view, and cluster merging may be performed if there are too many key-frames in the original summary. The approaches proposed in [14,15] both contain a two-stage clustering structure, which is very similar to the method used in [13] but instead of performing shot detection segments are identified by clustering of video frames. Hanjalic and Zhang use features and cluster validity analysis techniques that are similar to those in [13]. Farin, Effelsberg, and de With [15] propose a two-stage clustering technique based on luminance histograms extracted from each frame in the sequence. First, in an approach similar to time-constrained clustering, segments in the video sequence are located by minimizing segment inhomogeneity, which
is defined as the sum of the distances of all frames within a segment to the mean feature vector of the segment. Then, the segments obtained are clustered using the Earth-Mover’s distance \[16\]. Yahiaoui, Merialdo, and Huet \[17\] first cluster frames based on the L1 distance between their color histograms using a procedure similar to k-means clustering. Then, a set of clusters is chosen as to maximize the coverage over the source video sequence.

**Techniques based on frame clustering by dimensionality reduction** perform a bottom-up clustering of the video frames selected at fixed intervals. A high dimensional feature vector is extracted from each frame. This dimensionality is then reduced either by projecting the vectors to a much lower dimensional space \[18, 19\] or by using local approximations to high dimensional trajectories \[20, 21\]. Finally, clustering of frames is performed in this lower dimensional space. DeMenthon, Kobla, and Doerman \[20\] extract a 37-dimensional feature vector from each frame by considering a time coordinate together with the three coordinates of the largest blobs in four intervals for each luminance and chrominance channel. They then apply a curve splitting algorithm to the trajectory of these feature vectors to segment the video sequence. A key-frame is extracted from each segment. Stefanidis et al. \[21\] propose a similar system. However, they split the three-dimensional trajectories of video objects instead of feature trajectories. Gong and Liu \[18\] use singular value decomposition (SVD) to cluster frames evenly spaced in the video sequence. Each frame is initially represented using three-dimensional RGB histograms, which results in 1125-dimensional frame feature vectors. Then, SVD is performed on these vectors to reduce the dimensionality to 150 and clustering is performed in this space. Portions of shots from each cluster are selected for the summary. Cooper and Foote \[19\] sample the given video sequence at a rate of one frame per second and extract a color feature vector from each extracted frame. The cosine of the angle between feature vectors is taken to be the similarity measure between them and a non-negative similarity matrix is formed between all pairs of frames. Non-negative matrix factorization (NMF) \[22\], is used to reduce the dimensionality of the similarity matrix. NMF is a linear approximation similar to SVD, the difference being the fact that the basis vectors are non-negative.
Techniques based on domain knowledge enhance summarization algorithms by exploiting domain-specific knowledge about events. Summarization of sports video has been the main application for such approaches. Sports programs lend themselves well for automatic summarization for a number of reasons. First, the interesting segments of a program occupy a small portion of the whole content. Second, the broadcast value of a program falls off rapidly after the event so the processing must be performed in near real-time. Third, compact representations of sports programs have a large potential audience. Finally, often there are clear markers, such as cheering crowds, stopped games, and replays, that signify important events. Li, Pan, and Sezan [23] develop a general model for sports programs where events are defined to be the actions in a program that are replayed by the broadcaster. Ekin and Tekalp [24] divide each key-frame of a soccer program into 9 parts and use features based the color content to classify shots into long, medium, and close-up shots. They also detect shots containing the referee and the penalty box. Goal detection is performed similar to [23] by detecting close-up shots followed by a replay. Cabasson and Divakaran [25] detect audio peaks and a motion activity measure to detect exciting events in soccer programs. Based on the heuristic that the game generally stops after an exciting event, they search the program for sequences of high motion followed by very little motion. If an audio peak is detected near such a sequence it is marked as an event and included in the summary.

Techniques based on closed-captions or speech transcripts are useful for some types of videos, like news programs, presentation videos, documentaries. In these videos, a large portion of the informational content is carried in the audio. Using the spoken text to generate video summaries becomes a powerful approach for these types of sequences. Content text is readily available for most broadcast programs in the form of closed captions. For sequences, like presentations and instructional programs, where this information is not available, speech recognition may be performed to obtain the speech transcript. Once the text corresponding to a video sequence is available, one can use methods of text summarization to obtain a text summary. The portions of the video corresponding to the selected
text may then be concatenated to generate the video skim. Processing text also provides a high level of access to the semantic content of a sequence that is not possible to achieve using the image content only. Agnihotri et al. [26] search the closed-caption text for cue words to generate summaries for talk shows. Cues such as "please welcome" and "when we come back" in addition to domain knowledge about program structure are used to segment the programs into parts containing individual guests and commercial breaks. Keywords are then used to categorize the conversation with each guest into a number of predetermined classes such as movie or music. In their ANSES system Pickering, Wong, and Rueger [27] use key entity detection to identify important keywords in closed-caption text. Working under the assumption that story boundaries always fall on shot boundaries, they perform shot detection followed by the merging of similar shots based on the similarity of words they contain. They then detect the nouns in text using a part of speech tagger and use lexical chains [28] to rank the sentences in each story. The highest scoring sentences are then used to summarize each news story. An example of a technique that uses automatic speech recognition (ASR) is the one proposed by Taskiran et al. [29]. The usage of ASR makes their method applicable to cases where the closed-caption text is not available, such as presentations or instructional videos. In their approach the video is first divided into segments at the pause boundaries. Then, each segment is assigned a score using term frequencies within segments. Using statistical text analysis, dominant word pairs are identified in the program and the scores of segments containing these pairs are increased. The segments with highest scores are selected for the summary.

**Techniques based on multiple information streams** combine data from images, audio, and closed-caption text. The MoCA project [30], one of the earliest systems for video summarization, uses color and action content of shots, among other heuristics, to obtain trailers for feature films. The Informedia project constitutes a pioneer and one of the largest efforts in creating a large video database with search and browse capabilities. It uses integrated speech recognition, image processing, and natural language processing techniques for the analysis of video data [31, 32]. Video segments with significant camera
motion, and those showing people or a text caption, are given a higher score. Audio analysis includes detection of names in the speech transcript. Audio and video segments selected for summary are then merged while trying to maintain audio/video synchronicity. Ma et al. [33] propose a generic user attention model by integrating a set of low-level features extracted from video. This model incorporates features based on camera and object motion, face detection, and audio. An attention value curve is obtained for a given video sequence using the model and portions near the crests of this curve are deemed to be interesting events. Then, heuristic rules are employed, based on pause and shot boundaries, and the SR, to select portions of the video for the summary.

2.2 Techniques for visualizing summarizations

The techniques for visualizing summarizations can be roughly divided into following groups:

- Series of still images (key frames)
- Montage of still images (synopsis mosaics and dynamic stills)
- Collection of short clips (video skimming)
- Montage of moving images (webcam synopsis)

In the remainder of this section, these groups of techniques are described in some details.

**Series of still images** represents simplest static visualization method. Key-frame selection can be shot boundary based. Video is divided into segments (shots). Current segment ends and new segment begins if difference of one or more features is greater than threshold. Features can be pixel values [34], color/grayscale histograms [35], edges [36]. Each segment is represented by one or more frames. Segment can be represented by first, middle and last frame [37], or further segmentation can be done inside each segment [38]. Key-frame selection can also be perceptual feature based. One option is to do color based selection [39]. Color space is quantized into N cells (e.g. 64) and selection is based on distance between histograms. Color based selection may not be enough given significant
motion, so [39,40] use motion metric based on optical flow. In this case key-frame is selected as a minimum point between two local maxima where difference exceeds threshold. Object-based key-frame selection [41] starts with video object extraction and connected components labeling. If number of regions is changed, key-frame is selected by event. If number of regions is not changed, feature extraction is performed and key-frame is selected by action (shape change of video object). In feature vector space based key frame selection [20, 42], frame is represented as point in multi-dimensional feature space. Entire video is curve in same space and key frames are selected based on curve properties (sharp corners, direction change, etc.).

**Montage of still images** is visualization method which can be achieved using synopsis mosaics or dynamic stills. Synopsis mosaics [43,44] are made by selecting or sampling key-frames and computing affine transformations between successive frames. After one frame is selected as reference frame, other frames are projected into plane of reference coordinate system. Median of all pixels mapped to same location is used for background. Advantage of this approach is that it combines key-frames in single shot and it can recreate full background when occluded by moving objects. Disadvantage is that it may require manual key-frame selection to get complete background and also, moving objects may not display well. Dynamic stills [45] are made by selecting or sampling key-frames and performing object segmentation on selected frames. Final image is made by adding different segments of objects to the background. Advantages of this approach are good sense of motion and good screen usage. Disadvantages are that single image is limited in complexity and exact spatial information is lost.

**Video skimming** methods can be used for displaying highlight sequences. Displaying highlight sequences is important in movie previews [30]. Video is segmented and segments with the highest scores are selected and concatenated to generate a video skim. Various factors can be used for selecting portions of the source video to be included in the video skim. Factors can be: scene boundary detection [46, 47], high contrast scenes, high motion scenes, scenes with average color composition, scenes with dialog. Video skimming
is also used in model based summarization [48]. Highlight sequences are also displayed in model based summarization, which is used for summarization of football broadcasts. Model video represents sequence of plays with removed non-play footage and selected most important/exciting plays. Start-of-play detection is achieved by detecting field color, field lines, team jersey colors, player line-ups. End-of-play is detected when camera breaks after start of play. Video skims are also used for summarization of the full content of video. Summarized video is produced using techniques described earlier in this chapter: Speedup of playback, Techniques based on domain knowledge, Approaches using multiple information streams.

**Webcam synopsis** methods [49,50] are used to display short (couple of minutes) videos of complete activity recorded during much longer (couple of hours or more) period of time. For each frame, median background image over surrounding four-minute stretch is computed. Moving objects are detected using background subtraction. Connected components are found to get the object tubes. Method tries to find the best synopsis, optimizing the activity, background consistency, collision, and temporal consistency costs. A synopsis is a mapping, for each tube b, from its original time extent to a shifted extent. In order to get optimal solution, energy cost of the synopsis should be minimized. Energy cost is the summation of activity cost of a tube, background consistency of a tube, collision cost between two tubes and temporal consistency cost between two tubes. Advantages of this method are: it does efficient compression of long videos, there is user-controllable compression threshold, user can select and see activity of specific types of objects. Disadvantages are that it cannot support event-based selection and cannot handle videos with unpredictable background shift.
Chapter 3: Technical Approach

The method consists of four phases: ground plane calibration, detection and tracking of scene objects, extracting information about objects in the scene, producing and visualizing summarized sequences. The method assumes single, static camera with both extrinsic parameters — 3D position and orientation, and intrinsic parameters — focal length, principal point, lens distortion unknown. Ground plane calibration is described in Sec. 3.1. Sec. 3.2 describes detection and tracking. Extracting information about objects in the scene is described in Sec. 3.3. Finally, Sec. 3.4 describes producing and visualizing summarized sequences.

3.1 Ground Plane Calibration

Calibration is the primary step of many vision applications as it allows systems to determine a relationship between what appears on an image and where it is located in the world. In order to detect and track scene objects and to produce and visualize summarization sequences it is important to estimate correspondence between points in the image and points in the world plane. The image coordinates are in pixels and corresponding points in the world plane are represented by GEO locations [find a reference for GEO locations].

With an assumption that all pixels are on the ground, the image plane can be transformed directly to the physical planes using plane to plane homography [1]. A homography is described by a $3 \times 3$ matrix $H$. Once the matrix is determined, the back-projection of an image point to a point on a world plane is straightforward. The distance between two points on the world plane is simply computed from the distance between their back-projected images.
Points on the world plane are represented by upper case vectors, \( \vec{X} \), and their corresponding images are represented by lower case vectors \( \vec{x} \), where \( \vec{X} \) and \( \vec{x} \) are homogeneous 3-vectors, \( \vec{X} = (X, Y, 1)^T \) and \( \vec{x} = (x, y, 1)^T \). Under perspective projection corresponding points are related by:

\[
\vec{X} = H\vec{x}
\]  

\[
\begin{pmatrix}
X \\
Y \\
1
\end{pmatrix} =
\begin{pmatrix}
h_{11} & h_{12} & h_{13} \\
h_{21} & h_{22} & h_{23} \\
h_{31} & h_{32} & h_{33}
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
1
\end{pmatrix}
\]  

(3.2)

Dividing the first row of Eq. (3.2) by the third row and the second row by the third row we get the following two equations:

\[
h_{11}x + h_{12}y + h_{13} - h_{31}xX - h_{32}yX - h_{33}X = 0
\]  

(3.3)
\[ h_{21}x + h_{22}y + h_{23} - h_{31}xY - h_{32}yY - h_{33}Y = 0 \]  

(3.4)

Eqs. (3.3) and (3.4) can be written in matrix form as:

\[ A_i h = 0 \]  

(3.5)

where

\[
A_i = \begin{pmatrix}
-x & -y & -1 & 0 & 0 & 0 & Xx & Xy & X \\
0 & 0 & 0 & -x & -y & -1 & Yx & Yy & Y
\end{pmatrix}
\]

and

\[
h = \begin{pmatrix}
h_{11} \\
h_{12} \\
h_{13} \\
h_{21} \\
h_{22} \\
h_{23} \\
h_{31} \\
h_{32} \\
h_{33}
\end{pmatrix}^T
\]

Since each point correspondence provides 2 equations, 4 correspondences are sufficient to solve for the 8 degrees of freedom of \( H \). The restriction is that any 3 points cannot be collinear. Four \( 2 \times 9 \) matrices \( A_i \) (one per point correspondence) can be stacked on top of one another to get a single \( 8 \times 9 \) matrix \( A \). Solution for \( h \) can be found by using singular value decomposition (SVD) [51] of \( A \). \( A = USV^T \), where \( h \) is given by the last column of \( V \), i.e. the right singular vector of \( A \) corresponding to the smallest singular value of \( S \).

Since the location of the scene is known, for four features in the scene corresponding features are found using Google Earth.

Once the parameters of matrix \( H \) are obtained, GEO location of any pixel from the scene can be calculated by solving Eq. (3.3) and Eq. (3.4) for \( X \) and \( Y \):

\[
X = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}
\]  

(3.6)

\[
Y = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}
\]  

(3.7)

Numeric example of application is shown in Sec. 3.5.
3.2 Detection and tracking

Locating foreground objects and tracking them through the sequence of frames is essential part of the method. Algorithms for summarization and visualization use result of the tracker as the input data. Result of the tracker is data structure which contains information about every detected object for each frame in the sequence. Each detected object is represented by following parameters:

<table>
<thead>
<tr>
<th>Table 3.1: Output of the tracker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object ID</td>
</tr>
<tr>
<td>Location of the object</td>
</tr>
<tr>
<td>Height of the object’s bounding box</td>
</tr>
<tr>
<td>Width of the object’s bounding box</td>
</tr>
<tr>
<td>Number of foreground pixels inside the box</td>
</tr>
</tbody>
</table>

Detection and tracking are based on techniques described in [2, 3].

3.2.1 Foreground objects detection

A primary motivation for choosing technique described in [3] is the fact that representing the background by the mean image is not effective in many applications. A fixed camera will typically experience vibrations. The background, even if fixed, will move slightly: leaves will flutter, waves will shimmer, and distant objects will move a little. Another motivation is that a fixed camera will over time see a change in the fixed background: lights will go on and off, the sun’s shadows will move, and the color palette will drift. Chosen technique addresses these concerns by using two ideas: use of clustering techniques and use of streaming model of algorithm design.

The goal of the foreground objects detection is to identify regions of a frame that contain moving objects. Initially, a background model is built using the sequence of frames that contains a typical background. After the initial phase, as new frames are encountered they are compared with the background model. The areas in which the difference between foreground and background is large are marked as foreground. Model is constructed by
overlying the grid (see Fig.(3.2)) on top of each frame. The size of the cells in the grid is 4 × 4. Fig. 3.2 illustrates the overlapping grid structure used in the method. This structure allows for the possibility of detecting and tracking parts of foreground objects when they appear in the corner of a cell.

![Grid structure](image)

Figure 3.2: Grid structure

Clustering is done on color values in each cell. Each 4×4 cell is represented by 16 r, g, and b pixel values in that cell. All triples (r, g, b) are regarded as points in three dimensional space. They are clustered with the Euclidean distance as the similarity measure. The constant, $k_{\text{max}}$, is the maximum number of clusters that is maintained for each cell. $T_{\text{max}}$ is a threshold that is used to determine if additional clusters should be created. Algorithm starts by collecting $n$ color values from several frames for each cell. A value of $n$ used in the method is 64. A single cluster is created for each cell and if the distance of the mean and any outlier is greater than $T_{\text{max}}$, k-Means Algorithm with $k = 2$ is used to cluster the color values. If there are any outliers with the distance greater than $T_{\text{max}}$ from these centers k-Means with $k = 3$ is used, otherwise the results are returned. The mean and the number of color values supporting it is computed for each center. For each color value in each cell its nearest cluster center is determined. If it is sufficiently close, below a given threshold ($T_{\text{max}}$), it is judged to be the same as the prior background. Otherwise it is flagged as potentially part of the foreground. However, it is also introduced as a new cluster center.
(since it might be the beginning of new equivalence class of background in that cell).

### 3.2.2 Foreground objects tracking

Foreground objects tracking is based on the method described in [2]. Tracking of regions is performed. Regions are connected components that have been tracked for at least $T_{fr}$ frames. Each region has its location, bounding box, a supportmap (mask), a timestamp and a tracking status. A list of tracked regions is maintained during tracking. Matching is performed by finding regions with overlapping bounding boxes and with similar area. In each frame, a new region tracker is initialized for each novel region, if any. Regions with no match are deleted. Regions can split and merge. When a region splits, all the resulting regions inherit their parent’s timestamp and status. When regions merge, the timestamp and status are inherited from the oldest parent region. Once a region is tracked for three frames, it is considered to be reliable.

Figs. 3.3, 3.4 and 3.5 show the object with ID=116 as it is tracked through the sequence of images. Position of the upper left corner of the bounding box represents object location. Tabs. 3.2, 3.3 and 3.4 show tracker results for frames 1970, 2000 and 2050.

![Figure 3.3: Frame 1970](image1.png) ![Figure 3.4: Frame 2000](image2.png) ![Figure 3.5: Frame 2050](image3.png)
3.3 Extracting information about objects in the scene

Information about the physical sizes of detected objects is important issue in video surveillance. Such information is helpful in tracking and detection of objects and human understanding of the scene. Because of perspective effects on the image, determining physical sizes of objects is hard. In one portion of the image, a pixel may correspond to a distance of only a few centimeters. In other portions, a pixel may correspond to a distance of 50 centimeters or more.

Since the intrinsic as well as the extrinsic parameters of the camera are not known, 3D geometric information about scene can be recovered using single view metrology [4, 5]. Techniques introduced in Criminisi’s work describe simple and effective algorithms for extracting 3D information, given only minimal information determined from the image. This minimal information is: the vanishing line of a reference plane, the vanishing point for a direction not parallel to the plane and reference height in the image. On the other hand, scene constraints such as orthogonality and parallelism of structures are exploited, thus making the algorithms suitable for scenes containing man-made structures such as architectural elements and geometric patterns.

Single view metrology is used to measure lengths of segments on planar surfaces (widths of scene objects) and distances of points from planar surfaces (heights of scene objects).
3.3.1 Calculating distances and widths of objects in the scene

Distance in the scene and width of the scene object on the ground can be estimated by computing GEO locations of two corresponding points and calculating the distance between these GEO locations. Pixel values of the chosen points are converted into their GEO locations using plane to plane homography. Distance between GEO locations can be calculated using Haversine formula [52]. Haversine formula is an equation which calculates great-circle distances between two points on a sphere using their GEO locations (longitudes and latitudes).

\[
a = \sin^2(\Delta \text{lat}/2) + \cos(\text{lat}_1) \cdot \cos(\text{lat}_2) \cdot \sin^2(\Delta \text{long}/2) \tag{3.8}
\]

\[
c = 2 \cdot \arctan(\sqrt{a}, \sqrt{1 - a}) \tag{3.9}
\]

\[
\text{dist} = R \cdot c \tag{3.10}
\]

where \(\text{lat}_1\) and \(\text{long}_1\) represent latitude and longitude of the first point, \(\text{lat}_2\) and \(\text{long}_2\) are latitude and longitude of the second point, \(R\) is earth’s radius (mean radius = 6,371km) and \(\text{dist}\) is calculated distance.

Calculation of distances and widths of objects in the scene is described in Algorithm 1.

**Algorithm 1** Calculating distances and widths of objects in the scene

1. Estimate the plane-to-plane homography matrix \(H\);

2. Repeat

   (a) Select two image points \(x_1\) and \(x_2\) from the scene;

   (b) Back-project each image point into the world plane using Eq. (3.1) to obtain two world points \(X_1\) and \(X_2\);

   (c) Compute the Haversine distance \(d(X_1, X_2)\).
3.3.2 Calculating heights of objects in the scene

In order to compute the height of an object in the scene, it is necessary to compute the vanishing line of a reference plane and the vanishing point for a direction not parallel to the plane. Also, it is necessary to have reference height in the image.

*Computing the Vanishing Point:* Computing the vanishing point is based on the assumption that there exist groups of parallel straight structures in the scene, and that these structures produce line segments in the image. According to the laws of perspective, such a group of image line segments, when extended, will intersect in the common vanishing point (see Fig. 3.6). This point has the following interpretation: it is the projection of the intersection of the parallel lines at infinity. Vanishing points can be computed automatically by detecting straight lines in the image using image derivatives and non-maximum suppression, quantizing the gradient orientation into a set of bins containing pixels with similar gradient orientations, performing connected component analysis within each bin and line fitting [53]. Lines which represent a physical edge in the world might appear broken in the image because of occlusions. Merging aligned edges to create longer ones increases the accuracy of their location and orientation. Since the images of parallel world lines intersect each other in the same vanishing point, the point is defined by at least two such lines. However, if more than two lines are available, a Linear Least Square or a Maximum Likelihood Estimate algorithm can be used to estimate the point [54].

*Computing the Vanishing Line:* Images of lines parallel to each other and to a plane intersect in points on the plane vanishing line. Therefore two sets of lines with different directions are sufficient to define the plane vanishing line. If more than two orientations are available then the computation of the vanishing line can be done using Linear Least Squares algorithm, Maximum Likelihood algorithm [54], or Expectation Maximization algorithm [55].

In the video used for this thesis, number of parallel lines that can be used for vanishing point calculations is small. There are two sets of parallel lines with five lines in one set and three lines in other (see Fig. 3.7). Parallel lines were selected manually. Intersection for
each pair of parallel lines is computed and exact location of vanishing points is determined using Linear Least Squares method. Since there are parallel lines in only two directions in the scene, vanishing line is calculated as a line that contains both vanishing points.

If \( v \) is the vanishing point for the vertical direction, \( l \) is the vanishing line of the ground plane, \( t_r \) and \( b_r \) are the top and base points of the reference, respectively and \( t_x \) and \( b_x \) are the top and base points of the object to be measured, then the following holds:

\[
\alpha Z_i = -\frac{\|\vec{b}_i \times \vec{t}_i\|}{Z_r (\vec{l} \cdot \vec{b}_i) \|\vec{v} \times \vec{t}_i\|} \quad i \in \{r, x\}
\]

(3.11)

where \( Z_x \) is the height of the object which is measured, \( Z_r \) is the height of the reference object and \( \alpha \) is a scalar quantity herein referred to as metric factor. Since \( \alpha Z_r \) scales linearly, affine structure is obtained. If \( \alpha \) is known, then a metric value for the height \( Z \) is obtained. Conversely, if the height \( Z \) is known, there is a way of computing \( \alpha \) and hence removing the affine ambiguity.

The heights of objects in the scene are calculated using Algorithm 2.

As a consequence of the camera orientation and relatively low resolution of video used
for this thesis (640 × 480), parallel lines used for the calculation of vertical vanishing point are parallel in the image. Therefore, an accurate position of the vertical vanishing point cannot be found. However, using the fact that Y-coordinate of the vanishing point is very large it can be shown that it is not necessary to know its value in order to calculate heights of objects in the scene with satisfactory accuracy:

If \( \vec{v} = (v_x, v_y, 1), \vec{t_r} = (t_{rx}, t_{ry}, 1) \), from Eq. (3.12):

\[
\vec{v} \times \vec{t_r} = \vec{i}(v_y - t_{ry}) + \vec{j}(t_{rx} - v_x) + \vec{k}(v_x t_{ry} - v_y t_{rx})
\]

(3.14)

\[
\|\vec{v} \times \vec{t_r}\| = \sqrt{(v_y - t_{ry})^2 + (t_{rx} - v_x)^2 + (v_x t_{ry} - v_y t_{rx})^2}
\]

(3.15)

Given that \( v_y \gg v_x, v_y \gg t_{rx}, v_y \gg t_{ry} \):

\[
\|\vec{v} \times \vec{t_r}\| \approx \sqrt{v_y^2 + v_y^2 t_{rx}^2} = v_y \sqrt{1 + t_{rx}^2}
\]

(3.16)
Algorithm 2 Calculating heights of objects in the scene

1. Estimate the vanishing point $v$ for the vertical direction;

2. Estimate the vanishing line $l$ of the reference plane;

3. Select top and base points of the reference segment (points $t_r$ and $b_r$);

4. Compute the metric factor $\alpha$ by applying:

$$\alpha = \frac{-\|\vec{b}_r \times \vec{t}_r\|}{Z_r (l \cdot \vec{b}_r) \|\vec{v} \times \vec{t}_r\|};$$ (3.12)

5. Repeat

(a) Select top and base of the object to measure (points $t_x$ and $b_x$);

(b) Compute the height $Z_x$ by applying:

$$Z_x = \frac{-\|\vec{b}_x \times \vec{t}_x\|}{\alpha (l \cdot \vec{b}_x) \|\vec{v} \times \vec{t}_x\|};$$ (3.13)

Similarly, if $\vec{t}_x = (t_{xx}, t_{xy}, 1)$ and $v_y \gg v_x$, $v_y \gg t_{xx}$, $v_y \gg t_{xy}$, from Eq. (3.13):

$$\|\vec{v} \times \vec{t}_x\| \approx \sqrt{v_y^2 + v_y^2 t_{xx}^2} = v_y \sqrt{1 + t_{xx}^2}$$ (3.17)

Changing Eq. (3.16) and Eq. (3.17) in Eq. (3.13):

$$Z_x \approx -\frac{\|\vec{b}_x \times \vec{t}_x\|}{Z_r (l \cdot \vec{b}_x) v_y \sqrt{1 + t_{xx}^2}} \frac{(l \cdot \vec{b}_x) v_y \sqrt{1 + t_{xx}^2}}{\|\vec{b}_r \times \vec{t}_r\| (l \cdot \vec{b}_x) \sqrt{1 + t_{xx}^2}};$$ (3.18)

It is clear from Eq. (3.18) that value $Z_x$ does not depend on $v_y$. 

22
3.4 Video summarization and visualization

This section describes techniques and algorithms used for producing and visualizing summarized sequences. Method uses the output of the tracker, described in previous section. Output of the tracker is data structure which contains information about every detected object for each frame in the sequence. Each detected object in each frame is represented by parameters from Tab. 3.1.

The method modifies output of the tracker and creates two lists: $M$ and $M^\ast$. Objects which appear in fewer than $min$ frames are not shown. $min = 50$ was used in this example.

$M$ is a list which stores parameters about every tracked object for every frame in which it is present. Element of list $M$ have a structure shown in Tab. 3.5.

<table>
<thead>
<tr>
<th>Frame number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location of the object in X direction</td>
</tr>
<tr>
<td>Location of the object in Y direction</td>
</tr>
<tr>
<td>Width of the object’s bounding box</td>
</tr>
<tr>
<td>Height of the object’s bounding box</td>
</tr>
<tr>
<td>Foreground/background ratio inside the box</td>
</tr>
</tbody>
</table>

$M^\ast$ is a list which contains information about every tracked object. It contains average width, height, area and speed values for every object. These values are used for producing and visualizing summarization sequences filtered by a number of different rules (size, speed, area). Every element of list $M^\ast$ has the structure shown in Tab. 3.6.

Before calculating average value, all the values that make no sense (outliers) are removed using boxplot technique. Values are sorted from the smallest to the largest. The bottom and top of the box are the 25th and 75th percentile (the lower and upper quartiles, respectively), and the middle quartile is the median value. If any value is 1.5 times or more smaller than the lower quartile or 1.5 times or more larger than the higher quartile, it is declared as the outlier and removed from further calculations.
Table 3.6: An element of list $\mathcal{M}^*$

<table>
<thead>
<tr>
<th>Description</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average width of the object’s bounding box in pixels</td>
<td>Summing up widths and dividing by number of frames</td>
</tr>
<tr>
<td>Average width of the object’s bounding box in real world</td>
<td>After applying techniques described in Sec. 3.3.1 and Sec. 3.3.2</td>
</tr>
<tr>
<td>Average height of the object’s bounding box in pixels</td>
<td>Summing up heights and dividing by number of frames</td>
</tr>
<tr>
<td>Average height of the object’s bounding box in real world</td>
<td>After applying techniques described in Sec. 3.3.1 and Sec. 3.3.2</td>
</tr>
<tr>
<td>Average area of the object’s bounding box in real world</td>
<td>Product of average height and width</td>
</tr>
<tr>
<td>Average speed of the object</td>
<td>Summing up velocities calculated on ten frame intervals and dividing by number of intervals</td>
</tr>
</tbody>
</table>

Average width and height in pixels of the object’s bounding box are calculated by summing up widths and heights of the object’s bounding box from every frame in which the object is present and dividing these sums with number of frames. Average real world width and height of the object’s bounding box are calculated in the same way, but after applying techniques described in Sec. 3.3.1 and Sec. 3.3.2. Average area of the object’s bounding box is the product of average height and width. Average speed of the object is calculated by summing up velocities calculated on the ten frame intervals (first and eleventh, second and twelfth, third and thirteenth etc.) and dividing by the number of intervals. Speed for each interval is calculated by dividing the distance calculated using equations from Sec. 3.3.1 with time elapsed. Since the video used in this thesis has 10 frames per second, distance is divided by $10 \times 0.1 \text{ second} = 1 \text{ second}$.

Video summarization is produced by placing foreground objects and adding their trajectories on the corresponding background image. It is the process which can be described in two steps. The first step produces the summarization image by adding foreground objects to the chosen background image. The second step calculates and adds object’s trajectories to the summarization image.

Creating the summarization image begins with choosing the right background image. Background images are created and saved for every 450 frames (45 seconds) by calculating the mean for each pixel. Having various background images is necessary because of the possible changes in lighting in the scene or situations where some objects become part of
the background. Choice of background image depends on the sequence $SE$ for which the video is summarized. If starting frame of the sequence is $S$ and ending frame is $E$, then the background image for that sequence will be the one closest to the frame $\frac{S + E}{2}$. After choosing the right background image, algorithm finds objects which completely or partially appear in $SE$. It is enough for objects to appear in single frame in $SE$ in order to be selected. Once the background image and objects are selected, algorithm solves the problem of finding the best way of presenting all the objects on the background. In solving this problem algorithm uses greedy approach. The idea is to find the best frame for every object and to put the object on the resulting image using object’s properties from that frame. The best frame is the one for which the object’s bounding box has the smallest overlap with all other objects already put on the resulting image. In the ideal situation, the overlap would always be 0. In scenarios where there are lot of objects and their trajectories are similar, some amount of overlap is inevitable. Each object is put on the resulting image by copying content of its bounding box from the chosen frame to the same location on the resulting image. After putting object on the resulting image, algorithm draws black rectangle around it which makes it more visible. The whole process of creating the summarization image is given in
Algorithm 3 and example of summarization image is given in Fig. 3.8.

**Algorithm 3 Creating the summarization image**

**INPUT:** sequence SE, structure M, structure M^*

**OUTPUT:** summarization image

Choose background image based on the middle frame in the sequence;
Choose objects which completely or partially appear in SE
Construct resulting image

for every object do
  for every frame in which object exist do
    Calculate the overlap of the object’s bounding box with content of the overlap image;
    end for
  Save the frame index with the minimum overlap;
  Update the overlap image
end for

for every object do
  Copy an object from the frame determined by a frame index on a resulting image;
  Draw a rectangle around its bounding box;
  Update resulting image;
end for

Once the summarization image is created, next step is to add trajectories of the foreground objects. Adding trajectories allows user to get information about motion of objects in the scene. Algorithm 4 chooses foreground objects in the same way as Algorithm 3. For every chosen object algorithm draws a vector from the center of its bounding box in current frame to the center of its bounding box in the next frame. Vectors drawn for one object represent its trajectory. Color of the vector can be blue, red or green. Different colors help user to get information about object’s behavior before and after the specified sequence. There is a possibility that some objects appear before starting frame S or stay active after the ending frame E. Part of the object’s trajectory related to frames before S is red. Part of the object’s trajectory related to frames before S is red. Part of the object’s trajectory related to frames between S and F is blue. Part of the object’s trajectory related to frames after E is green.

Algorithms 3 and 4 are used to create video synopsis - summary of all activity for specified period of time. They are modified in order to get different kinds of summarization. Due to the content of structure M^* objects are classified based on size, speed, area or trajectory location. As a consequence, summarization method gives answers to various queries. It shows summaries of particular types of objects, or objects with particular behavior. Results
Algorithm 4 Creating object trajectories

INPUT: sequence $\overline{SE}$, structure $\mathcal{M}$, structure $\mathcal{M}^*$, summarization image

OUTPUT: summarization image with objects trajectories

Choose objects which completely or partially appear in $\overline{SE}$

for every object do
    for every frame do
        if $((frame \geq S) \land (frame \leq E))$ then
            draw blue vector between object location at current and successive frame
        end if
        if $(frame < S)$ then
            draw red vector between object location at current and successive frame
        end if
        if $(frame > E)$ then
            draw green vector between object location at current and successive frame
        end if
    end for
end for

of different queries are shown and discussed in the next chapter.

3.5 A numerical example

Figure 3.10: Image of the scene

Figure 3.11: Google Earth view of the scene

Four corresponding points from Fig. 3.10 and Fig. 3.11 have the following coordinates:
Using a method described in Sec. 3.1 we obtained:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Latitude</th>
<th>Longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>144</td>
<td>379</td>
<td>39.028544</td>
<td>-76.965623</td>
</tr>
<tr>
<td>392</td>
<td>316</td>
<td>39.028713</td>
<td>-76.965697</td>
</tr>
<tr>
<td>303</td>
<td>154</td>
<td>39.028849</td>
<td>-76.965554</td>
</tr>
<tr>
<td>110</td>
<td>113</td>
<td>39.288847</td>
<td>-76.965460</td>
</tr>
</tbody>
</table>

Using a method described in Sec. 3.1 we obtained:

\[
\begin{align*}
    h_{11} &= 0.000358993270888 \\
    h_{12} &= 0.003247534155592 \\
    h_{13} &= 0.891819312632688 \\
    h_{21} &= 0.000182044706525 \\
    h_{22} &= 0.001646834456496 \\
    h_{23} &= 0.452228512089250 \\
    h_{31} &= 0.000004664208515 \\
    h_{32} &= 0.000042194812095 \\
    h_{33} &= 0.011587239036270
\end{align*}
\]

These parameters are used to calculate distance between object position in frames 3344 and 3483. Frames 3344 and 3483 are illustrated in Fig. 3.12 and Fig. 3.13. Object position is:
Calculating the distance is done using Algorithm 1.

Eqs. (3.6) and (3.7) calculate latitudes and longitudes of object position:

\[ X_1 = \frac{h_{11} \cdot 350 + h_{12} \cdot 265 + h_{13}}{h_{31} \cdot 350 + h_{32} \cdot 265 + h_{33}} = -76.965630 = \text{long}_1 \]
\[ Y_1 = \frac{h_{21} \cdot 350 + h_{22} \cdot 265 + h_{23}}{h_{31} \cdot 350 + h_{32} \cdot 265 + h_{33}} = 39.028769 = \text{lat}_1 \]
\[ X_2 = \frac{h_{11} \cdot 210 + h_{12} \cdot 364 + h_{13}}{h_{31} \cdot 210 + h_{32} \cdot 364 + h_{33}} = -76.965645 = \text{long}_2 \]
\[ Y_2 = \frac{h_{21} \cdot 210 + h_{22} \cdot 364 + h_{23}}{h_{31} \cdot 210 + h_{32} \cdot 364 + h_{33}} = 39.028589 = \text{lat}_2 \]

Distance is calculated using longitude and latitude values: \text{long}_1, \text{lat}_1, \text{long}_2, \text{lat}_2 in Eqs. (3.8), (3.9) and (3.10)

\[ a = \sin^2\left(\frac{\text{lat}_2 - \text{lat}_1}{2}\right) + \cos(\text{lat}_1) \cdot \cos(\text{lat}_2) \cdot \sin^2\left(\frac{\text{long}_2 - \text{long}_1}{2}\right) = 2.487358 \cdot 10^{-12} \]
\[ c = 2 \cdot \arctan(\sqrt{a}, \sqrt{1 - a}) = 3.154272 \cdot 10^{-6} \]
\[ \text{dist} = R \cdot c = 20.09m \]

Fig. 3.14 is used to demonstrate height calculation. Height of the object inside the blue bounding box is known. It is used as the reference height. In order to apply Algorithm 2 and calculate height of the object inside the red bounding box it is necessary to know the following parameters:

- Equation of the vanishing line in the image: \[ l_1x + l_2y + l_3 = \frac{84}{3114} \cdot x + 1 \cdot y + \frac{139045}{173} \]
Figure 3.14: Frame 3483 with objects shown in a red and blue bounding boxes. Height of the object inside the blue bounding box is known. Height of the object inside the red bounding box is unknown.

- Position of the vertical vanishing point in the image: $\vec{v}=(v_x, v_y, 1)=(320, 50000, 1)$

- Bottom location of the reference height: $\vec{b}_r=(b_{rx}, b_{ry}, 1)=(264, 100, 1)$

- Top location of the reference height: $\vec{t}_r=(t_{rx}, t_{ry}, 1)=(264, 153, 1)$

- Reference height: $Z_r=180$ cm

- Bottom location of the unknown height: $\vec{b}_x=(b_{xx}, b_{xy}, 1)=(210, 360, 1)$

- Top location of the unknown height: $\vec{t}_x=(t_{xx}, t_{xy}, 1)=(210, 394, 1)$

Heights are calculated using Algorithm 2. The first step is to compute the matric factor $\alpha$ from Eq. (3.12), using location and value of the reference height. After the metric factor $\alpha$ is calculated, Eq. (3.13) is used to calculate unknown height.

- Matric factor $\alpha = 0.183209$

- Value of the unknown height: $Z_x = 182$ cm
Chapter 4: Experiments and Discussion

This chapter shows results of applying the summarization method on surveillance video. Resulting summary can be a single image or several images. If the video sequence is shorter or equal to 100 seconds, summary is represented by a single image. If the sequence is longer than 100 seconds, resulting summary is represented by several images. Images are presented as a video clip that runs at one frame per second. Value of 100 seconds is experimentally chosen and depends on specific video. It should be short enough to avoid huge overlap between different objects. On the other hand, it should be long enough so that every image contains enough objects to represent a summary of activity.

The method is used for creating video synopsis (Figs. 4.2, 4.3 and 4.4) as well as for showing summaries of particular types of objects, or objects with particular behavior (Figs. 4.5 and 4.6). The first example (Figs. 4.1 and 4.2) is used to illustrate summary of 100-second video sequence. The summary is represented in a single image. The second example (Figs. 4.3 and 4.4) illustrates summary of 35 minutes of video. Since the video sequence is long and number of active object in the sequence is large, result of the summary is sequence of 21 images. The third example (Fig. 4.5) shows results of using objects speed to classify them in two groups: people and vehicles. The last example (Fig. 4.6) shows results of using both objects speed and height in order to distinguish between objects that represent single person and objects that represent group of people.

Fig. 4.1 shows some of the frames from sequence 4000-5000. These frames are chosen to show all foreground objects from the sequence. The sequence corresponds to 100 seconds of video. Summary is shown in Fig. 4.2.
Figure 4.1: Selected frames from sequence 4000-5000

Figure 4.2: Video synopsis of sequence 4000-5000
Figs. 4.3 and 4.4 illustrate summary produced for 35 minutes long video sequence. Since each frame of the summary corresponds to period of up to 100 seconds, the summary is represented by 21 frames. These frames contain complete activity from the sequence.

Figure 4.3: Video Synopsis for frames 1-12000. Each image represents a 1000 frames
Fig. 4.5 shows results of using objects speed to classify them in two groups: people and vehicles. The first row of the figure represents complete summary for first 6000 frames (10 minutes) of the video. Each image is synopsis of 2000 frames. The second row shows the summary of all objects that have average speed higher than $15 km/h$. It is rare for people to move faster than $15 km/h$ and almost all the vehicles move faster than $15 km/h$. Therefore this row shows all the activity from the sequence made by vehicles. The third row represents the summary of all objects that have average speed lower than $15 km/h$. It is a summary of all the activity made by people.
Figure 4.5: Object classification based on speed: Top row shows synopsis of the sequence 1-6000. Middle row shows summary of objects faster than $15\text{km/h}$, while bottom row shows summary of objects slower than $15\text{km/h}$ from the sequence 1-6000.

Fig. 4.6 shows results of using both objects speed and height in order to distinguish between objects that represent single person and objects that represent group of people. The first image in the figure represents complete summary for 2000 frames (200 seconds) of the video. The second image shows the summary of all the objects that have average speed lower than $15\text{km/h}$ and that are higher than $210\text{cm}$. Using the speed restriction helps to distinguish between people and vehicles. Using the height restriction helps to distinguish between objects that represent single person and objects that represent group of people.
Any object that represents a group of people has to have an average height higher than 210 cm since people in a group do not walk as one person and resulting object is almost always higher than any person from a group. The third image shows the summary of all the objects that have an average speed lower than 15 km/h and that are lower than 210 cm. Therefore, it shows the summary of objects that represent a single person.

Figure 4.6: Object classification based on speed and height: Image on the left shows synopsis of the sequence 5000-7000. Image in the middle shows summary of objects that represent a group of people, while image on the right shows summary of objects that represent a single person.
Chapter 5: Conclusion and Future Work

This thesis describes a method for efficient summarization of long surveillance videos. The method consists of four phases: ground plane calibration, detection and tracking of scene objects, extracting information about objects in the scene, generating and visualizing the summarizations. The method assumes a static camera. Both extrinsic parameters—3D position and orientation, and intrinsic parameters—focal length, principal point, lens distortion of the camera are unknown. Ground plane calibration is achieved by computing a homography between the scene and corresponding location in Google Earth. Detection and tracking are based on techniques described in [2, 3]. Planar homography and single view metrology are used to calculate widths, heights, position and speed of objects in the scene. The method generates video summarization for video sequence by choosing a single image of each tracked object and overlaying it on the background image. The method chooses images of tracked objects in a way to minimize the overlap between them. For each tracked object its trajectory is shown as a sequence of vectors corresponding to object motion between successive frames. The method generates video synopsis—a summary of all activity for specified period of time. Since speeds and sizes of objects are calculated, the method also generates sequences using various combinations of object properties. To determine the effectiveness of the method, four experiments on a 23000 frame sequence were done. The first experiment creates video synopsis of 1000 frames of the video sequence. The summary is represented in a single image. The second experiment creates video synopsis of 21000 frames of the sequence and it is represented using 21 images. The third experiment creates separate video summaries for people and for vehicles from the sequence. Finally, the last experiment creates video summaries for objects that represent single person and objects that represent group of people.

Incorporating better or additional information from tracker could be the subject of
future work. Current information obtained by tracker include object’s location, bounding box dimensions and foreground/background ratio inside the bounding box. With additional information from improved tracker it would be possible to come out with different types of summary. By saving object’s color histogram it would be possible to classify objects based on color. This could be useful in case of vehicles. Summaries could also be based on ROI—region of interest or on more specific types of behavior. Current bounding box has rectangular shape and ratio between foreground and background inside the box depends on the orientation of the object. In order to minimize background area inside the box, it is possible to use Rotating Calipers [56] and create much tighter bounding boxes. This would improve classification based on object area. Knowing the shape of the bounding box could also help in distinguishing between object and its shadow. Having an option to remove shadow would dramatically improve accuracy of the method since the shadow is the main reason for imprecise measurements of height and width of the object.

Algorithm from the method that creates summarization images uses greedy approach for placing foreground objects on the background image. Introducing the backtracking step into the algorithm would lower the overlap between foreground objects and improve resulting images. Also, the method could be implemented using interactive interface. Using Google Earth API or Google Maps API it would be possible to allow user to interactively change results of the method: to remove or add displayed objects and trajectories; to get additional information about objects by clicking on trajectories; to get a video of object by clicking on trajectories.
Bibliography


Curriculum Vitae

Dusan I. Puletic was born on December 1, 1981, in Nis, Serbia. He received his Engineering Diploma, major in Computer Science, from University of Nis, Serbia in 2009. While progressing through the Master of Science in Computer Science program at George Mason University he worked as an intern for videoNEXT Network Solutions, Inc. in Chantilly, VA.