MULTI-CAMERA MOTION DETECTION

Lung-Chang Hsieh

May 8, 2007

Boston University

Department of Electrical and Computer Engineering

Technical report No. ECE-2007-3

BOSTON
UNIVERSITY

MULTI-CAMERA MOTION DETECTION

Lung-Chang Hsieh

MS PROJECT REPORT

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
www.bu.edu/ece

May 8, 2007

Technical report No. ECE-2007-3

Table of Contents

1. Abstract

The objective of this MS project was to investigate change detection in a scene using two synchronous cameras with overlapping fields of view. Specifically, of interest was monitoring of vehicle traffic on I-90 interstate highway.

This report summarizes work completed within the MS project. First, a state-of-the-art change detection algorithm using single camera is described. The algorithm is then extended to two cameras by applying affine mapping to align video sequences from two synchronous cameras. Since affine mapping is accurate only within planar surfaces, two cases were considered: affine transformation applied to highway surface, and affine transformation applied to side of a truck/bus/car.

The experimental results illustrate that affine mapping using feature points selected from the background (highway surface) tend to match target objects more accurately than affine mapping using feature points selected from foreground objects (large trucks). It was also observed that optical distortions caused by wide-angle camera lens contribute to inaccuracies in the affine transformation between cameras.

It is concluded that the bi-camera change detection as implemented in this project is inferior to single camera detection due to lens distortions and simple mapping model.

2. Introduction

Change detection in a scene using multiple cameras can potentially improve detection accuracy over a single-camera detection since more information about the scene is available. Expected benefits are: increased detection accuracy and sensitivity. The objective of this project is to investigate such a multi-camera change detection in the particular context of highway traffic monitoring from cameras positioned above the I-90 interstate highway.

Two assumptions were made: 1) all cameras are stationary, 2) background is planar. Based on these assumptions, there are three major challenges. The first challenge is to construct a network of video cameras and to develop suitable synchronous multi-camera video acquisition software. The second challenge is to analyze and implement a state-of-the-art single-camera change detection algorithm, such as one proposed by Elgammal al et [1]. The third challenge is to extend this single-camera algorithm to two cameras assuming planarity of the background.

This report first introduces the network camera setup and the video acquisition software. The state-of-the-art change detection algorithm using single camera is then reviewed, followed by implementation of the change detection algorithm using two cameras. Experimental results, conclusions, and suggestions for future work follow at the end of the report.

3. Network Camera Setup and Video Acquisition Software

3.1 Network Camera Setup

The hardware setup of this project involves Axis 207W and Axis 213 PTZ network cameras along with Linksys WRT54G routers. Both camera models are capable of providing live video streams in M-JPEG and MPEG-4 formats. Each camera contains a built-in Web server, which allows viewers to see live video via a standard Web browser, while permitting selected users to control some camera functions (such as pan-tilt-zoom). Currently, four Axis 207W and two Axis PTZ network cameras are set up for users to access.

All cameras are connected to one of the Boston University NTP time servers for time synchronization. The hourly time adjustment in each camera varies based on the camera server reports. However, the adjustments for some cameras can be as large as 0.03 seconds. This implies that there might be a one-frame offset while streaming at a rate of 30 frames per second. In addition, heavy network traffic causes frequent frame drop outs, despite the fact that some cameras are connected through wired Ethernet..

The Axis 207W's are located on the fourth floor of the Photonics Building. Two of them wirelessly communicate to a router that is assigned a static IP address. The router is then connected to a local area network. The rest are connected directly to the same LAN. All four 207W's are looking down at the I-90 interstate highway and have partially overlapping views.

The list of available Axis 207W network cameras:

- Router 1 (PHO 439)
  - PHO 438: http://vsn1-iss.bu.edu:10438
  - PHO 435: http://vsn1-iss.bu.edu:10435
- PHO 443: http://vsn3-iss.bu.edu
- PHO 440: http://vsn2-iss.bu.edu

The Axis 213 PTZ cameras are installed on the roof of the Photonic Building. Each camera is assigned a static IP address and is wire-connected to a subnet on the 9-th floor. Both cameras are set to face the City of Cambridge.

The list of active Axis PTZ network cameras:

- PHO West: http://ptz1-iss.bu.edu
- PHO East: http://ptz2-iss.bu.edu

3.2 Video Acquisition Software

The video acquisition software, Photonics Camera Network (PCN), was developed in C-sharp along with the Axis Media Control SDK, which contains graphic decoder and various media control functions for Axis products, under the Microsoft Visual Studio 2005 environment. The PCN allows users to simultaneously view up to four video streams and save them, at the rate of 10 frames/sec, to local disk. The development of PCN intends to provide for future image and video processing

applications in the department. The PCN software is likely to prove valuable in courses such as SC463/464, SC520, SC717, and SC720. Fig. 1 below show a snapshot of PCN window.

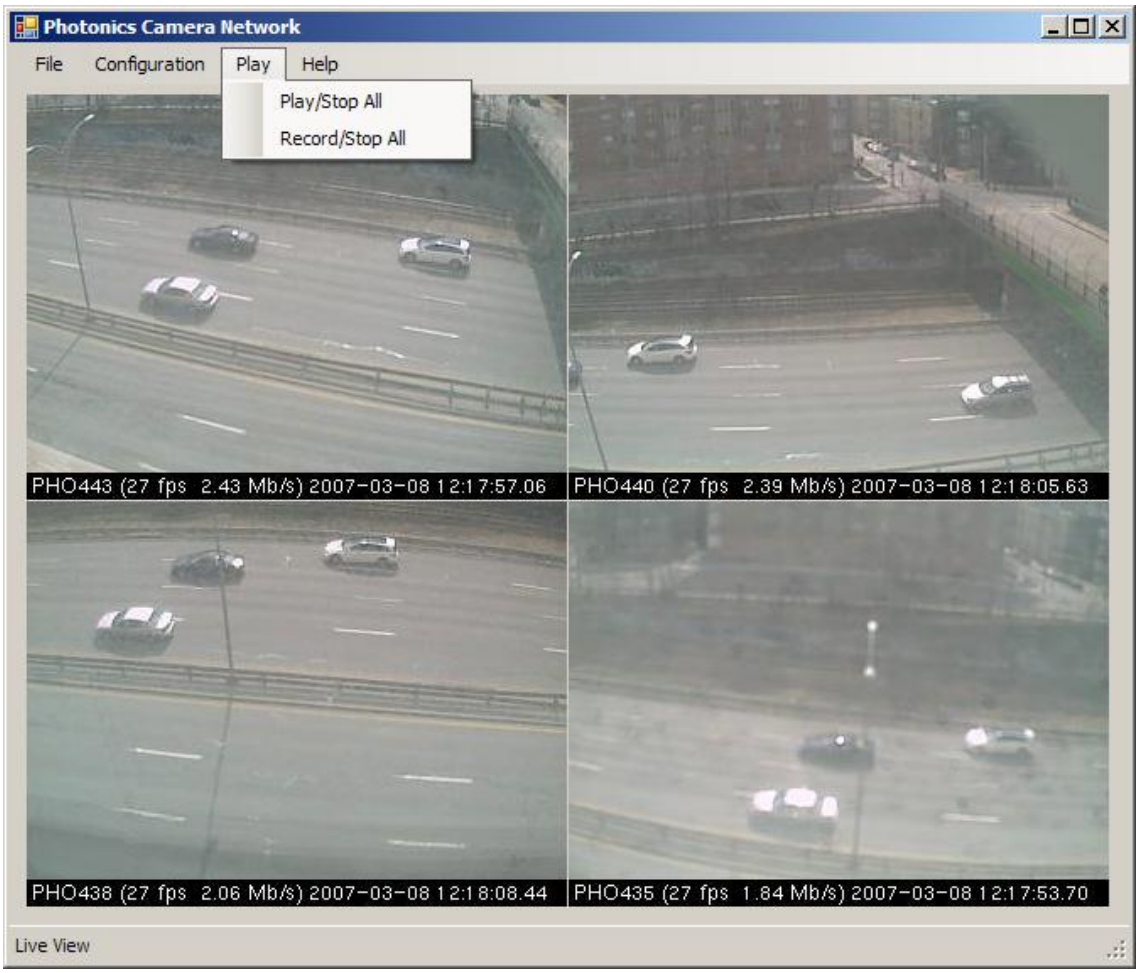


Figure 1. A snapshot of PCN window

4. Single-Camera Change Detection Algorithm

Background subtraction techniques are often used in change detection and motion detection. The most fundamental method is to average a number of consecutive frames and threshold the difference between a new frame and this average using a global threshold. This method is simple to implement, but it results in low detection accuracy. Elgammal al et [1] proposed a more sophisticated method. The main idea is to construct a non-parametric background model for each pixel in form of a probability distribution of

intensity values. This background model can handle situations where the background of the scene contains small motion or variations in luminance. The model is sensitive to motion however, and adapts quickly to changes in the background.

4.1. Basic Background Model

4.1.1 Probability Density Estimation

Let $x_1, x_2, ..., x_N$ be a recent sample of intensity values for a single pixel. Using these samples, the probability that this pixel will have intensity value x_t at time t can be non-parametrically estimated [1] using the kernel estimator K as follows

$$Pr(x_t) = \frac{1}{n}\sum_{i=1}^{N} K(x_t - x_i) \tag{1}$$

The kernel estimator function, K, is chosen to be a Normal function N(0, σ^2), where σ^2 represents the kernel function width. Then, the density for grayscale images can be estimated as

$$Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \quad e^{-\frac{1}{2}\frac{(x_t - x_i)^2}{\sigma^2}} \tag{2}$$

For color images, K becomes $N(0, \Sigma)$, where Σ is a covariance matrix that represents the kernel function width. If different color channels are treated independently with different kernel widths σ_j^2, then the matrix Σ is diagonal

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{bmatrix}$$

and the probability function can be written as

$$Pr(x_t) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma_j^2}} \quad e^{-\frac{1}{2}\frac{(x_{t_j} - x_{i_j})^2}{\sigma_j^2}} \tag{3}$$

Using the estimated probability, a pixel is classified as a foreground pixel, i.e., changed (moving) pixel, if $Pr(x_t) < th_1$ where the threshold th_1 is a global threshold over the entire image. This threshold can be adjusted to achieve a desired percentage of false positives. Clearly, if probability of x_t is low, i.e., it is unlikely to come from the same probability distribution as the background pixels, x_t is likely to belong to the foreground. For fast implementation, pre-calculated lookup tables for the kernel function values given the intensity difference $(x_t - x_i)$ and the kernel function width are used.

(a) (b)

Figure 2. (a) Original image. (b) Result of thresholding $Pr(x_t)$

4.1.2 Kernel Width Estimation

To estimate the kernel width σ_j^2 for the jth color channel for a given pixel, the median absolute deviation is computed over a sample of consecutive intensity values of the pixel. The median, denoted m, of consecutive differences $|x_i - x_{i+1}|$ is calculated independently for each color channel. Since the deviations between two consecutive intensity values usually come from the same local-in-time distribution, only few pairs are likely to come from cross distributions (i.e., those of the background and foreground). If this local-in-time distribution is assumed to be Normal $N(\mu, \sigma^2)$, then the difference $(x_i - x_{i+1})$ is also Normal but with double variance, $N(0, 2\sigma^2)$. The standard deviation of the first distribution can then be obtained from the following relationship:

$$\sigma = \frac{m}{0.68\sqrt{2}}$$

4.2 Suppression of False Detections

False detections that result from random noise, which should be homogeneous over the entire image, and from small movements, due to camera or background object displacements, often appear after thresholding $Pr(x_t)$. Clearly, a suppression of false detections is needed as a post-processing step. The goal is to alleviate false detections due to small movements in the background, such as leaves fluttering in the wind, waves on water surface, etc.

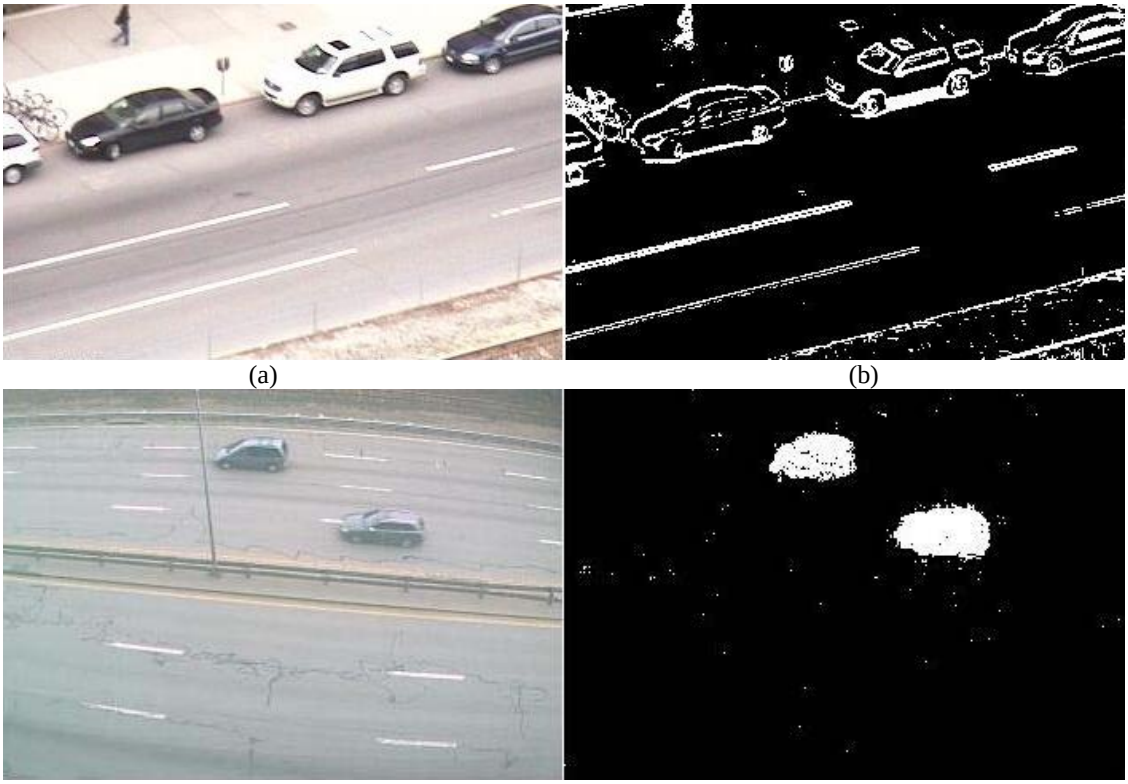


Figure 3. (a) Original image from 213PTZ camera, and (b) false detections due to vibration of this camera. (c) Original image from 207W camera, and (d) false detections due to random noise.

Let x_t be the observed value of a pixel, *x*, detected as a foreground pixel from the probability estimate (first stage) at time *t*. Define the pixel displacement probability, $P_{\mathcal{N}}(x_t)$, to be the maximum probability that the observed value, x_t, belongs to the background distribution of some point in the neighborhood $\mathcal{N}(x)$ of x

$$P_{\mathcal{N}}(x_t) = \max_{y \in \mathcal{N}(x)} Pr(x_t \mid B_y)$$

where B_y means that the pixel y, which is within the neighborhood of x, belongs to the background model. The probability estimate $P_r(x_t \mid B_y)$ is calculated using the kernel function estimation as in (2) for grayscale images and in (3) for color images.

By thresholding $P_{\mathcal{N}}$ many false detections due to random noise and due to small motions in the background can be eliminated. However, it is also possible to lose true detections during this process because some true detected pixels might accidentally fit the background distribution of neighbor pixels. This happens more often on grayscale images.

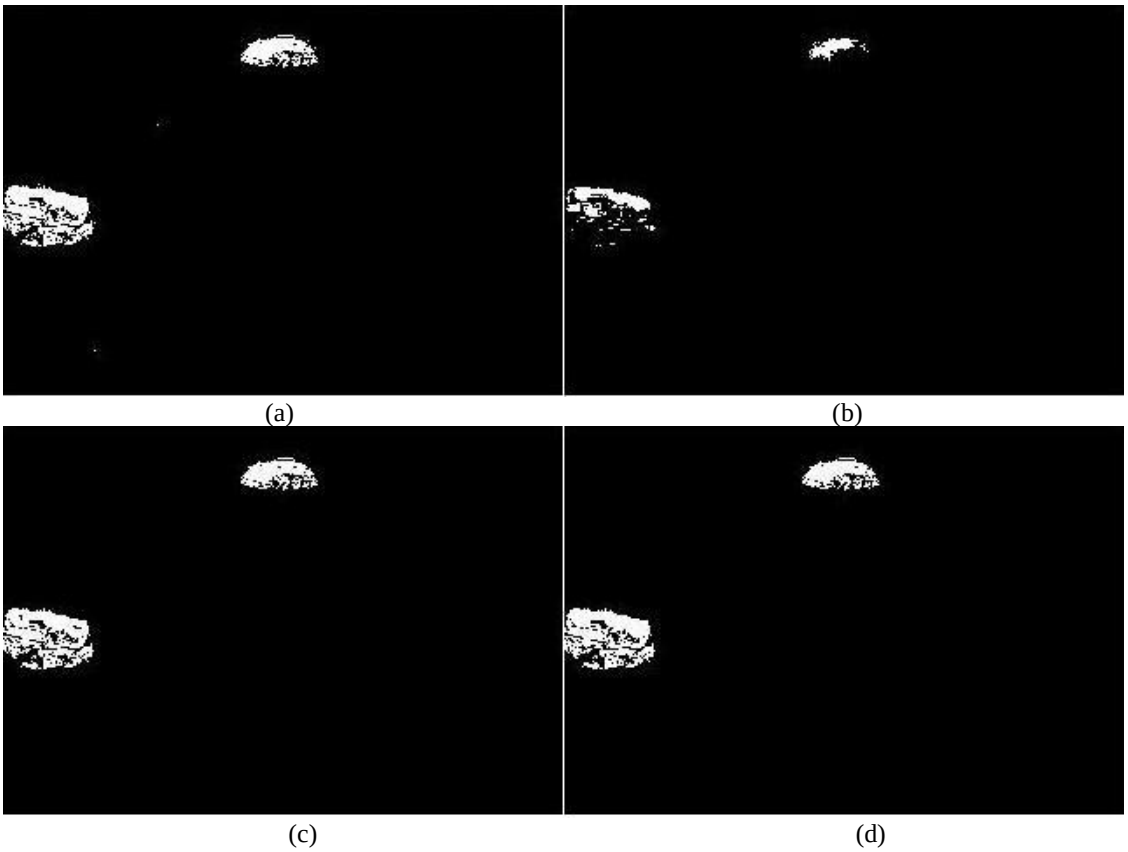
(a) (b)

Figure 4. (a) Original image (b) Example of over suppression when thresholding $P_{\mathcal{N}}$

In order to avoid losing true detections, an additional constraint can be added, such as that the whole detected foreground object, not only some of its pixels, must have moved from a nearby location. The component displacement probability $P_{\mathcal{C}}$ is the probability that a detected connected component *C* has been displaced from a nearby location. Although the meaning of "connected component" is not detailed in the original paper [1], we understand this to mean a group of detected (changed) pixels that are mutual neighbors, e.g., in the sense of first- or second-order neighborhood structure (MRF). This probability is estimated by

$$P_{\mathcal{C}} = \prod_{x \in \mathcal{C}} P_{\mathcal{N}}(x)$$

For a connected component corresponding to a real target, the probability that this object has displaced from the background will be very small. Thus, a detected pixel x will be considered to be a part of the background only if$(P_{\mathcal{N}}(x) > th_1) \bigcap (P_{\mathcal{C}}(x) > th_2)$.



(a) (b)

(c) (d)

Figure 5. (a) Result of raw detection. (b) Result of pixel displacement suppression. (c) Result of component displacement suppression. (d) Result of $(P_{\mathcal{N}}(x) > th_1) \bigcap (P_{\mathcal{C}}(x) > th_2)$

In my implementation, a square neighborhood of five is used to determine pixel displacement probabilities. The first threshold th_1 is identical to the threshold used in the first stage. The second threshold th_2 can effectively distinguish between real moving objects and displaced ones because a real moving object has a much lower component displacement probability. This threshold is selected independent of th_1 and has a much smaller value.

4.3 Updating the Background Model

At this point, a background model has been constructed and needs to be updated continuously in a first-in-first-out manner. There are two alternative mechanisms to update the background. One is the selective update method, which adds the new sample to the model only if it is classified as a background sample. Although selective update adapts to changes quickly, it potentially results in error propagations. The other is the blind update method, which adds the new sample to the model regardless of the classification of the pixel. However, the blind update adapts slowly to changes in the background.

The selective update method is used as a short-term model, which takes M recent sample pixels from previous frames. The blind update method, on the other hand, is used as a long-term model, and contains N ($N > M$) sample pixels from previous frames. A combination of both methods results in a background model that adapts quickly to changes in the scene to support sensitive detection and low false detection rate. The final detection results are produced by intersecting the results from both updating models.

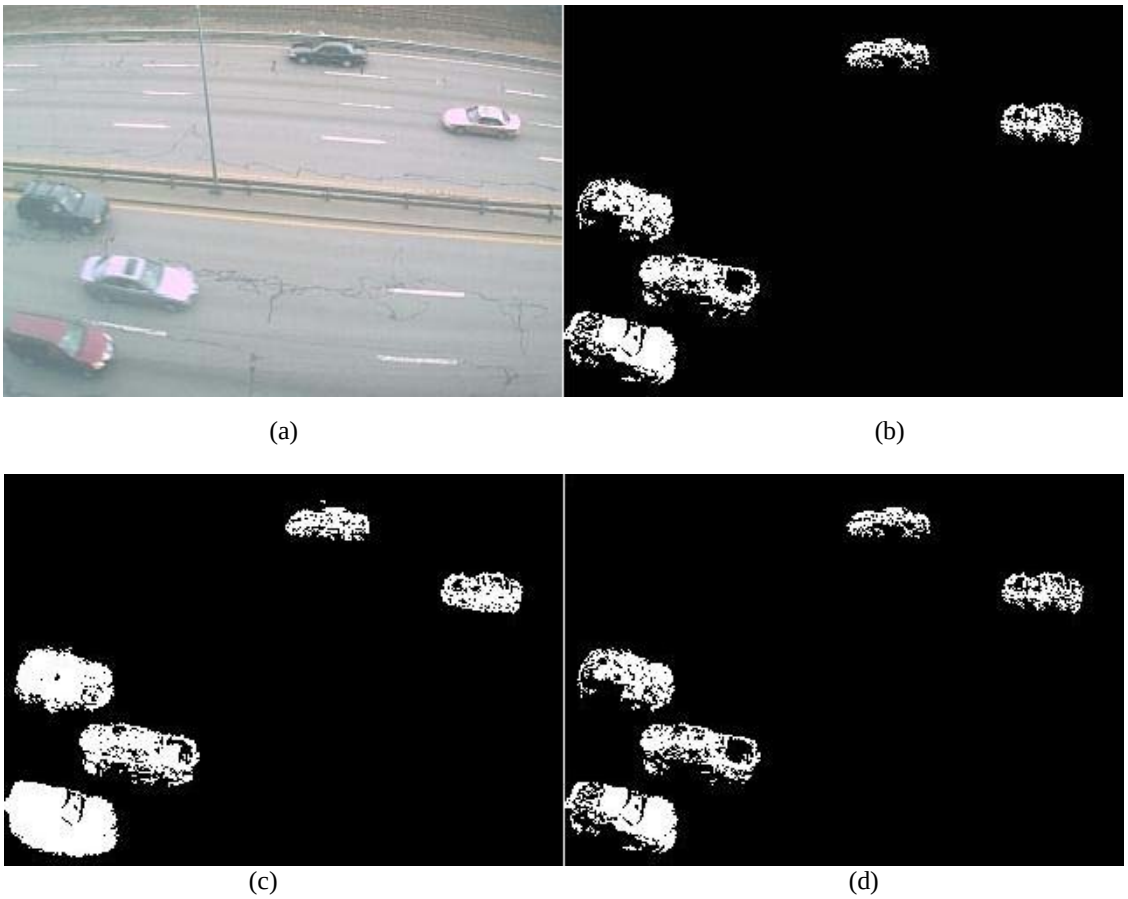


Figure 6. (a) Original image. (b) Result using blind update. (c) Result using selective update. (d) Result of intersection of (b) and (c).

5. Change Detection from Two Cameras

In order to be able to use intensities from two cameras with overlapping fields of view, we need to establish correspondence between features in images captured at the same time by the cameras. In general, such a correspondence can be established be a dense vector field that to each location in one image assigns a location in the other image. However, establishing such a dense correspondence is computationally expensive. Moreover, since we assumed that the background is planar (highway surface), a simpler parametric correspondence can be established by means of affine transformation which accounts for rotation, scaling, and displacement, and requires to compute only 6 parameters. Efficient techniques exist to compute these 6 parameters of affine transformation.

5.1 Affine Transformation

The affine transformation can be represented as follows:

$$\vec{w} = A\,\vec{y} + \vec{d} \qquad \{\vec{y} \in image\ B \mid \vec{w} \in warped\ image\ B\} \tag{4}$$

where A is a 2 × 2 transformation matrix and $\vec{d}$ is the displacement vector.

The above transformation is accurate for image points *y* that are projections of 3D points on a plane in the original scene, such as the highway surface. We calculated transformation matrix A_{BG} and displacement vector $\vec{d}_{BG}$ in this case from manually selected feature points on highway surface (lane markings) as shown in Fig. 7(a).

Since the cars/buses/trucks on the highway are 3D, non-planar structures traveling on a planar surface, they do not obey the affine constraint. Therefore, a point on a car in one image affine-mapped to the other image is likely not to correspond to the same feature. Moreover, any image point resulting from a projection of any feature not on highway surface will be likely detected as a changed (moving) point. To alleviate this we also considered estimation of affine parameters A_{FG} and $\vec{d}_{FG}$ from feature points on one side of a large truck (foreground mapping) that can be considered planar. Feature points are marked manually and affine parameters are computed using *Matlab* functions available in VIP Laboratory.

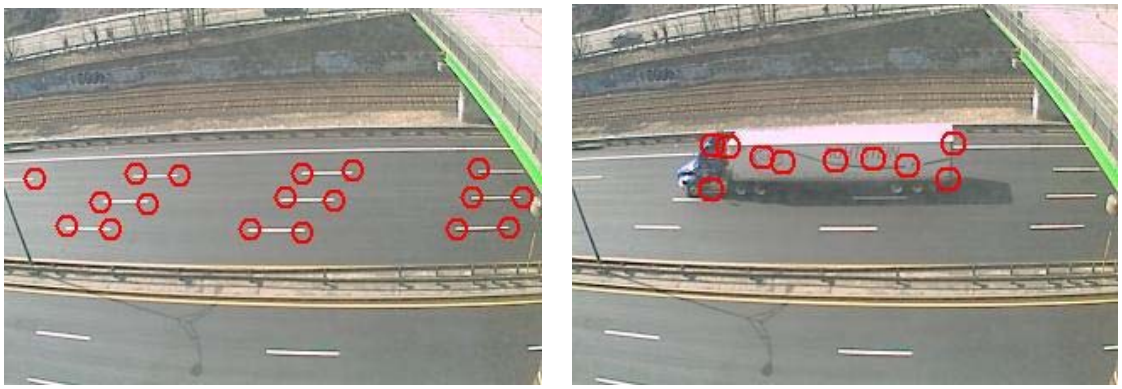


Figure 7. (a) Feature points on highway surface. (b) Feature points on a large truck.

5.2 Change Detection from Two Cameras

With the two images aligned, at each time instant there are two intensities available for each pixel in one image that has a correspondence in the other image. Note that some pixels may have no correspondence due to occlusions and also due to slightly different fields of view. We consider only pixels that are present in both fields of view and we ignore occlusion effects.

Thus, by projecting one of the images (image *B*) using the affine mapping onto the coordinate system of the other image (image *A*), and placing this warped image immediately after image *A*, we generate a new video sequence of double length. We apply the algorithm of Section 4 to this new sequence. We expect that, due to affine mapping inaccuracies, intensity variability for a given pixel will be larger than that of the original sequence *A*, and thus we estimate a new kernel width parameter.

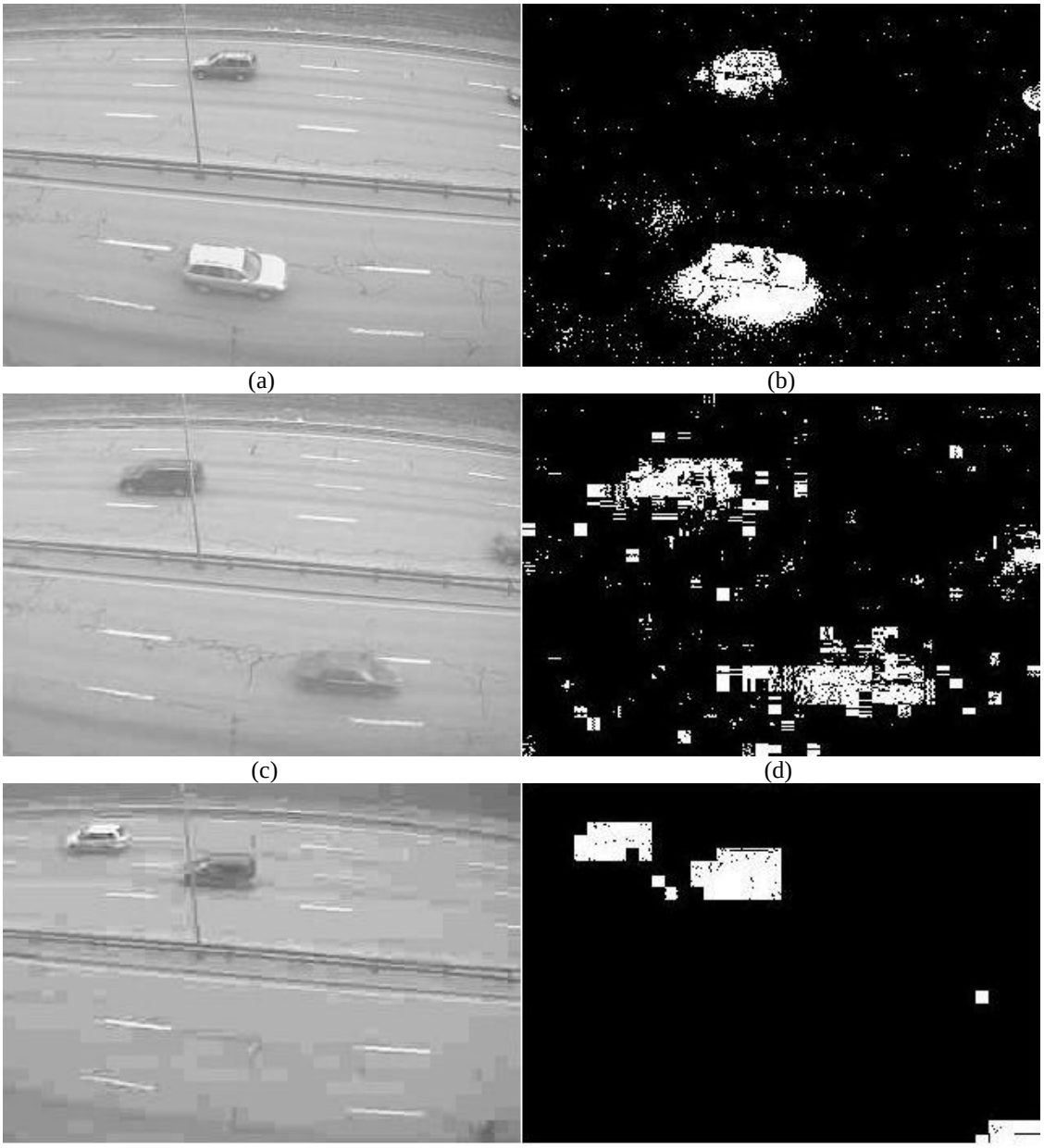
6. Experimental Results

All image sequences were captured from 207W's at a rate of 30 fps with the resolution of 320×240 and were processed offline. Morphological closing was used to clean up the final detection results. The sample size N varies in different experiments. Most of sequences used were captured in good weather conditions.

6.1. Impact of JPEG Compression on Change Detection

During early experiments, we observed block-structured clusters in the detection results. We suspected this to be the result of JPEG compression used to stream the video. In order to confirm our hypothesis, we varied the compression ratio and performed change detection using the same parameters in each case.

Let C be the rate of compression ranging from 0 to 100; 0 and 100 being the minimal and maximal compression, respectively. Several image sequences were captured at various rates of compression to demonstrate how JPEG compression affects the detection results.



(a) (b)

(c) (d)

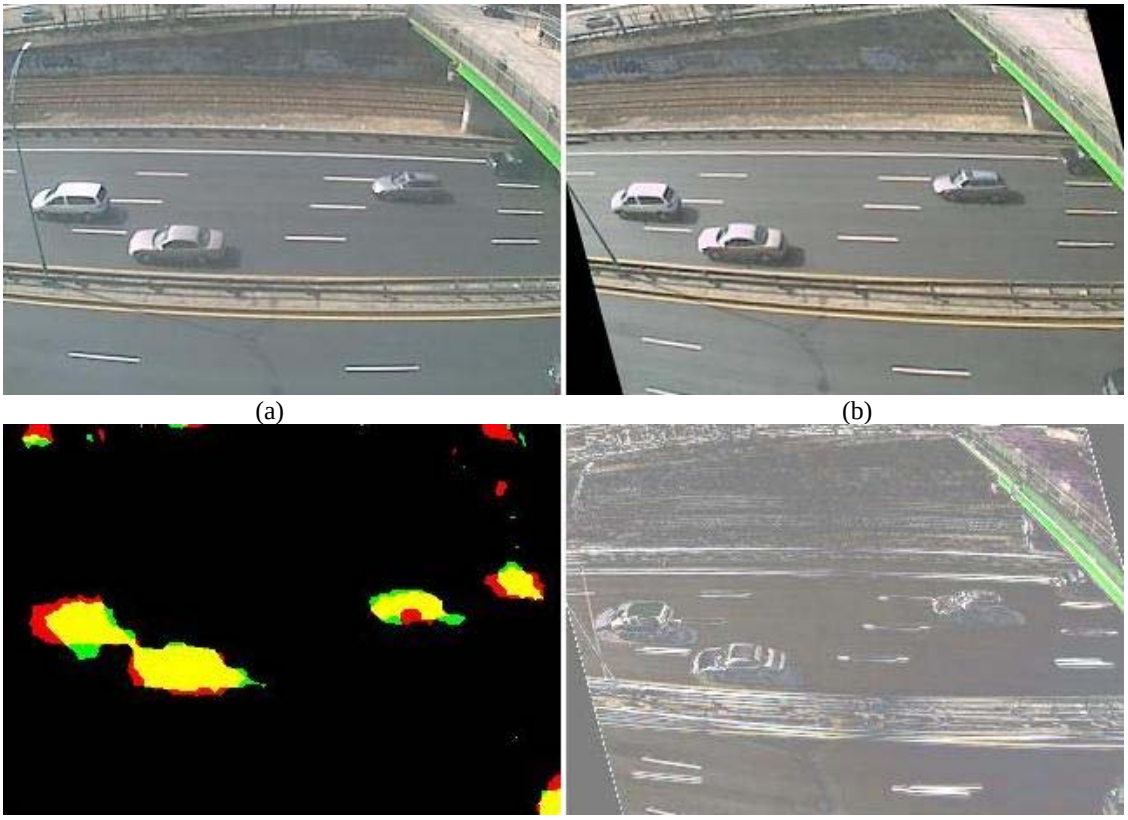
(e) (f)

Figure 8. (a) Original image at C = 0. (b) Detection result of (a). (c) Original image at C = 50. (d) Detection result of (c). (e) Original image at C = 90. (f) Detection result of (e).

As can be seen in Fig. 8, the impact on the detection result is minimal at C = 0. Most details in the target objects are still distinguishable. The background model is also sensitive to the shadow of the target objects although this effect is barely visible. At C = 50, the detection result contains less details of the target objects. At C = 90, it is difficult to identify what the target objects really are because of the blocking artifacts.

The detection results from differently-compressed sequences may prove useful, even at low rates, depending on the type of application. If the detail of target objects is not a priority, the detection results from lower rates of compression may be sufficient. Furthermore, low rates of compression allows fewer frame drops if the cameras operate in wireless mode.

6.2. Results of Single-Camera Change Detection



(a) (b)

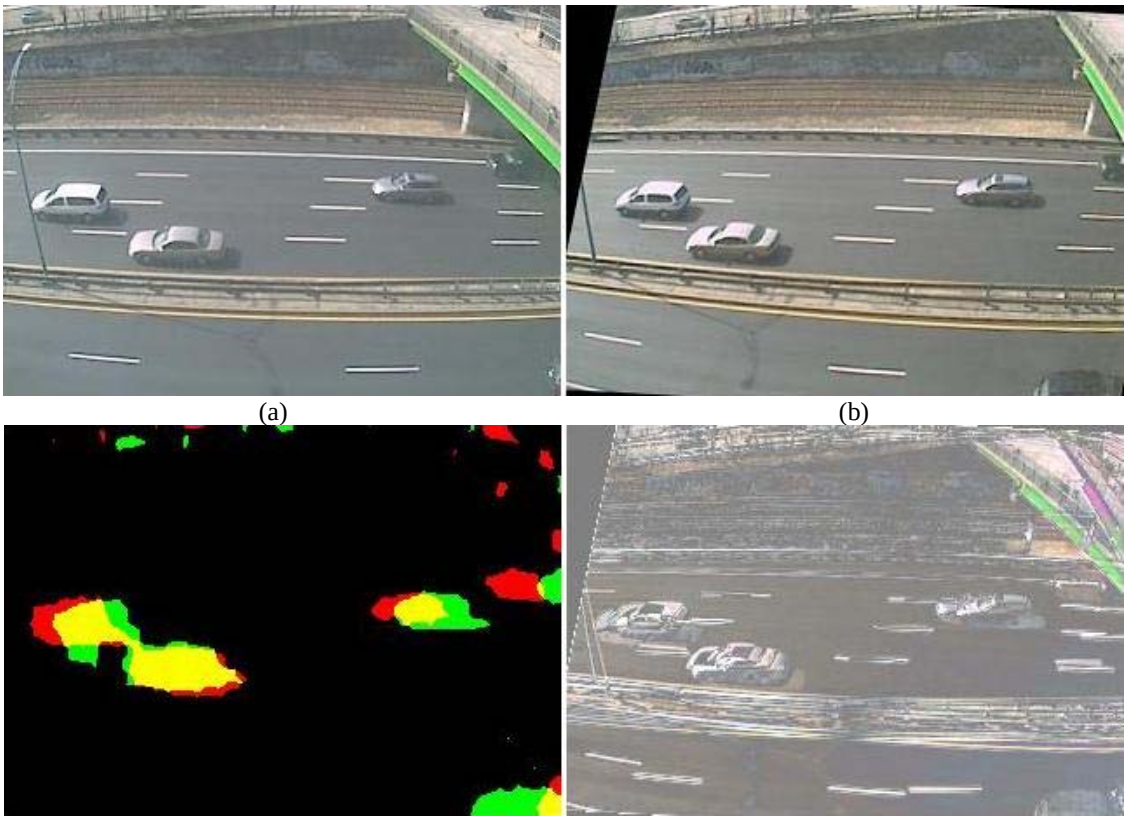
(c) (d)

Figure 9. Change detection using the background affine mapping: (a) image A; (b) image B warped to coordinate system of image A; (c) intersection (yellow) of detection results from image A (red) and from warped image B (green); (d) difference of intensities between cropped image A and warped image B.

Fig. 9 shows change detection results for single-camera algorithm described in Section 4. First, affine mapping is used to back project image B to the coordinate system of image A (Fig. 9(b)) using feature points on the pavement (background affine mapping). Then, the change detection algorithm is applied to image A and warped image B. Fig. 9(c) shows both detection results overlaid on top of each other. The intersection of detected pixels is shown in yellow. Since red (image A) and green (image B) detected

pixels are rather few, the detection results from two cameras are in a reasonably good alignment.

The results of detection using foreground affine parameters (computed from feature points on a large truck – Fig. 7(b)) are shown in Fig. 10.



(a) (b)

(c) (d)

Figure 10. Change detection using the foreground affine mapping: (a) image A; (b) image B warped to image A; (c) intersection (yellow) of detection results from image A (red) and from warped image B (green); and (d) difference of intensities between cropped image A and warped image B.

By looking at the overlays in both cases, it seems that the mapping using background affine parameters outperforms mapping using foreground affine parameters. All moving objects (vehicles on the highway) in the scene are well-aligned using the background mapping. Misalignments in the case of foreground mapping are due to the inability to capture vehicle shape at different depths (lanes) using a single plane orthogonal to highway surface, and also due to potential inaccuracies in estimating foreground affine parameters (few, closely clustered feature points).
in wireless mode.

6.3. Results of Dual-Camera Change Detection

Instead of processing image sequences from the two cameras independently, a new sequence was composed by interleaving sequence A and warped sequence B. The resulting sequence has twice as many samples as the individual ones. The algorithm was then applied to the new sequence. Interleaving the two sequences increase intensity (color) variability between frame pairs. This is similar to the situation when leaves in the background swing back and forth. However, the algorithm does not perform as well as it is expected.

Fig. 11 shows change detection results based on video sequences from two time-synchronized cameras. An obvious observation is that the bridge and the railing are detected as foreground while some parts of the vehicles in the scene are detected as background. This is because the railing and the bridge are not on the same planar surface as the highway. The simple mapping model used is not inaccurate.

Moreover, we also observed that the 207W camera, having a wide-angle lens, introduces severe geometric distortion (straight lines are curved close to image periphery). This is an additional handicap for the dual-camera detection using an affine mapping; although both cameras may have similar geometric distortions, since their fields of view differ the geometric distortion has, in general, different impact on differently located same features in both images. One needs to compensate for this geometric distortion in order to improve image alignment.

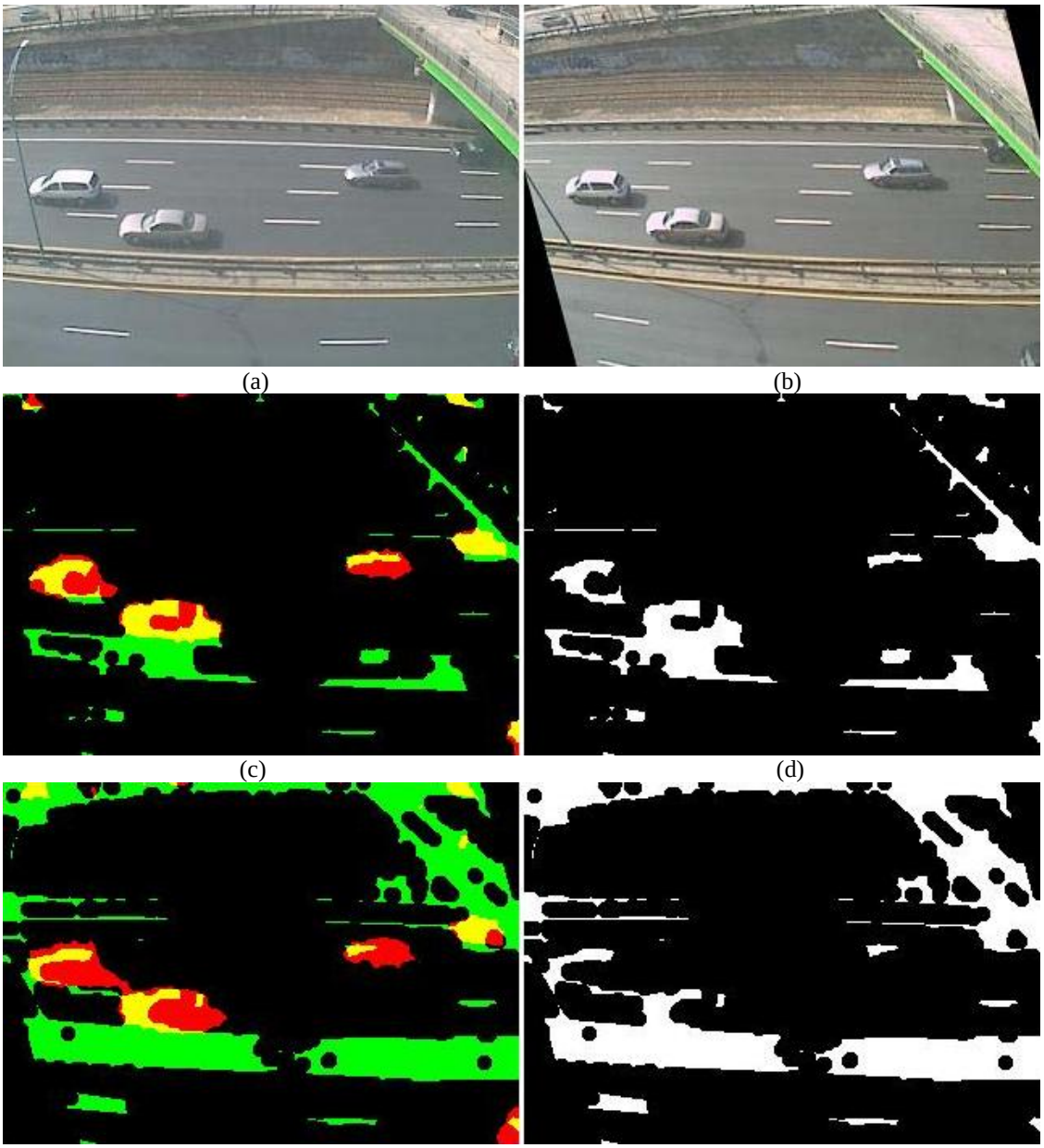


Figure 11. Change detection using two cameras (sequence A interleaved with warped sequence B). (a) image A; (b) warped image B; (c) intersection (yellow) of single-camera detection result from image A and dual-camera detection result (cropped image A interleaved with warped image B) in grayscale; (d) dual-camera detection result (cropped image A interleaved with warped image B) in grayscale; (e-f) similar ual-camera change detection results using color.

7. Conclusions and Future Extensions

One of the obstacles in this project is in-synchronous video acquisition. Frequent 0.03 seconds (1 frame) errors due to NTP synchronization make synchronous video acquisition difficult. Another obstacle is due to network traffic. Although the two cameras used for video acquisition are connected to LAN using wired ehternet, frequent frame drops occur while acquiring video data.

The change detection algorithm by Elgammal at el [1] works well for large sample size, N. As the sample size grows, the background effectively removes undesired false positives with appropriate threshold. False positive suppressio may be omitted to reduce implementation time and computation complexity.

The experimental results basically indicate that background affine parameters tend to match target objects more accurately than foreground affine parameters. Overall, the dual-camera change detection is inferior to single camera detection due to lens distortions and simple mapping model.

This project can be extended in three aspects in the future. First, a model that can account for lens distortion needs to be developed. According to the experimental results, lens distortions can severely jeopardize mapping accuracy. Second, a more sophisticated correspondence model needs to be employed in order to minimize mapping errors. Finally, the number of cameras can be increased to three.

8. References

1. A. Elgammal, D. Harwood, and L. Davis, “Non-parametric Model for Background Subtraction,” 6th European Conference on Computer Vision. Dublin, Ireland, June/July 2000.

2. A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis “Background and Foreground Modeling using Non-parametric Kernel Density Estimation for Visual Surveillance”, Proceedings of the IEEE, July 2002.

3. Linksys WRT54G router user’s manual link: http://www.linksys.com/servlet/Satellite?c=L_Product_C2&childpagename=US%2FLayout&cid=1149562300349&pagename=Linksys%2FCommon%2FVisitorWrapper&lid=0034939789B01

4. Axis 207W network camera user's manual link:
   http://www.axis.com/files/manuals/207W_207MW_UM.pdf

5. Axis 213 PTZ network camera user's manual link:
   http://www.axis.com/files/manuals/213_UM_20070425.pdf

6. Axis Media Control user's manual link:
   http://www.axis.com/files/manuals/24533r3.pdf

9. Appendix