ABSTRACT

Title of dissertation: KEY-FRAME APPEARANCE ANALYSIS FOR VIDEO SURVEILLANCE

Kyongil Yoon, Doctor of Philosophy, 2005

Dissertation directed by: Professor Larry Davis
Department of Computer Science

Tracking moving objects is a commonly used approach for understanding surveillance video. However, by focusing on only a few key-frames, it is possible to effectively perform tasks such as image segmentation, recognition, object detection, and so on. In this dissertation we describe several methods for appearance analysis of key-frames, which includes region-based background subtraction, a new method for recognizing persons based on their overall extrinsic appearance, regardless of their (upright) pose, and appearance-based local change detection.

To encode the spatial information into an appearance model, we introduce a new feature, *path-length*, which is defined as the normalized length of the shortest path in the silhouette. The method of appearance recognition uses kernel density estimation (KDE) of probabilities associated with color/*path-length* profiles and the Kullback-Leibler (KL) distance to compare such profiles with possible models. When there are more than one profile to match in one frame, we adopt multiple matching algorithm enforcing a 1-to-1 constraint to improve performance. Through a comprehensive set of experiments, we show that with suitable normalization of

color variables this method is robust under conditions varying viewpoints, complex illumination, and multiple cameras. Using probabilities from KDE we also show that it is possible to easily spot changes in appearance, for instance caused by carried packages.

Lastly, an approach for constructing a gallery of people observed in a video stream is described. We consider two scenarios that require determining the number and identity of participants: outdoor surveillance and meeting rooms. In these applications face identification is typically not feasible due to the low resolution across the face. The proposed approach automatically computes an appearance model based on the clothing of people and employs this model in constructing and matching the gallery of participants. In the meeting room scenario we exploit the fact that the relative locations of subjects are likely to remain unchanged for the whole sequence to construct more a compact gallery.

KEY-FRAME APPEARANCE ANALYSIS
FOR VIDEO SURVEILLANCE

by

Kyongil Yoon

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2005

Advisory Commmittee:

Professor Larry Davis, Chair/Advisor
Professor Ramani Duraiswami
Professor David Mount
Professor Amitabh Varshney
Professor Kyu Yong Choi

DEDICATION

To my family: Shin-Yeon, Yejin, and Hojin

ACKNOWLEDGEMENTS

I owe my gratitude to all the people who have made this thesis possible. First I'd like to thank my advisor, Professor Larry Davis for giving me invaluable advice and guide. Especially when I was so exhausted and almost gave up, he was kind enough to encourage me to continue and finish the work. Also, I am grateful to the other committee members for their valuable time and advice.

I would also like to thank David Harwood, Yaser Yacoob, and Daniel DeMenthon for working with me on interesting problems for several years. They were very patient and helpful when I was off the track. Without their wonderful guidance and inspiration, it would not be possible to finish this thesis.

I am indebted to many friends and colleagues in the university with whom I have had many fruitful discussions and from whom I have learnt a lot, especially KG-VISA group. Also I do not know how to thank all the people who did not hesitate to help me to capture video data many times: Kyungnam, Bohyung, Jaeyoon, Hyunmo, Seungryul, Minho, Jiksoo, Dongkeun, Joonhyuk, Yooah, Jihwang, Bongwon, Seungjoon, Seungjong, Minkyung and much more.

This dissertation could not have been done without the encouragement and support of my family. My wife, Shin-Yeon Jeon, has been very patient with me during this long time. My two kids, Yejin and Hojin, have been a constant source of love and zest in my life. I also wish to express my gratitude to all other family

members who have been watching and praying for me for long time.

It is impossible to remember everyone, and I apologize to those who I have inadvertently missed. Lastly, thank God!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

There are many ways to understand events in surveillance video data. One of main trends is tracking across frames and acquiring necessary features from tracked objects to recognize events. However, human we can understand events in the video data when viewing only several important “key” frames. For example, Figure 1.1 shows a series of 10 frames which were picked from a 17-second long video clip.

As human, we can conclude that there is an event, “delivery” in this footage based on the following two facts

1. The person who came into the room is the same as the person who left the room.
2. He was carrying a package when he entered the room, but left without it.

In addition to the basic event, it is also possible to infer more things from the video clip, if we carefully analyze the frames. For instance, the frame with timestamp 14.23 shows a human figure at the end of hallway, and it implies there can be a witness of the delivery event. By comparing the first three frames with the

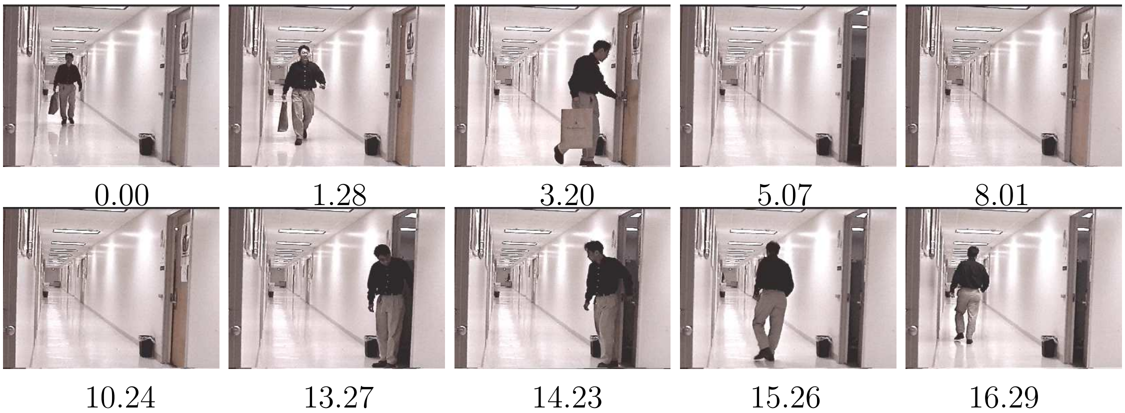


Figure 1.1: These are ten frames of a man who delivers a package to a room. Without tracking him continuously, we know that he approaches with a package, enters a room, comes out, and leaves without the package.

fifth frame (timestamp 8.01), we can see that the door is ajar before he enters the room and there might a security problem with this room. Similarly, after he leaves, the door is not closed perfectly. The frames we can concentrate on for analysis are called key-frames or snapshots. Our research focuses on algorithms for analyzing key frames, as opposed to key frame selection.

1.2 Related Work

Various key-frame extraction methods based on shot boundary detection have been studied citeBoreczkyR96,Girgensohn1999,Smoliar1993 and the purpose of most research is to summarize a full length video in a more compact way. In contrast, we are interested in indentifying interactive frames to which we can apply computationally intensive, but accurate, image analysis. We do not consider the key-frame selection process in this dissertation, although it is a topic for future research.

Background subtraction (BGS) is a typical first step in surveillance. There have been many pixel-based approaches using single Gaussian distribution [30, 60], mixture of Gaussian distributions [54, 26], kernel density estimation based non-parametric technique [19], and so on. Evaluation of some of those algorithms can be found in [7].

While most background models are based on pixels, there have been a few studies employing region or frame information by segmenting an image into uniform size regions or by refining low-level classification obtained at the pixel level [15, 26, 56]. Our region-based background subtraction method is different from these methods in that we employ a color segmentation algorithm to obtain "natural" regions, and then the aggregation of deviation values are used in the detection phase to detect foreground elements at the region level.

Considering the recognition of people, exhaustive studies using biometric features have been done. Biometric features of a person are intrinsic values that are not likely to change under different environments. Some of those features are face [8, 61], gait [4, 10, 62], finger print [6, 23], iris [59, 63], and sound of walking [38, 40] etc. In contrast, non-biometric models of full bodies of people based on appearance have been used for recognition [45] and tracking [18]. We describe an approach to model people's appearance that combines color and intrinsic geometry.

We are interested in identifying changes in people's appearance due to objects that they acquire and deliver. For local change detection, in Haritaoglu's Backpack [24] system, both shape and motion cues are used to locate outlier areas which are usually significantly protruding regions of the silhouette. BenAbdelkader [4] suggested a method to determine whether or not a walking person is carrying an object prior to applying gait recognition to a video sequence. We show how our color appearance model can be used to detect structural changes to people's appearance, and illustrate its use for detecting carried objects.

1.3 Contribution

The contributions of the thesis include:

- We show how pixel-based background subtraction can be improved by introducing segmentation information.
- We propose a new feature, *path-length*, to encode spatial information into color appearance model.
- Comprehensive experiments using the new method show that it outperforms previous approaches that do not consider geometry.
- We propose a method to locate local change areas usually caused by carried packages.
- We propose and perform experiments to construct a gallery of human models for a given video clip or a set of images.

1.4 Overview of Dissertation

In this thesis, we describe a collection of key-frame based appearance analysis algorithms for video surveillance. In Chapter 2, region-based background subtraction is described. This involves hierarchical segmentation of enhanced image using SNF and an improved approach for better background subtraction. In Chapter 3, a new approach to represent an appearance of a person based on *path-length* and an effective way to match them for recognition and local change detection is presented. As an application of the recognition method, we suggest an algorithm to build a compact gallery for given video clips or a set of frame images in Chapter 4. In Chapter 5, conclusion and future work is discussed.

Chapter 2

Region-Based Background Subtraction

2.1 Background Subtraction

Extracting moving objects from a video clip is the first step in video surveillance. The most popular approach to acquiring moving objects, or foreground objects, is background subtraction. In the process of background subtraction (BGS), a reference background model is subtracted from the current image. Although various background models have been suggested, most of them are based on pixels, and segmentation information is not used.

If we have good segmentation information, we can use it to improve the quality of background subtraction. Figure 2.1 contains an example showing how pixel-based BGS can be improved by employing segmentation information.

For a given frame, Figure 2.1 shows the difference between the results of a pixel-based method and region-based method. The image in 2.1(a) is the original frame image, and 2.1(d) is a segmentation image. The codebook BGS algorithm is used in this example [31, 34]. After a training period, background models using codewords are constructed. In the detection period, a deviation value, which is the difference between a new color at a pixel and the nearest codeword in the codebook

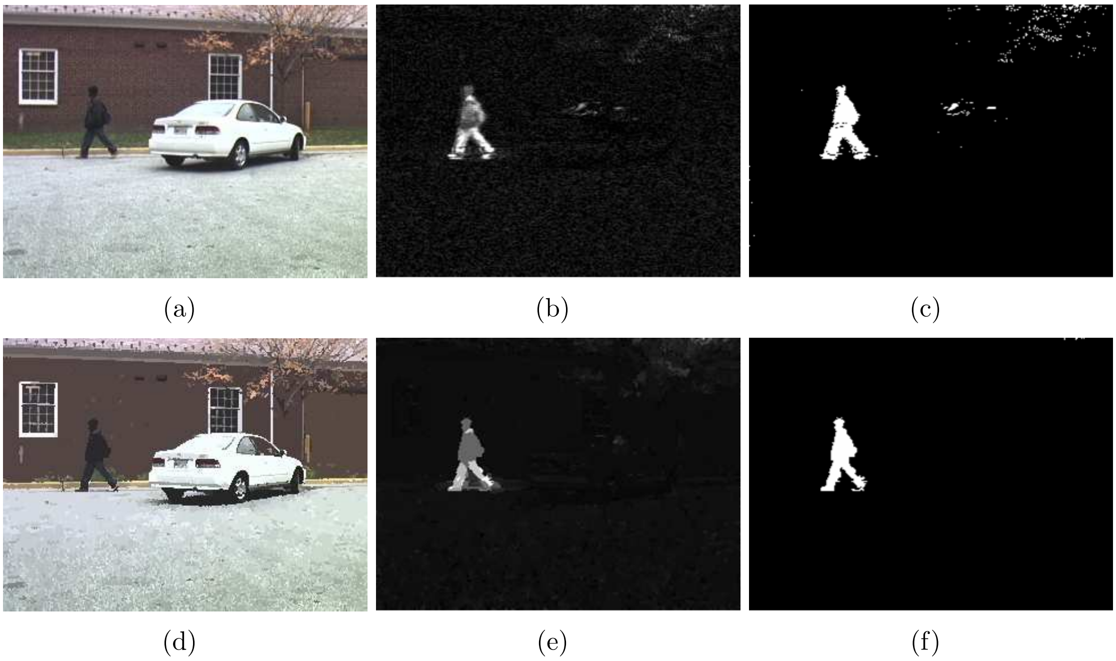


Figure 2.1: Comparison between pixel-based and segmentation-based BGS. Images from pixel-based approach are in the top row. Images using segmentation-based method are in the bottom row.

for that pixel, is calculated for each pixel in the image. Because foreground regions are decided based on these values, it is important to obtain deviation values which can be thresholded easily depending on which region the pixel belongs to.

Figure 2.1(b) is an image showing the pixel-based deviation values. By applying a proper threshold value, Figure 2.1(c) is acquired from 2.1(b). Without any special post-processing, the result has a high false alarm rate. In contrast, 2.1(e) and 2.1(f) were generated based on the segmented image 2.1(d). For each segmented blob, a new value is assigned by aggregating the deviation values of pixels in the blob. The simplest way is averaging the deviation values, and image 2.1(e) shows the average values. 2.1(f) is a thresholded image of 2.1(e). Unlike pixel-based case, there are few false alarms in the image even without any post-processing.

There are two important issues in this approach as following.

1. How to segment a given image

2. How to aggregate the deviation values in a segmented blob

In the following sections, we discuss these two issues.

2.2 Segmentation based on Hierarchical Connected Component Analysis

Image segmentation is a process to partition a given image into different regions such that each region is visually distinct from the others but uniform within itself with respect to some property, such as grey level, texture or color. The problem of segmentation has been an important research field and many segmentation methods have been proposed in the literature. In most segmentation methods, two basic properties of the pixels with respect to their local neighborhood are used: discontinuity and similarity. Methods using discontinuity property of the pixels are called boundary-based methods, whereas methods using similarity property are called region-based methods. Unfortunately, both techniques often fail to produce accurate segmentation results. To improve the segmentation result, a large number of new algorithms which integrate region and boundary information have been proposed [21].

We use the hierarchical segmentation method introduced in [58]. It consists of several steps to generate a basic segmentation and the construction of the full hierarchy of segmentation. The first step is an image enhancement procedure using the symmetric neighborhood filter (SNF). SNF was introduced in [28]; it deblurs edges and reduces local interior variation. By applying SNF in three separate stages, an enhanced image can be obtained [3].

The next step of the segmentation is applying a 1-nearest neighbor filter (1-NN). Single pixel regions rarely can be segmented even under the best circumstances. 1-NN filters these out by replacing them with the mean of its value and the value of an adjacent pixel which is closest to the pixel in terms of color distance.

Connected component analysis (CCA) is the next step, and it builds a basic segmentation. After SNF and 1-NN, the image is primitively segmented. The CCA step works as follows: For each pixel, it is determined whether the difference between an adjacent pixel and itself is smaller that a certain threshold value. When the difference is less than the threshold, those two pixels are given the same label. By using a different threshold value, we obtain a basic segmentation with a different level.

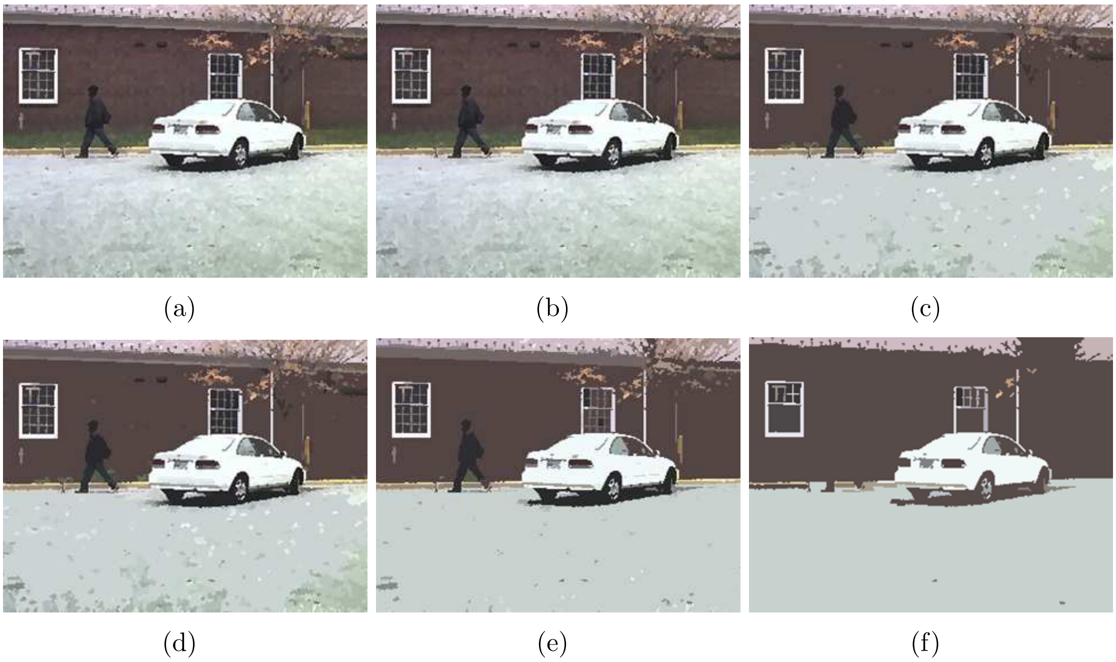
By changing the level in the hierarchy, a different degree of segmentation can be acquired. Figure 2.2 shows each segmentation step and segmentations of three different level for an image used in Figure 2.1.

Image (c) in Figure 2.2 shows the result of 1-NN on (b). Connected component analysis (CCA) is the next step in segmentation, and the last step is to build a basic segmentation. As seen in (c) of Figure 2.2, after SNF and 1-NN, the image is primitively segmented. The CCA step is a labeling process based on this result. For each pixel, it is determined whether the difference between adjacent pixels and itself is smaller that a given threshold. When the difference is less than the threshold, those two pixels will be given the same label. By varying the threshold value, it is possible to get various levels of segmentation. In Figure 2.2, (d) is the result after CCA is done on (c).

The basic segmentation is used to build a hierarchy of segmentations. The average boundary contrast of adjacent regions is computed, and the pair with the smallest average contrast is merged into one region. The boundary contrasts are recalculated and the same procedure is done repeatedly until there are no remaining regions to be merged.

For example, Figure 2.2 shows four images including the basic segmentation which is the result of CCA in Figure 2.2. The basic segmentation, (a), has 1443 components. By changing the level, a different segmentation can be acquired, and (b), (c) and (d) are the examples. They have 728, 301, and 58 regions respectively. (b) is the highest level of segmentation which has a human silhouette. The important thing in this approach is how to pick a correct level of segmentation automatically, since it is not feasible to use the entire hierarchy. In this example, there are 1443 different segmentations and it is hard to decide what the most proper level of segmentation is.

It is important to select a good level of segmentation, since it is not feasible to use the entire hierarchy. The best level of segmentation is chosen manually from the full hierarchy to generate final segmentation image in this paper. It is possible to find a proper level of segmentation based on the two-stage segmentation proposed in [58].



(a) (b) (c)

(d) (e) (f)

Figure 2.2: Images in segmentation process of 2.1(a). 2.2(a) is the image after SNF. 2.2(b) is the image after 1-NN, and 2.2(c) is basic segmentation result after CCA, where there are 1443 basic components. 2.2(d), 2.2(e), and 2.2(f) are segmented image with higher threshold values. Numbers of regions in three images are 728, 301, and 58 respectively

2.3 Aggregation of Deviation Values

In a pixel-based BGS [7], during the training period codewords are assigned to each pixel. In the detection phase, if a pixel in a given image has a greater distance to any codeword than a threshold, the pixel is detected as foreground. In region-based BGS, a similar process should be done in the detection phase for each region. Therefore, the aggregation of distance information of pixels in a region is a critical step. The simplest way to do this is to average the deviation values of all the pixels in a region. In addition to averaging, there are several possibilities such as using root-mean-square (2.1), calculating binomial probability for all the pixels and so on. Although the best aggregation method for all the cases cannot be easily determined, root-mean-square value works reasonably well in most cases.

$$R_{dev} = \sqrt{\frac{1}{n}\sum_{p \in R} p_{dev}^2} \tag{2.1}$$

2.4 Experiments

In Figure 2.1, the difference between pixel-based BGS and segmentation based BGS is shown. In this section, we also present two more examples which are extremely hard to segment using BGS without segmentation. Those are shown in Figure 2.3 and Figure 2.4.

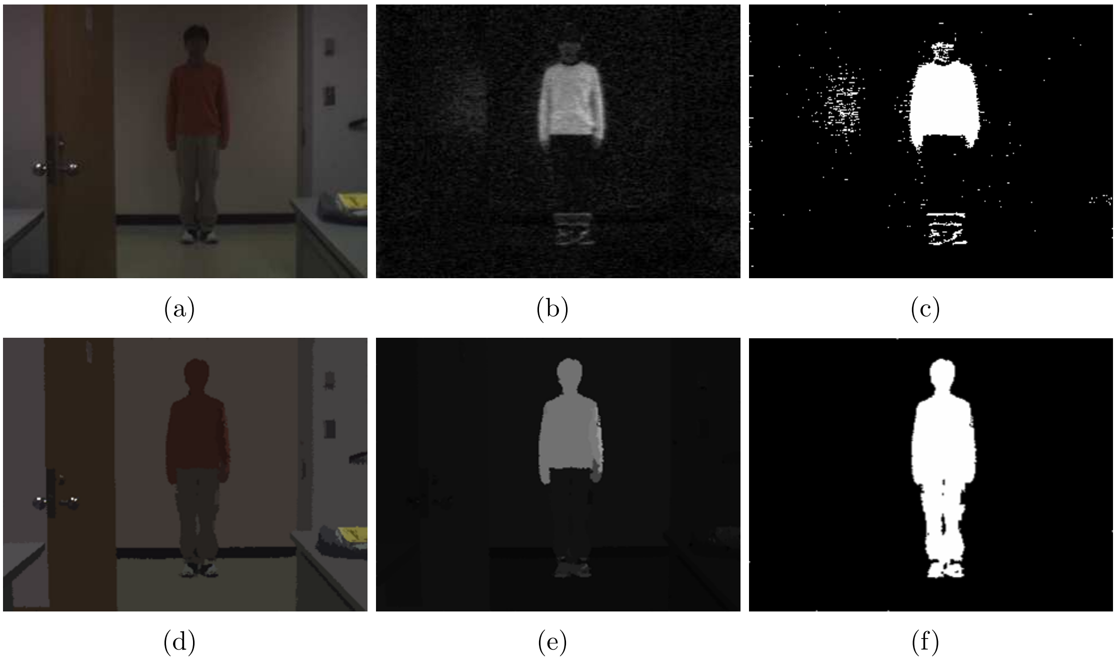
In Figure 2.3, the original image has a human figure which is almost indistinguishable from the background wall because the color of his pants is very similar to the wall. As before, the top row has results from the pixel-based approach, and the bottom row from the segmentation-based approach. The images in the third column present a clear difference of the two BGS methods.

In both cases, we could obtain a segmentation in which the area of pants are segmented into one region. Although there are no differences between deviation values of the pants and the wall, the big deviation values around the ankle area where the wall has a black band makes the aggregated value distinguishable from the background. Note that the pants area is detected as foreground in the segmentation-based BGS. Also, false detection on the door in Figure 2.3(c) has disappeared in Figure 2.3(f), and the falsely detected pixels in Figure 2.4(c) have been successfully removed in Figure 2.4(f).

Figure 2.5, Figure 2.6, and Figure 2.7 show more experimental results, and consistently the segmentation-based method performs better than the pixel-based method. The improvement by the segmentation-based method is described in the caption of each figure.

2.5 Conclusion

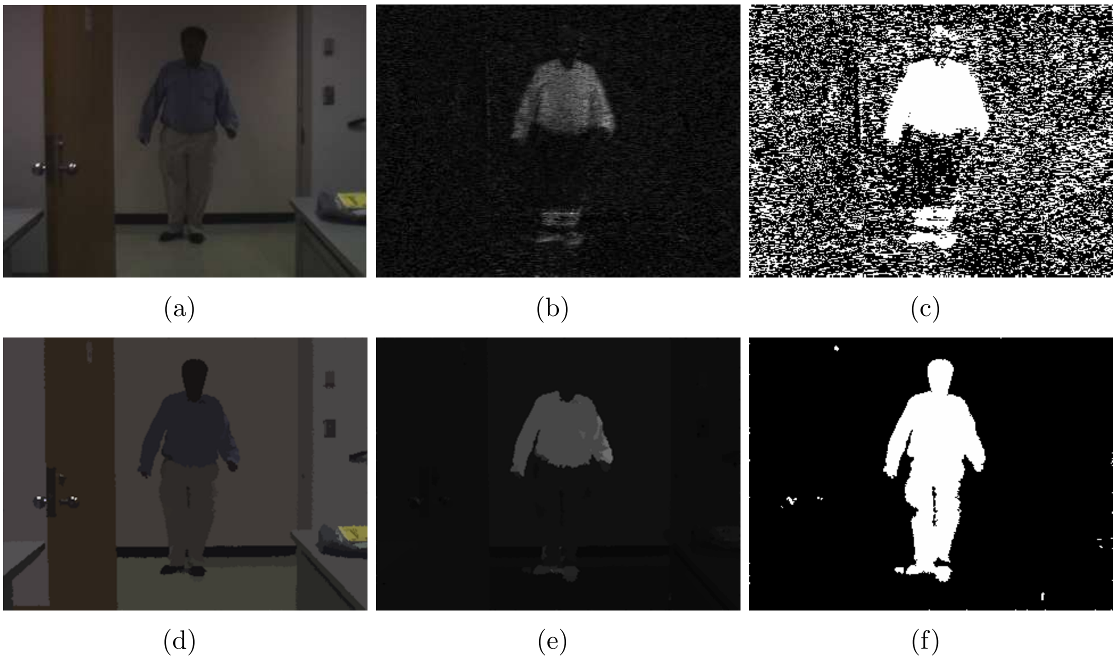
In this chapter we presented a segmentation-based BGS method that improves the performance of pixel-based BGS method by using spatial information. This approach is not limited to any specific BGS algorithm as long as it generates a deviation value for each pixel. We used a hierarchical segmentation method to obtain a good level of segmentation of a frame. We applied this method to the frames that have nearly indistinguishable foreground objects, and showed that improvement can



(a) (b) (c)

(d) (e) (f)

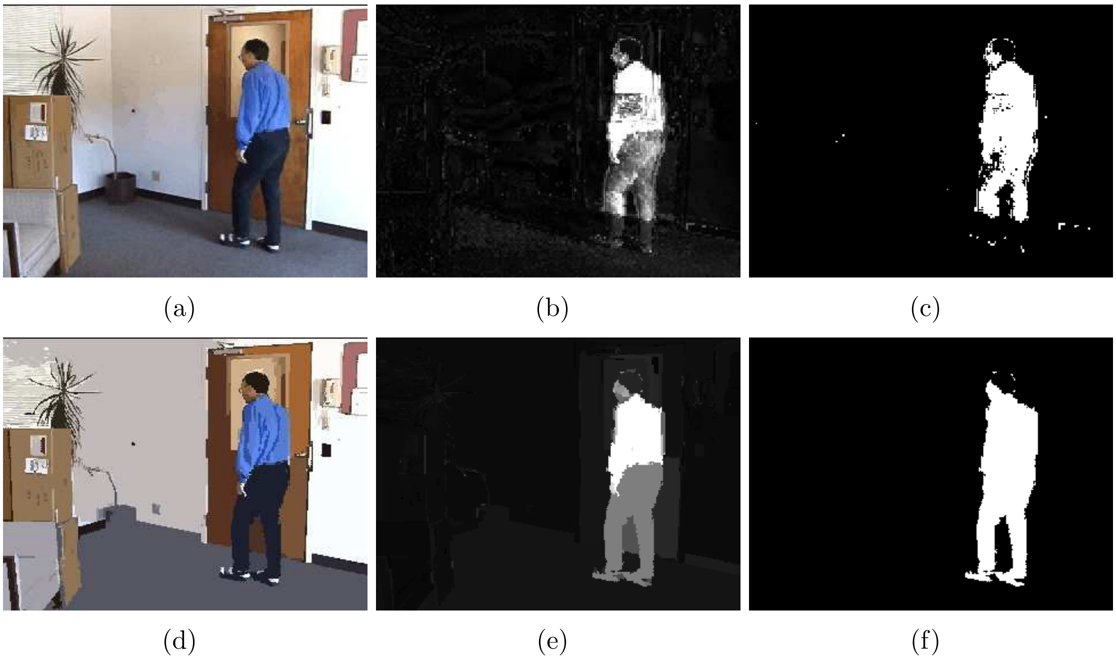
Figure 2.3: An example to compare results of pixel-based BGS and region-based BGS. Images are arranged in the same layout as Figure 2.1.



(a) (b) (c)

(d) (e) (f)

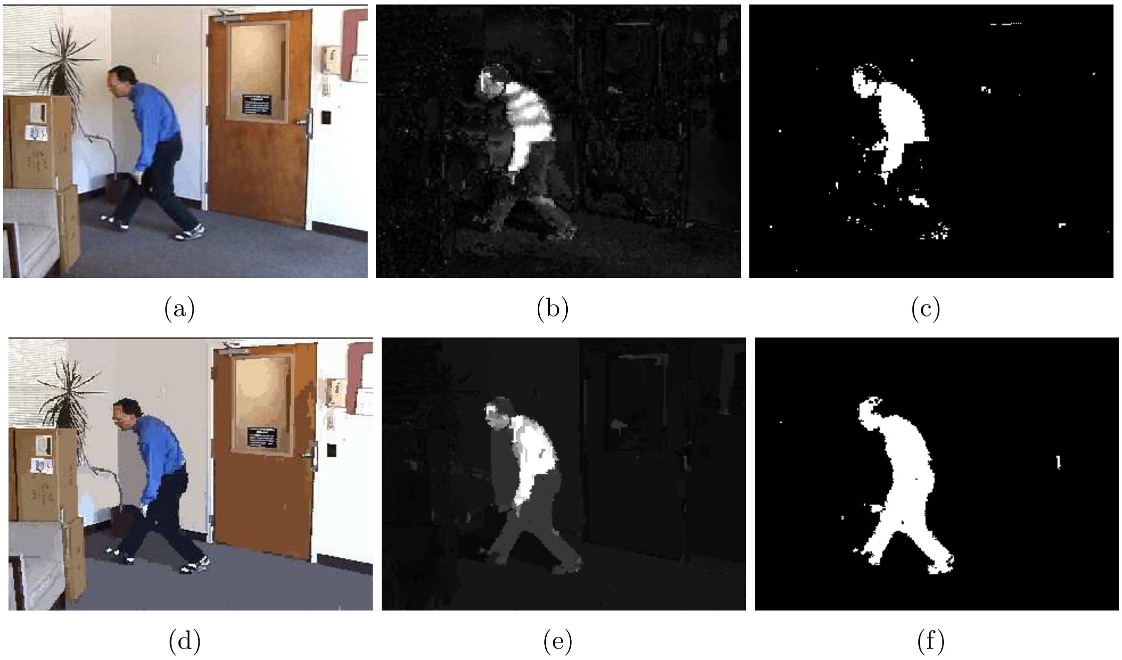
Figure 2.4: Another example to compare results of pixel-based BGS and region-based BGS.



(a) (b) (c)

(d) (e) (f)

Figure 2.5: In (f), area under ankle is detected, and noise pixels in (c) disappeared. The pants and the carpet have almost the same color, and correct detection failed in (c).



(a) (b) (c)

(d) (e) (f)

Figure 2.6: The bottom part of the body is detected successfully in (f), which is not detected in (c).

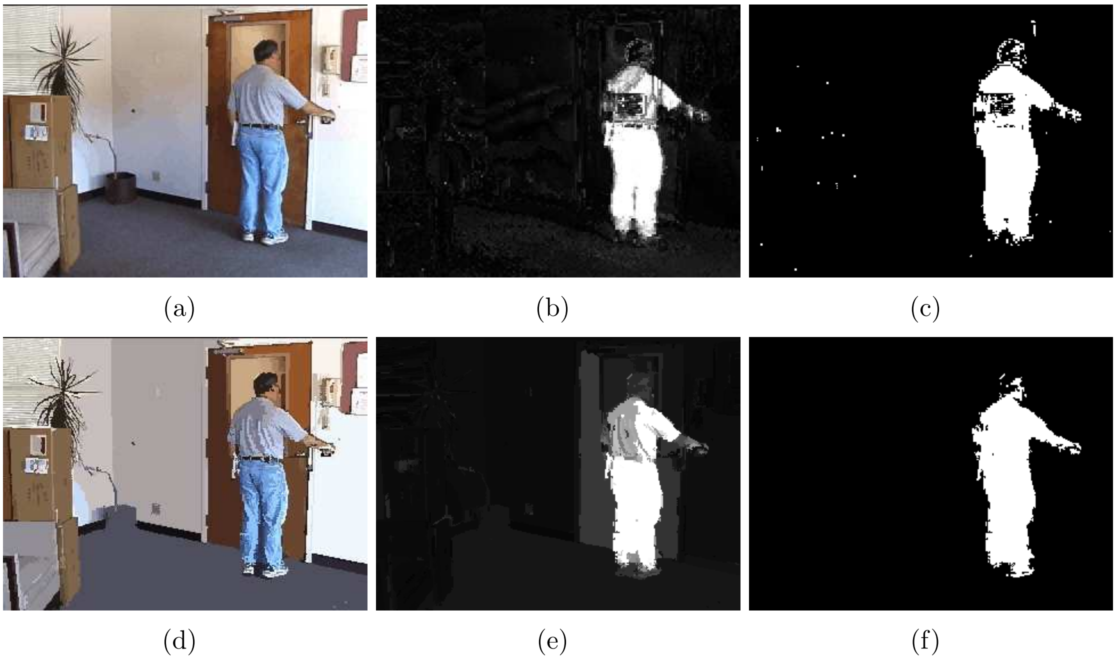


Figure 2.7: The back of the body is detected correctly. It is not detected in (c), because the color of the shirt is similar to the color of inside of the room through the door window.

be obtained compared to pixel-based BGS.

Chapter 3

Appearance-Based Recognition and Change Detection

Recognizing a person is a basic task in video understanding. As seen in Figure 1.1 of section 1.1, if a person is recognized correctly in key-frames, there are many types of events which can be detected. In this section, we present a method for recognizing persons based on their overall extrinsic appearance, regardless of their upright pose. The appearance is that of their visible clothing and bodies seen in silhouette obtained by background subtraction.

Our method of appearance recognition uses kernel density estimation (KDE) of probabilities associated with color/*path-length* profiles and uses Kullback-Leibler (KL) distance to compare such profiles with possible models.

Although there have been many approaches focusing on biometric features in people recognition and identification, it should be possible to use non-biometric features such as appearance for recognition. Modeling the color distribution of a foreground region or a homogeneous blob has been successful in tracking non-rigid bodies such as head, hand, or whole body [5, 13, 42, 43, 60]. Especially, Elgammal used kernel density estimation technique to model background and foreground color distribution. By segmenting a foreground region into three parts, he successfully built models for tracking and recognition [18]. However, in his method, the human body is assumed upright and three separate distributions are modeled and kept separately. We propose appearance modeling based on color/*path-length* (CPL) profile to overcome this issue.

In Nakajima et. al [45], a full-body recognition system based on color and shape features has been suggested. In the system, support vector machine (SVM) classifier was used to categorize appearances, and recognize persons against trained models. They used several features such as color histogram, normalized color histogram, combined histogram of shape and color, and local shape features. They showed that the approach can successfully categorize person in a short period. Their approach starts from the same assumption as ours-that appearance is the key feature for recognition, and all the features are constructed based on appearance of people. However, they did not combine spatial information with color as effectively as we do.

3.1 Appearance Model using Color/*Path-Length* Profile

We represent appearances of persons from silhouettes obtained by background subtraction [30]. In a short period of time, we assume appearances of the same person remain unchanged, except for small, local changes, for instance due to carried packages.

We want to model appearance based on its color as well as spatial information.

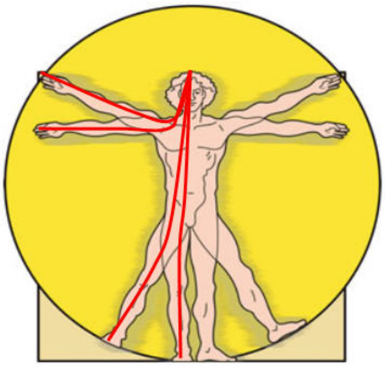


Figure 3.1: This is a simplified drawing of human body by Leonardo da Vinci. The red lines show shortest path inside the body. The *path-length* is the distance from the top of the head to a given point on the path. The *path-length* to the end of hand or foot is relatively unchanged by the motion of the arms and legs.

Elgammal used three blob segmentation, and built separate color distribution for each blob [18]. In his method, head, torso, and bottom parts are segmented by horizontal lines with the assumption that humans are in upright position. Hence, three separate distributions are modeled and kept for one appearance. To build a more general appearance model, we propose a simple, efficient feature *path-length*, which represents spatial information of a pixel. An easy measure of spatial location of parts of the body within a silhouette is height from the ground [44]. However, this varies when there is a motion such as waving arms, bending knees, or bending at the waist.

The *path-length* of a pixel is defined as the normalized length of the shortest path from the top of head to the pixel inside a silhouette. Starting from the top head point, the length of the shortest path can be calculated easily. Once all the points are processed, the length is normalized with respect to the maximum length in the silhouette. By normalizing, *path-length* can be used as scale-invariant feature.

Figure 3.1 shows the idea of *path-length*.

There are three reasons why the top of head is a good base point for *path-length*. Firstly, usually human body and clothing has bilateral symmetry. To combine the distribution at the same *path-length*, the base point should be on the center vertical axis. Secondly, *path-length* should discriminate parts of appearance. In other words, we do not want two different parts to have the same *path-length*. If we choose a middle point such as centroid as the base point, the upper and lower body will have similar *path-length*. Hence, only possible points are either topmost or bottommost points of a silhouette. Finally, the head is usually prominent when tracking a person [25].

To compute the *path-length* of each pixel, we need the foreground segmentation and the head point. We use the code-book based background subtraction algorithm to get the segmentation [30]. After training with a short period of video segments, we can successfully acquire the segmentation of each frame. In Figure 3.8-Figure 3.22, all the models are shown in silhouettes obtained by the background subtraction algorithm.

Once we have the segmentation, we need to locate the head point for each segmented region. As described in [25], the head point can be predicted using the major axis of the silhouette, the hull vertices, and the topology of the estimated body posture. However, we found that the topmost point of the silhouette is the head point in most cases. When there is more than one point at the topmost position, we use the middle point. Although this simple method could be wrong when there are some parts above the head point, it is usually correct with ordinary postures.

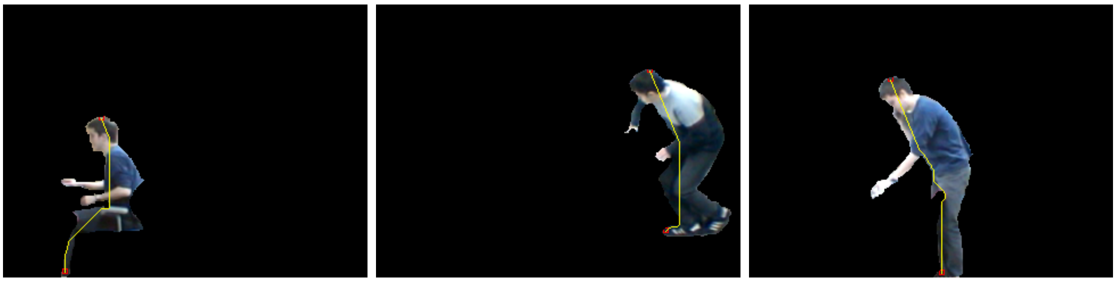


Figure 3.2: Three examples of *path-length* are shown. The longest path is drawn as a yellow line in each image. Silhouettes of these images are acquired manually.

In our implementation, *path-length* of each pixel is computed based on a propagation algorithm. Each pixel is visited in breadth first manner using a queue. When a pixel is dequeued, the *path-length* of all the unvisited neighbor pixels are computed and they are inserted to the queue. Euclidean *path-length* could be defined in special cases [39]; however it is not generally possible to compute the pure Euclidean distance in an arbitrary region without defining junction areas. Hence, the *path-length* is the sum of the distances between adjacent pixels on the path in our case. Some examples are shown in Figure 3.2.

In addition to *path-length*, color information is used to model appearance. Three color components, *RED*, *GREEN*, and *BLUE* can be used directly. It is also possible to separate brightness and color proportions. Brightness is the sum of the three components, as in (3.1) [2].

$$Brightness = RED + GREEN + BLUE. \tag{3.1}$$

Two of three color proportions are used (*red* and *green* usually), since the third is dependent:

$$red = \frac{RED}{Brightness}, green = \frac{GREEN}{Brightness}, blue = \frac{BLUE}{Brightness}. \tag{3.2}$$

To perform more robust comparison, rank order can be used. Especially when multiple cameras are involved, it performs better. Rank order can be applied to either brightness or original color components. In our experiments, we found that the best combinations were as follows:

1. *(path-length, brightness, red, green)*

2. *(path-length, rank of brightness, red, green)*

3. *(path-length, RED, GREEN, BLUE)*

4. *(path-length, rank of RED, rank of GREEN, rank of BLUE)*

Combination 1 includes brightness and two color proportions. Combination 2 replaces brightness with rank of brightness. Combination 3 uses original RGB values. In combination 4, ranks of the three components are used. Several experiments have been conducted to show the result when we use these features, and the results are summarized in section 3.4.2.

To show the improvement due to the introduction of *path-length*, six images for three people in Figure 3.3 are used. The result graph is shown in Figure 3.4. All the images are used both as models and test appearances. Distances are plotted on the graph and the same model is connected as a line. Test appearances are marked on the x-axis. The distance between a test appearance and model represents how different they are from each other. When a test appearance is compared to the same image as a model, the distance is 0. We expect the distance between the appearances of the same person will have smaller distance than other cases.

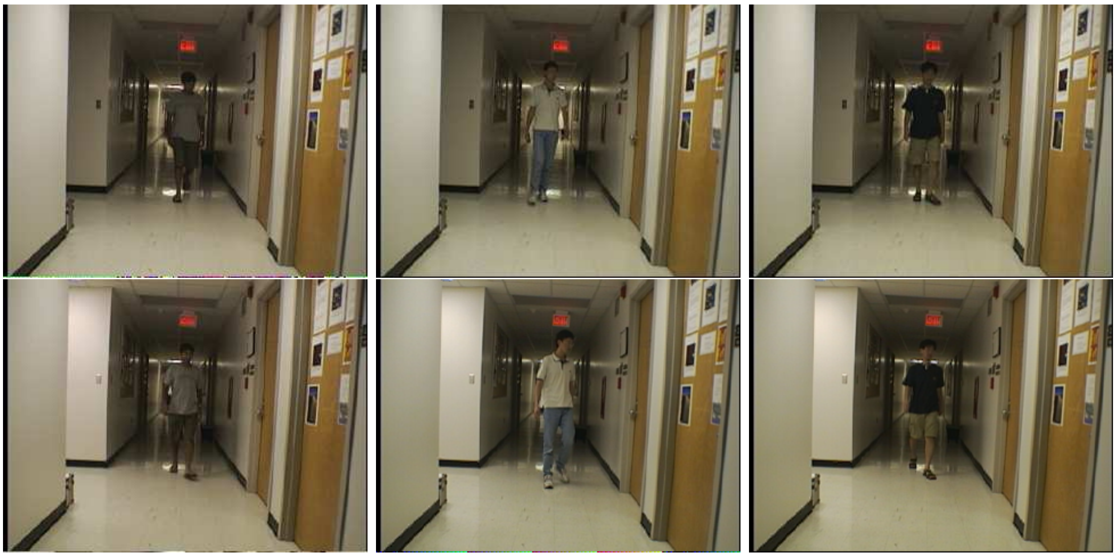


Figure 3.3: For three people, two different images are selected each to test how much the discrimination power improves.

In section 3.2, we will see how to compute distances between appearance. The graph in Figure 3.4 plots 12 tests. The 6 tests done with *path-length* are plotted in solid lines, and the other 6 without *path-length* in dotted lines. It is noticeable that the distances are much larger when *path-length* is used. More comprehensive experiments to show the impact by introducing *path-length* are given in section 3.4.1.

3.2 Distance metric between Models

When a test image is given, we need to compute the distance to an appearance model. In this section, we describe hoow to calculate the distance using KDE and KL.

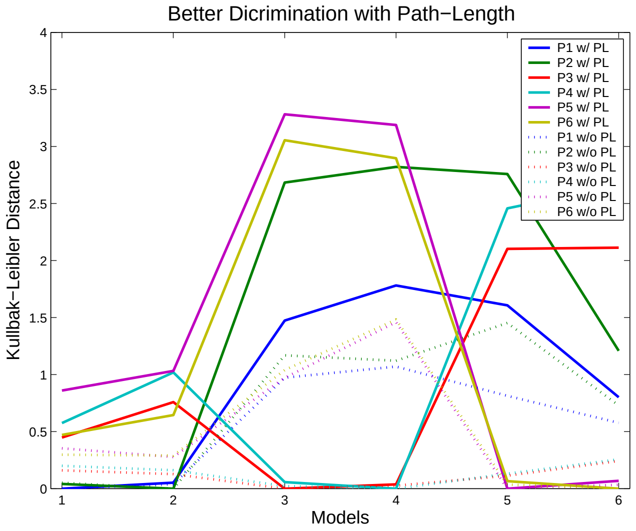


Figure 3.4: Result graph using six images from Figure 3.3. Solid lines are plots when *path-length* is included: *(path-length, brightness, red, green)*. Dotted lines are without *path-length*: *(brightness, red, green)*.

3.2.1 Kernel Density Estimation

Kernel density estimation is a general nonparametric technique to estimate underlying density using data points. In KDE, the probability for given feature x is estimated as

$$\hat{f}(x) = \sum_i \alpha_i K(x - x_i), \tag{3.3}$$

where K is a kernel function centered at data points x_i, $i = 1..n$, and α_i are weighting coefficients. Typically, a Gaussian is used for kernel function, and uniform weights are used, i.e., i = 1/n. The Gaussian is only used as a weighting function around the data points. Theoretically, suitable kernel density estimators can converge to any density function with enough samples [51, 17].

For a given model image $I = \{x_i\}, i = 1..N$ from a distribution $p(x)$, we can build an estimated density function $\hat{p}(x)$ using

$$\hat{p}(x) = \frac{1}{N}\sum_{i=1}^{N} K_\sigma(x - x_i), \tag{3.4}$$

where K_σ is a kernel function with a bandwidth σ such that $K_\sigma(t) = \frac{1}{\sigma}K(\frac{t}{\sigma})$. Although various kernel functions can be used with different properties in the literature, Gaussian kernel is generally used because of its continuity, differentiability, and locality properties [51].

As discussed before, data points have four dimensional feature values. We can estimate the density function using the product of one dimensional kernel based on the following [50].

$$\hat{p}(\mathbf{x}) = \frac{1}{Nh_1...h_d}\sum_{i=1}^{N}\prod_{j=1}^{d} K(\frac{x_j - x_{ij}}{h_j}). \tag{3.5}$$

where d is the dimension of the feature space and h_j is the bandwidth of the j-th kernel.

Using the Gaussian kernels, i.e., $K_\sigma(t) = \frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$ with different bandwidth in each dimension, the density estimation can be calculated for a given point $x = (d, x, y, z)$ as following

$$\hat{p}(d, x, y, z) = \frac{1}{N}\sum_{j=1}^{N} K_{\sigma_d}(d - d_j)\cdot K_{\sigma_x}(x - x_j)\cdot K_{\sigma_y}(y - y_j)\cdot K_{\sigma_z}(z - z_j) \tag{3.6}$$

where σ_d, σ_x, σ_y, and σ_z are bandwidths for each dimension. The probability is defined using all the N data points; however, practically data points which are far from the given point can be ignored. Here, we ignore data more than 3σ away which contribute negligible weight in the implementation.

3.2.2 Bandwidths

The extent to which data points are weighted in the probability calculation depends on the standard deviation of the Gaussian distribution, which is called the bandwidth. Selecting a proper bandwidth is critical to build a correct distribution. When the bandwidth is too large, the number of participating points in computation is greater than necessary. When the bandwidth is too small we will have too many unnecessary modes in the estimated distribution. Appropriate bandwidths are dependent on the characteristics of variables. Experiments to find reasonable bandwidth foe each variable were done and described in section 3.4.3.

3.2.3 Kullback-Leibler Distance

To compare a given appearance to a known appearance model, we need to compute the distance between two distributions which are represented by kernel density estimation. Since the feature space is four dimensional, Kolmogorov-Smirnov test is not appropriate [48]. The distribution is represented as the sum of Gaussian functions, and we can compute estimated probabilities for all the points in the given appearance profile using (3.6) against a model.

Assume that we are to compute the distance between model $M = \{x_i | i = 1, ..., N_p\}$, where N_p is the number of data points in the appearance model, and current instance $I = \{y_i | i = 1, ..., N_q\}$, where N_q is the number of data points in the current instance. The estimated probability distribution of model M is

$$\hat{f}_M(x) = \sum_{i=1}^{N_p} \frac{1}{N_p} K_\sigma(x - x_i) \tag{3.7}$$

and the distribution of the current instance, I is (3.8).

$$\hat{f}_I(x) = \sum_{i=1}^{N_q} \frac{1}{N_q} K_\sigma(x - y_i). \tag{3.8}$$

The distance between the instance and the model can be thought of as the distance between two distributions represented by KDE, $\hat{f}_M(x)$ and $\hat{f}_I(x)$. The two most frequently used methods for comparing two distributions are Chi-Square test and Kolmogorov-Smirnov test [48]. In our case, neither method is appropriate. Since the feature space is four dimensional, the Kolmogorov-Smirnov test is not suitable. The Chi-Square test involves dividing the data points into a number of bins; it is a good approximation when the number of bins is large ($\gg 1$), and number of events in each bin is large ($\gg 1$). However, for human appearance, the color distribution is very skewed, leading to many empty bins.

We instead use the Kullback-Leibler (KL) distance to compare $\hat{f}_M(x)$ and $\hat{f}_I(x)$. Based on the KL distance defined on two probability distributions [33, 36, 14], the distance measure, d, can be defined as

$$d = D(q, p) = \sum_{i=1}^{N_q} q_i \log \frac{q_i}{p_i} \tag{3.9}$$

where $p_i = \hat{f}_M(y_i)$ and $q_i = \hat{f}_I(y_i)$.

In other words, we can think of d as the summation of weighted log likelihood values on all the points in the current instance.

However, when we use all the points in the test image, the cost is prohibitively expensive. Since most data points repeatedly exist in the distribution, it would be possible to estimate the distance with a small number of samples. When we sample points from the given image, we need to reconsider the distance measure, and (3.10) can be used as the estimator of d.

$$\hat{d} = \hat{D}(q,p) = \sum_{i=1}^{n} \log \frac{q_i}{p_i} \tag{3.10}$$

where n is the number of sample points.

As long as the sample points are from the current instance, $\hat{d}$ is an unbiased estimator of d. To prove that $\hat{d}$ is an unbiased estimator, we can introduce an indicator function $\delta(.)$:

$$\delta(i) = \begin{cases} 1 & \text{if } y_i \text{ is sampled} \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

Then we can compute the expectation value of $\hat{d}$ as following.

$$\begin{aligned} E(\hat{d}) &= E(\sum_{i=1}^{n} \log \frac{q_i}{p_i}) \\ &= E(\sum_{i=1}^{N_q} \delta(i) \log \frac{q_i}{p_i}) \\ &= \sum_{i=1}^{N_q} E(\delta(i)) \log \frac{q_i}{p_i} \\ &= c \sum_{i=1}^{N_q} q_i \log \frac{q_i}{p_i} \end{aligned}$$

Notice that $E(\delta(i))$ is cq_i (c is a constant), when we sample points from the given instance. So, when we need to find the best matching model, $\hat{d}$ can be used as an unbiased estimator of d.

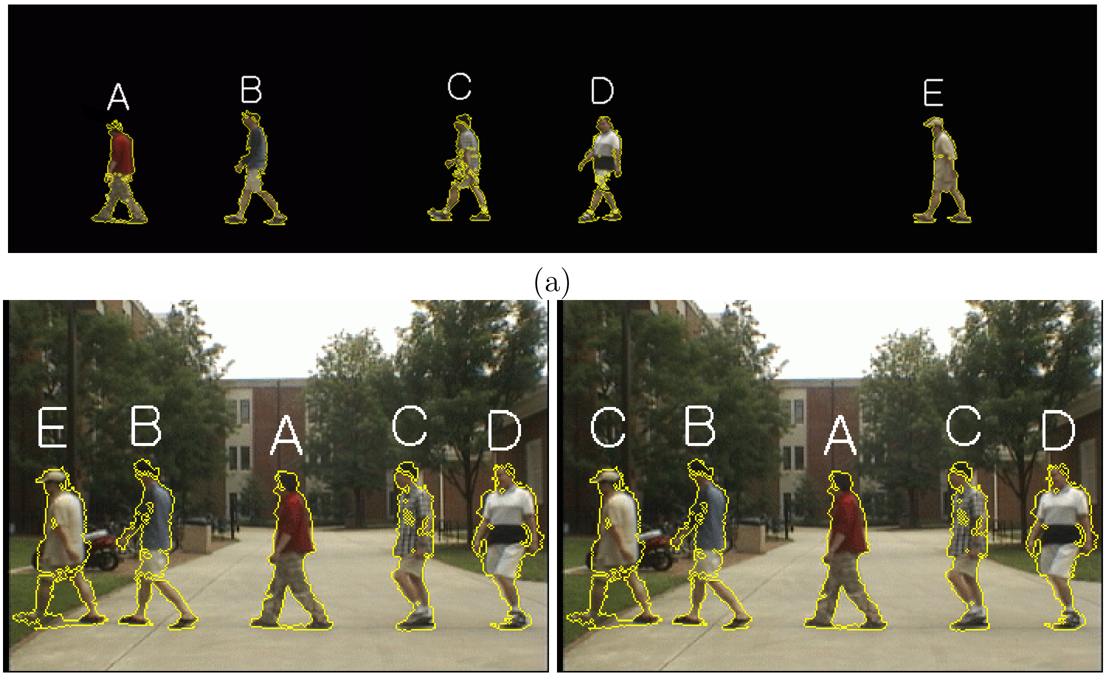
How to select sample points and how many points are used is also important. We tried several different approaches, and they are described in section 3.4.4 with experiments. Practically, when we use more than 50 points, the results are almost equivalent to the result when using all the data points.

3.2.4 Multiple Matching and Gallery Management

When there is more than one person to match in a frame, we have to impose a one-to-one constraint on the matching. Assume that there are two people seen in one frame. It is clear that they can not be matched to a single person. Each person should be matched to a different model. Another example can be found in temporal domain too. If there is a surveillance camera at the door of a room capturing all the people coming in and out, it is desirable to match all the people coming in at the same time to the gallery, since it is clear that no one can appear twice.

Figure 3.5 shows the possible difference when we use one-to-one constraint on the matching. Three top images are used as model appearances. The image (b) shows the result of multiple appearances matching algorithm, and (c) is the result without considering the one-to-one constraint. The first person has different matched model in two methods. In the second method, two different persons (first and fourth) are matched to the model C. We employ the Hungarian method for the multiple appearance matching [35].

It is possible that we have a solution with partial matching. For the instances without matched models, new models can be built based on the appearances and added to the gallery. The gallery is extended while new models are added using unmatched instances. How to maintain the gallery is dependent on the application. For example, when we process all the frames, we would have to remove noisy models. On the other hand, if we have a smaller number of key-frames and they are accurately segmented, we would probably not need to remove any models. Also, we can start



(a)

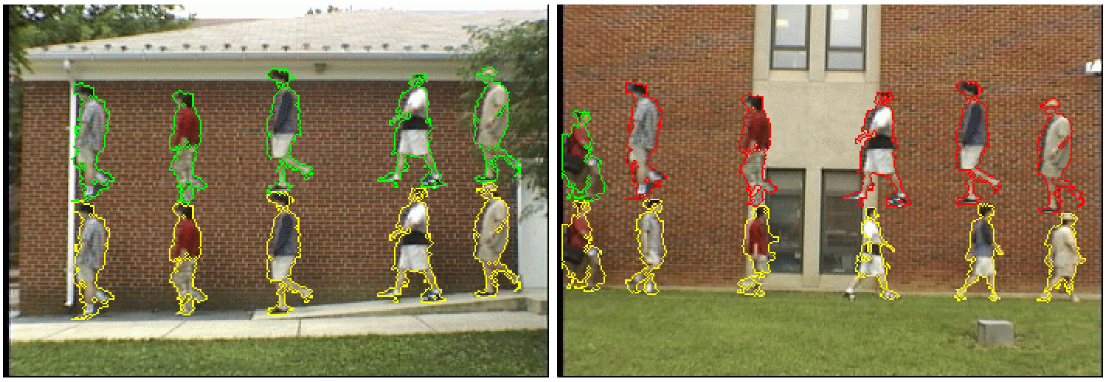
(b)

(c)

Figure 3.5: Comparison of multiple instances matching with one-to-one constraint (b) and greedy matching without the constrain (c). Three images are used for appearance models, and five models are shown in (a).

from an empty gallery and extend it while processing frames.

Figure 3.6 shows a simple case of the dynamic gallery with two key-frames. The first frame is processed with an empty gallery, and five models are built. The second frame has six people. All the five models in the gallery are matched successfully, and the one new person (the first person) is introduced as a new model. A red boundary means that the person is matched to an existing model in the current gallery. For a new model, the green boundary is used. In chapter 4, a more general way to construct a gallery for a given video clip is discussed.



(a) (b)

Figure 3.6: (a) shows five models built in the 1st key-frame. in (b), those five models are matched, and a new model for the first person is built. Red boundary means that the person is matched to an existing model in the current gallery. A new model has green boundary.

3.3 Local Change Detection

When an appearance profile is determined to be an instance of one of our known models, we can analyze the profile further to discover if there are any local changes. This idea is especially useful for the case when either package delivery or pickup occurs. As seen in Figure 1.1, when a person delivers a package, we should be able to determine two things to understand the event.

The first is the recognition of person, which is based on the distance measure discussed in section 3.2.3. The second is to localize the difference. For recognition of the person, a set of small number of sample points from the test appearance is enough. However, we need to examine the probability of all the pixels in the given appearance to see where the difference takes place.

In most cases, the difference comes from a carried package (or the empty hand after package delivery), and we would want to identify the package with a clear boundary. To achieve this, we use the segmentation information of the image based on the hierarchical approach in [27]. Figure 3.7 illustrates how to detect local changes using the profile. Image (a) with no package is used as a model, and image (b) is a test image in which he is carrying a package. In the bottom row, two probability images are shown. (c) is a raw pixel-based probability image, and (d) is regenerated probability image based on segmentation by aggregating the probabilities of pixels in a region.

Image (c) shows low probability in the head part as well as the package area. (The brighter a pixel is, the lower probability it has.) The head part in the model image (a) and (b) are different from each other because of viewing direction.

An important advantage of using segmentation information is that we can localize the difference more clearly. In Figure 3.7 (c), areas of change seem highlighted; however it is hard to extract a clear boundary of the object. On the contrary, in the Figure 3.7 (d), two highlighted areas are detected with clear boundaries as connected components and marked by rectangles. More examples of segmentation based local change detection are shown in section 3.4.7.

3.4 Experiments

Several different sets of experiments have been conducted to evaluate the performance of color/*path-length* based recognition. There are four different categories which have different goals as following.

- Finding the best parameters
  - **Features** *Path-length* encodes the spatial information of appearance, and

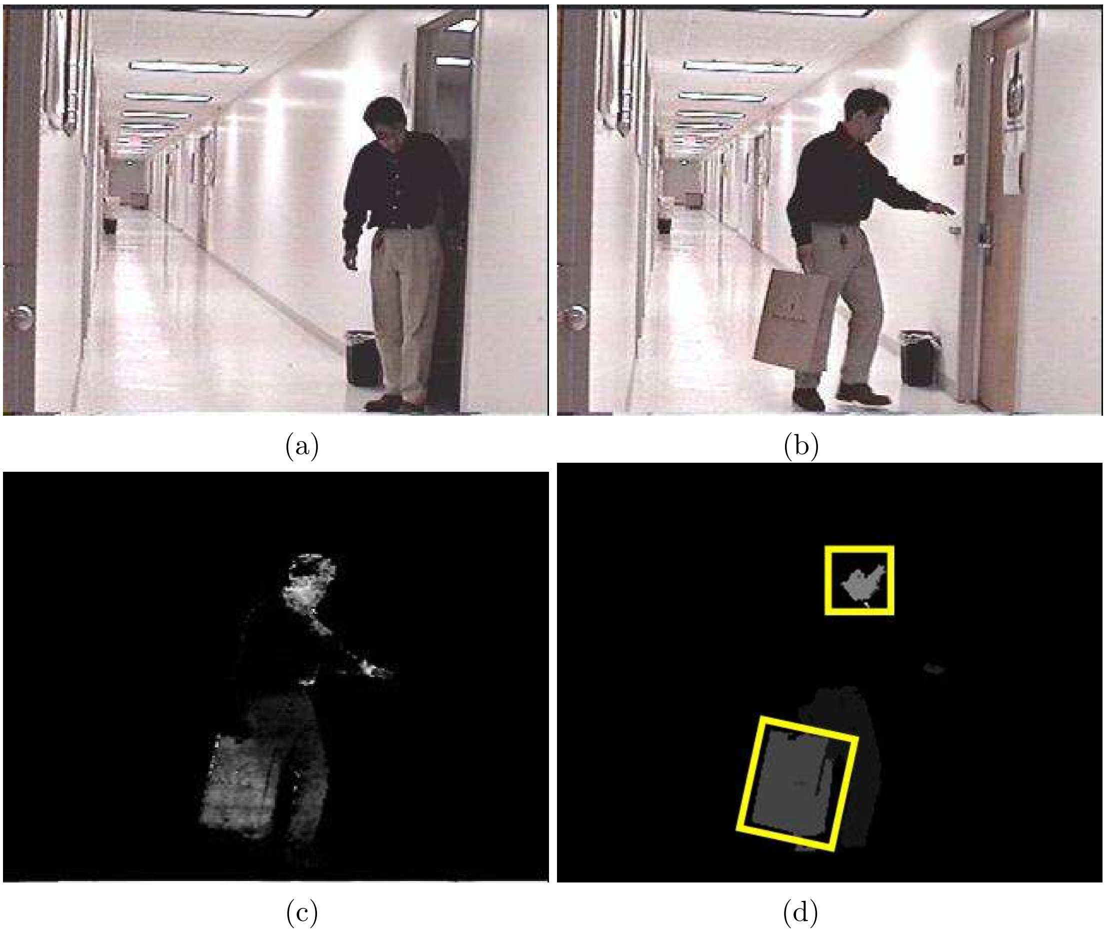

(a)
(b)
(c)
(d)

Figure 3.7: Local change detection using profile. (a) Model, (b) current instance, (c) pixel-based probability image, (d) segmentation-based probability image.

it is clear that brightness and color information should be used together. Several different ways to combine those features are tested. Also, the impact of *path-length* is shown by comparing the result to matching without it.

  - **Bandwidths** To find the appropriate bandwidths, several tests have been conducted using different bandwidths.
  - **Sampling methods** We want to have reliable results with only a small number of samples points.
  - **Multiple matching** When the one-to-one constraint is employed, we can achieve much higher performance. Video clips that have multiple people in each frame are used to show the effectiveness of the one-to-one constraint.

- Robustness test
  - Our approach is quite robust under various environment changes such as illumination changes, view changes, and even multiple cameras. Several video clips with different conditions are used to show the robustness.
- Comparison with other methods
  - The histogram method is a fairly simple and popular method. The performance of our method is compared to the histogram method.
- Local change detection

Table 3.1: Data sets for recognition experiments.

ID	Name	N_f	N_M	N_P	Description
D_1	0720	3707	4	3707	Indoor.
					Various light conditions.
					Frontal, side, and back views.
					16 separate video clips.
D_2	301	571	8	1223	Outdoor. On the grass field.
D_3	302	1070	8	2389	Outdoor. On the grass field.
D_4	303	852	8	1359	Outdoor. On the asphalt road.
					D_2,D_3,D_4: same people.
D_5	304	487	6	487	Indoor. Coming in and out.
D_6	305	691	6	691	D_5, D_6: similar
D_7	m11	311	5	899	Outdoor. Moving sideways.
D_8	m12	442	5	772	D_7, D_8: similar .
D_9	m2	570	6	1948	Outdoor. Moving sideways.
					Six people. Small appearances.
D_{10}	m3	467	5	1422	Outdoor. Moving sideways.
D_{11}	RedWall	1797	6	1797	3 people 6 appearances.
D_{12}	JVC-empty	1368	5	1368	Indoor. Two cameras (JVC, Sony)
D_{13}	SONY-empty	1445	5	1445	Empty hand & folder.
D_{14}	JVC-package	1305	5	1305	D_{12}-D_{15}: 20 clips
D_{15}	SONY-package	1322	5	1322	
D_{16}	Mcam1	608	4	608	Outdoor.
D_{17}	Mcam2	305	4	305	Multiple surveillance cameras.
D_{18}	Mcam3	608	4	608	

- Several experiments have been performed which successfully locate the package area.

To perform the experiments, several data sets were captured and used as in Table 3.1. For each data set, N_f is the number of frames, N_M the number of models, and N_P the number of foreground regions. Three collections of sets are used for multiple camera cases. D_{12} and D_{13} were captured simultaneously using two cameras. D_{14} and D_{15} were captured with the same gear. The last three sets, D_{16}, D_{17}, and D_{18} were captured using three different surveillance cameras.

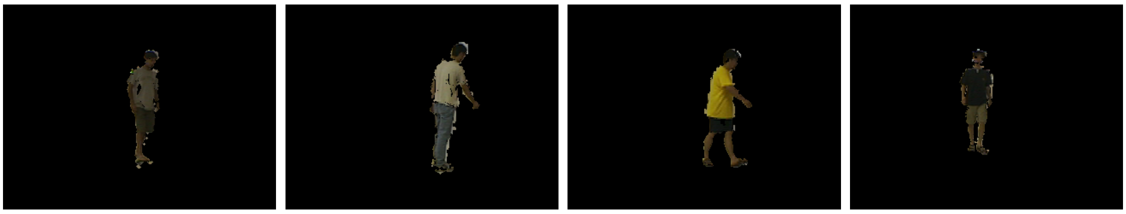


Figure 3.8: Model images from the test set D_1

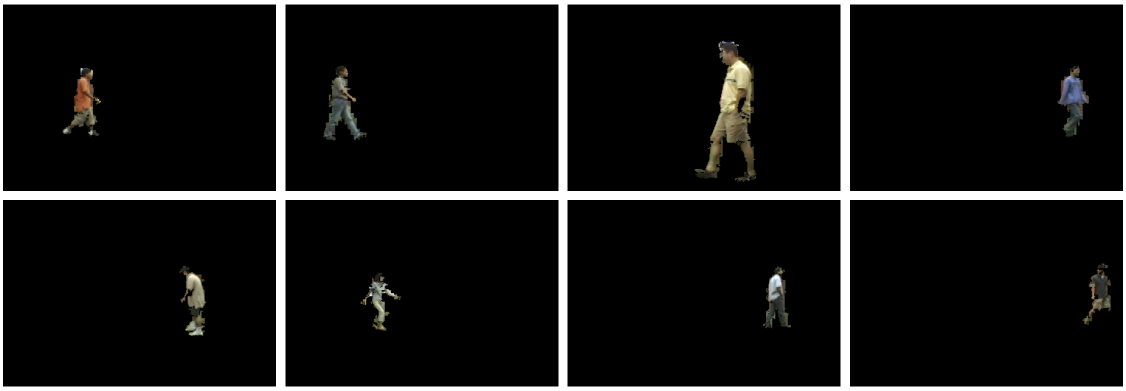


Figure 3.9: Model images from the test set D_2

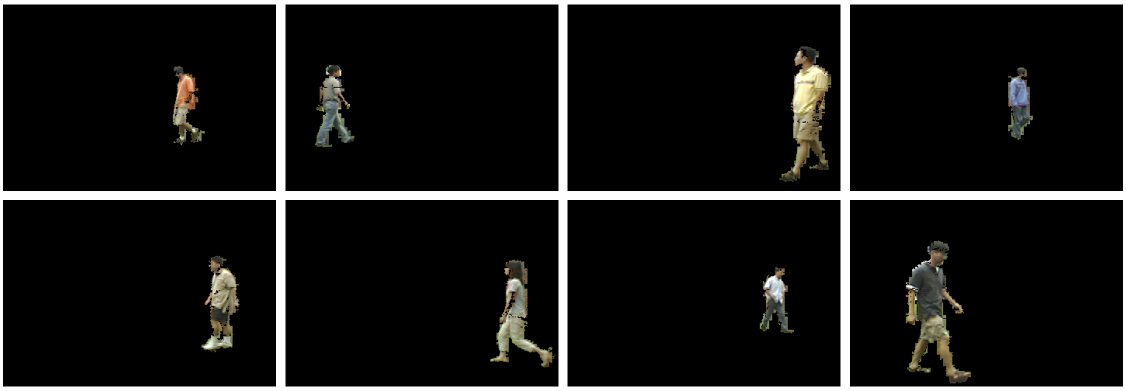


Figure 3.10: Model images from the test set D_3

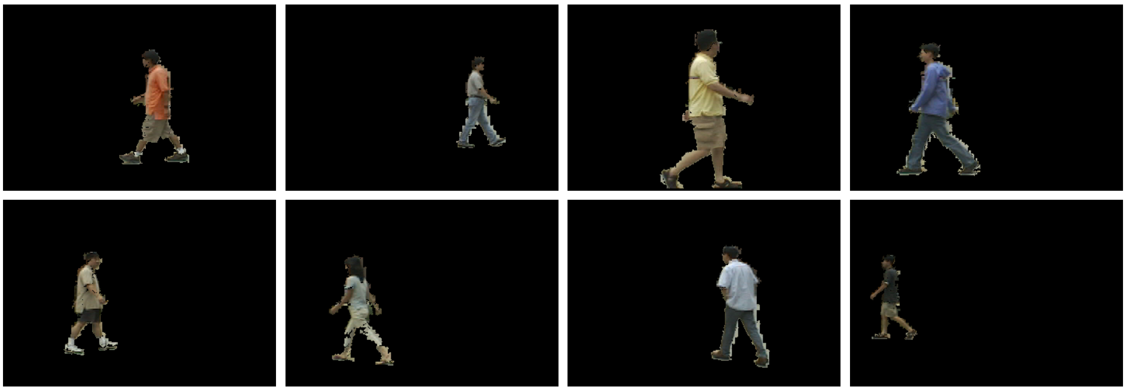


Figure 3.11: Model images from the test set D_4

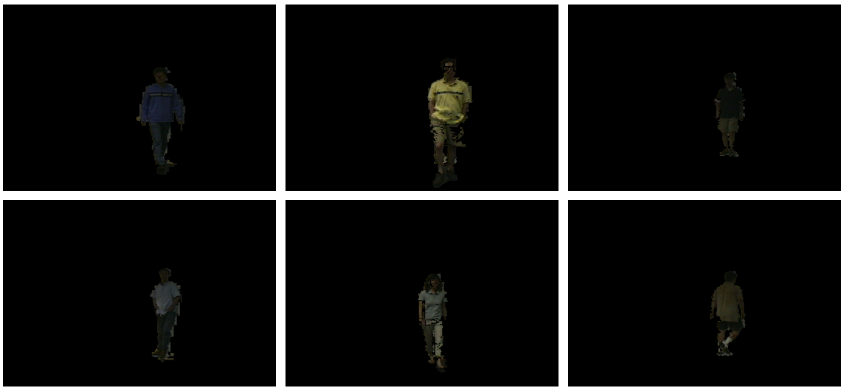


Figure 3.12: Model images from the test set D_5

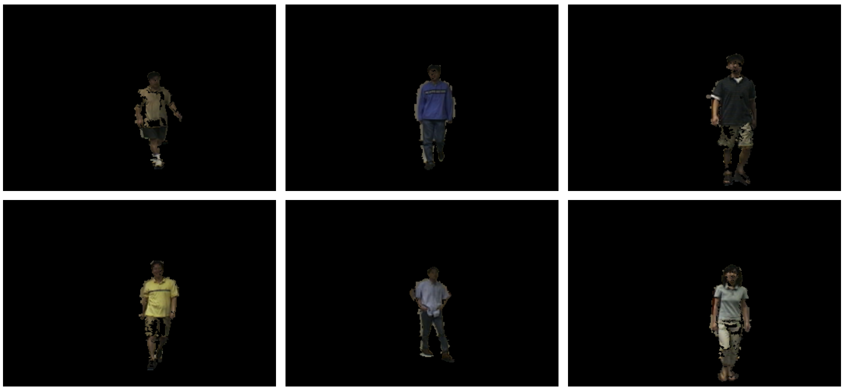


Figure 3.13: Model images from the test set D_6

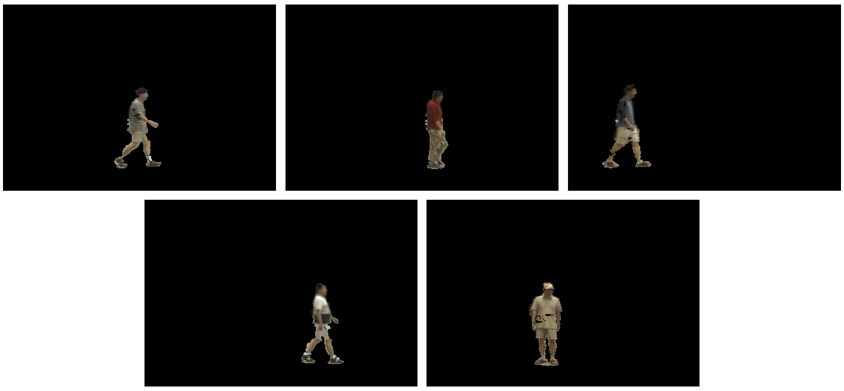


Figure 3.14: Model images from the test set D_7

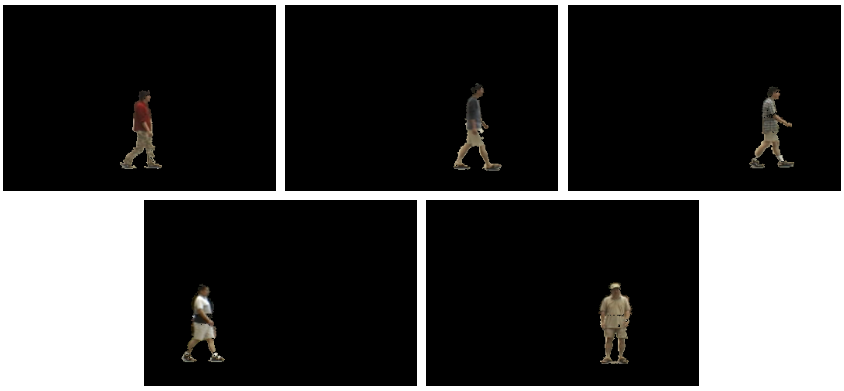


Figure 3.15: Model images from the test set D_8

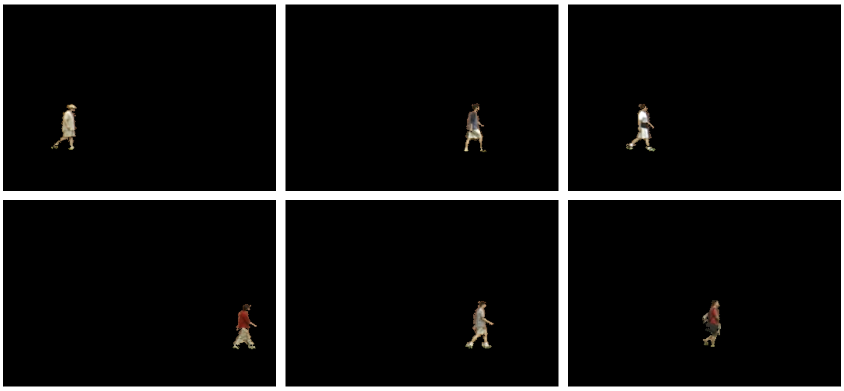


Figure 3.16: Model images from the test set D_9

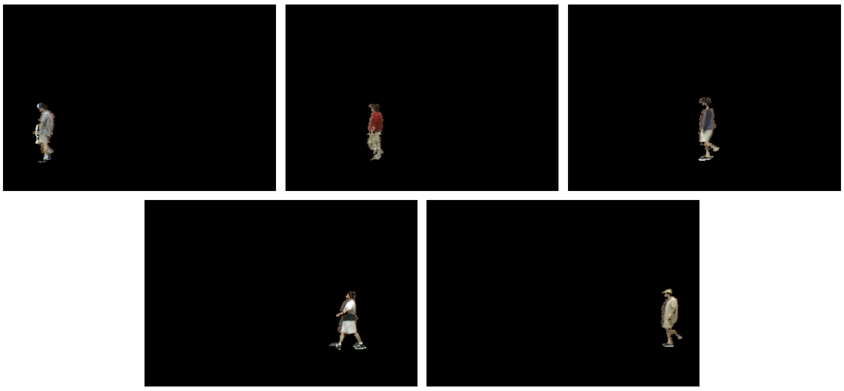


Figure 3.17: Model images from the test set D_{10}

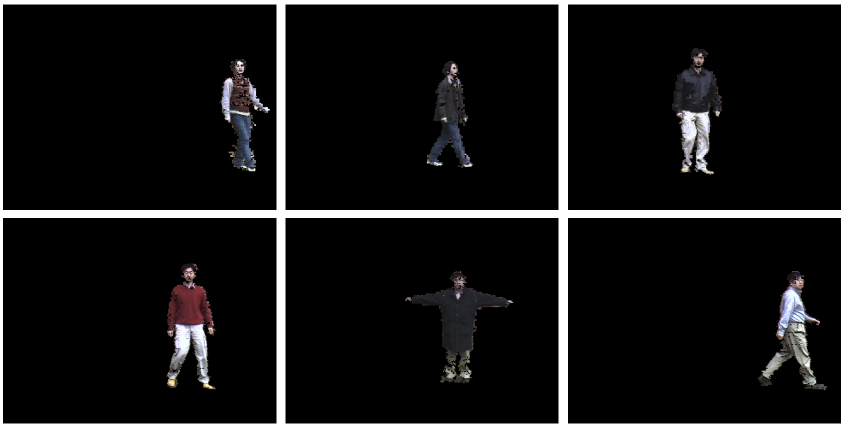


Figure 3.18: Model images from the test set D_{11}

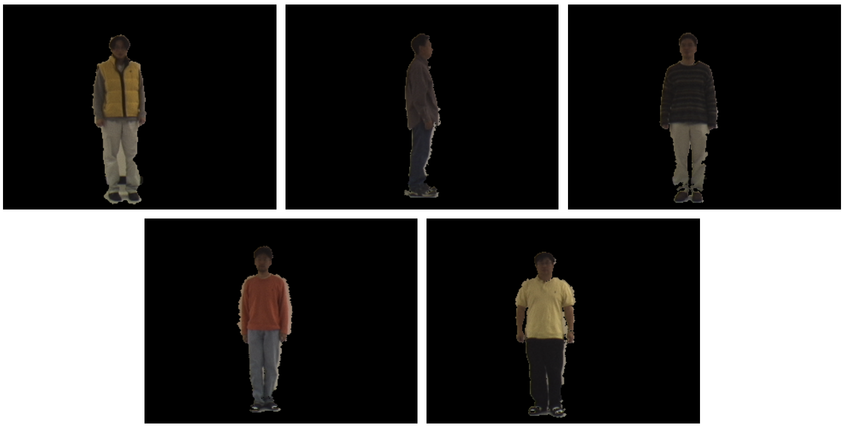


Figure 3.19: Model images from the test set D_{12}. These models can be used for D_{13}, D_{14}, and D_{15}.

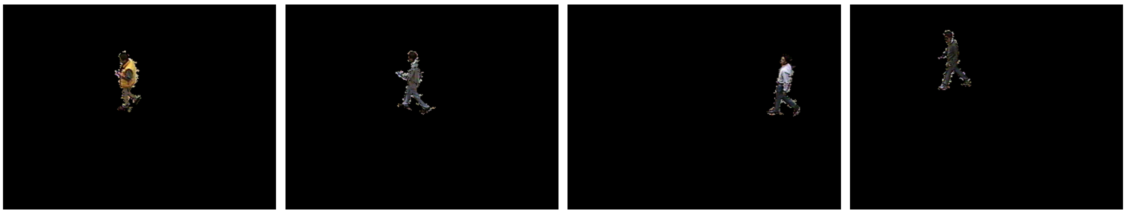


Figure 3.20: Model images from the test set D_{16}

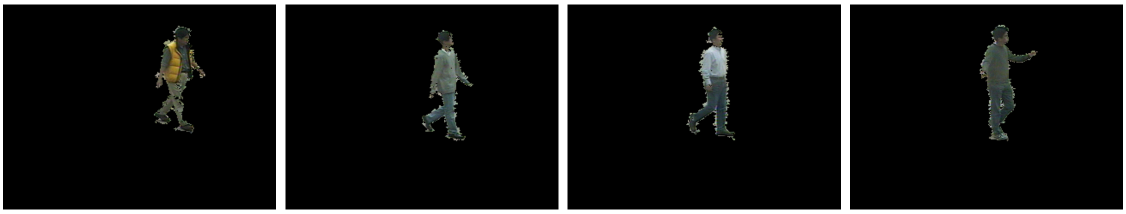


Figure 3.21: Model images from the test set D_{17}

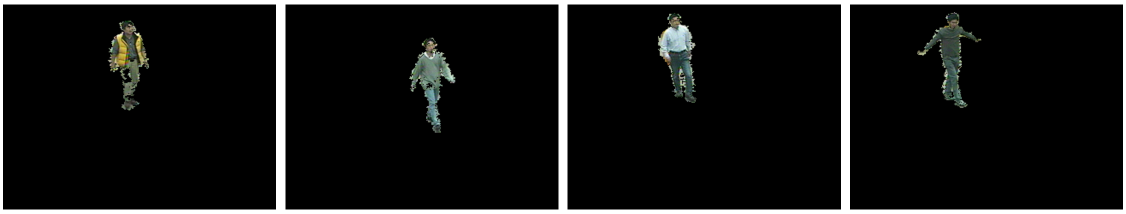


Figure 3.22: Model images from the test set D_{18}

Table 3.2: Matching results showing the impact of *path-length*.

Data Set	Number of Frames	Number of Appearances	Feature	Incorrect Matching	Matching Rate
D_1	400	400	With PL	0	100%
			Without PL	0	100%
D_2	571	1223	With PL	25	98.0%
			Without PL	132	89.2%
D_3	1070	2389	With PL	155	93.5%
			Without PL	331	86.0%

3.4.1 Impact of *Path-Length*

To show the impact of spatial information encoded in *path-length*, we tested the same frame set with two features as following.

- Feature1: (*path-length*, brightness, red, green)
- Feature2: (brightness, red, green)

As seen, the two features are the same except the second doesn't have the *path-length* variable. We use data set D_1, D_2, and D_3, and Table 3.2 summarizes the result. The result shows that employing *path-length* gives better performance when using the set D_2 and D_3 (8.8% for D_2, 7.5% for D_3). Both tests with set D_1 have perfect matching. In D_1, only one person appears in each frame, and four people have clearly different appearances from each other. The models used in D_1 test are in Figure 3.8. Only 400 frames (100 frames for each model) were used out of 3707 frames since most of frames have the same structure.

In D_2 and D_3, more than one person show in frames randomly. Also the background subtraction results are not as good as in D_1 because they are outdoor videos and have much more clutter. As we can see in Figure 3.9 and Figure 3.10, models have very jagged boundaries and it is challenging to match an appearance to a correct model.

Figure 3.23 shows graphs of distances between models and appearances. For each data set, we selected several frames that have all the possible appearances, and generated graphs for both cases. As for set D_1, there is little difference in both cases. However, for set D_2 and D_3, we find it is hard to find the correct models for some appearances. For instance, $Person5$ and $Person6$ (Figure 3.9) have similar color distributions, and if we do not employ *path-length*, we can not easily tell the difference between two. Figure 3.24 shows some frames that have incorrect matches when we use features without *path-length*.

3.4.2 Feature Selection

As discussed in section 3.1, the best features we found after trying many different combinations are as following.

1. *(path-length, brightness, red, green)*

2. *(path-length, rank of brightness, red, green)*

3. *(path-length, RED, GREEN, BLUE)*

4. *(path-length, rank of RED, rank of GREEN, rank of BLUE)*

To show the results using the features, data set D_1, D_6, D_{12}~D_{15} are used. The initial gallery was built with 4 models from D_1, 6 models from D_6, and 5 models from D_{12}. Since D_{13}, D_{14}, and D_{15} have basically the same scenes as D_{12}, no models

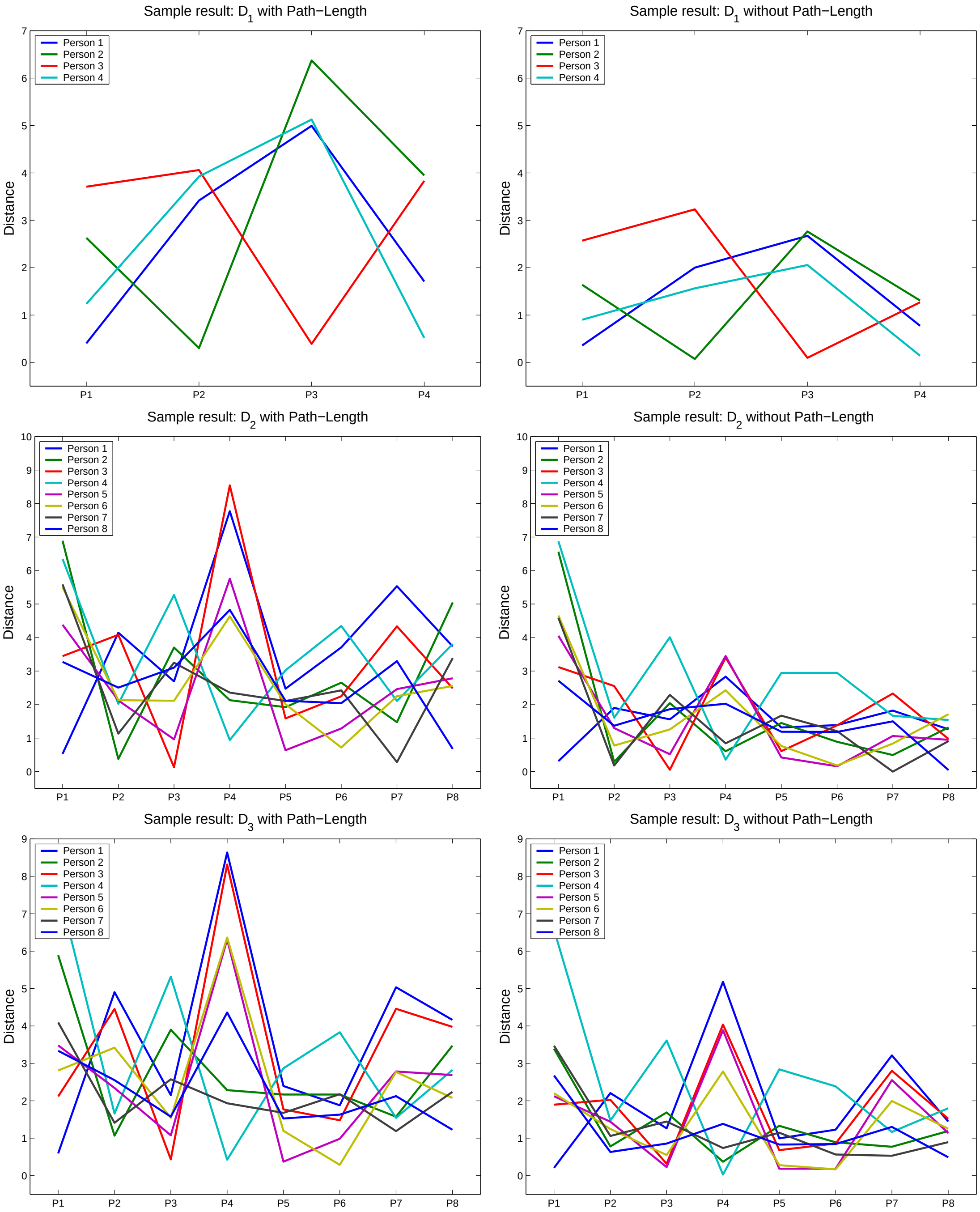


Figure 3.23: Three pairs of graphs to show the impact of *path-length*. The first pair of graphs from D_1 does not show big difference. The second and the third pair show that we can have more discrimination power when we use *path-length*.

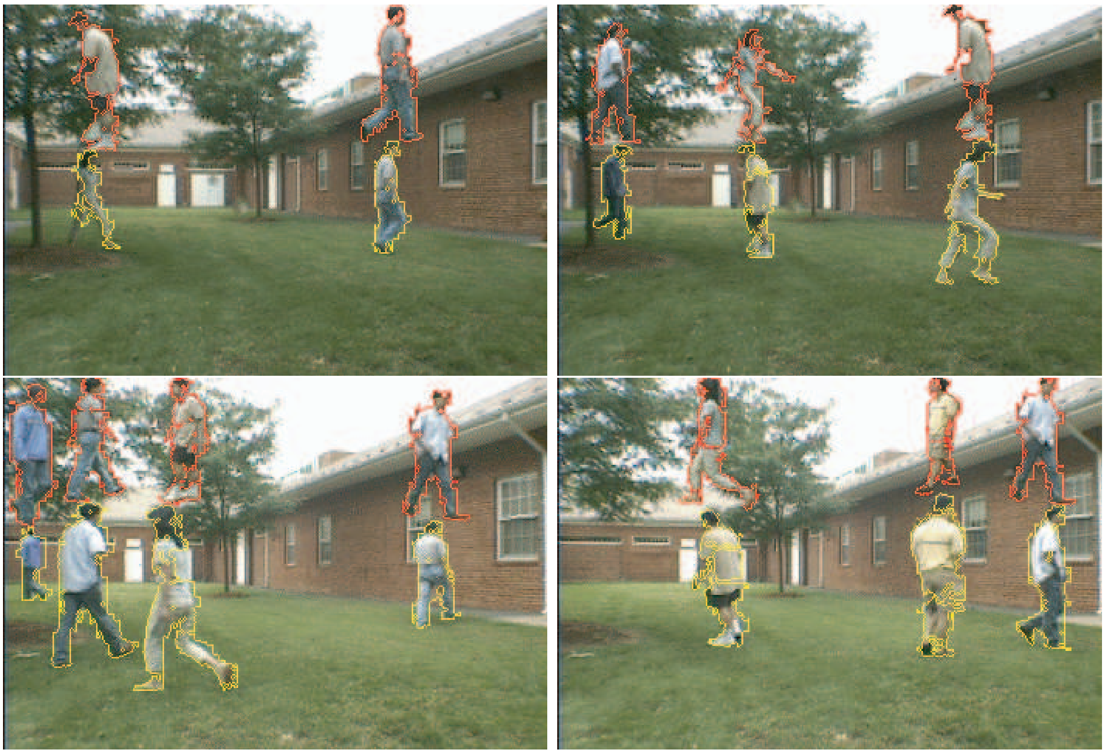


Figure 3.24: Example frames that have mismatched appearances when features without *path-length* are used. The first and second images are from D_2, and the third and fourth images are from D_3. In both data sets, $Person5$ and $Person6$ are not clearly distinguishable.

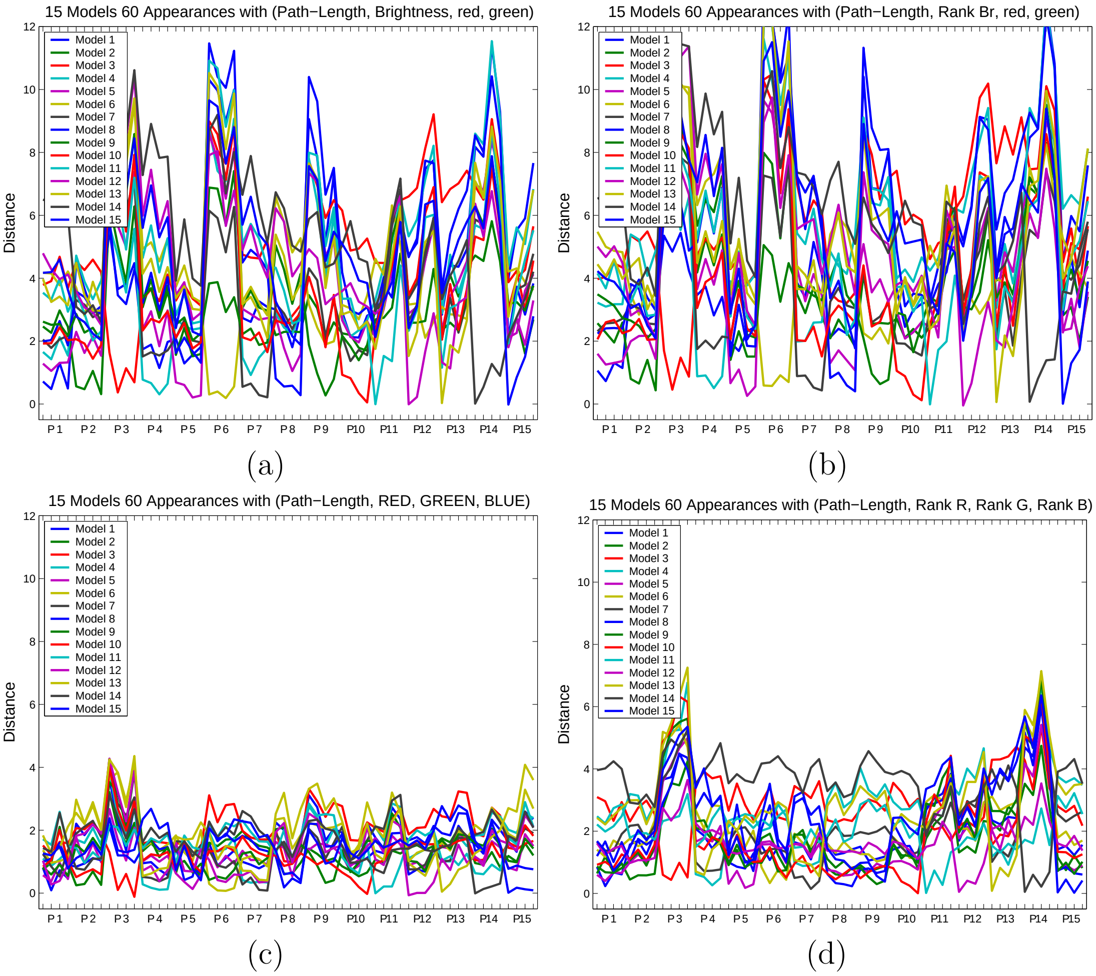
from those sets were used. Figure 3.25 shows the results of the experiment. For each model, we selected 4 frames so that we have 60 test images in total. We arranged them in order of models so that the first four images have the instance of the first model, the next four images have the instance of the second model and so on. Hence, the graph in Figure 3.25 have 60 instances marked on X axis. Each group of adjacent four instances are expected to match the same model.

Experiments are done with four features sets, and four graphs are shown in Figure 3.25. It is clear that the first and the second graphs have better result. However, we should notice that it shows an interesting result for the D_{12}˜D_{15} (right bottom part of the last graph). Four images chosen for the sets are from two different cameras, and it seems desirable to use the feature, *(path-length, rank of RED, rank of GREEN, rank of BLUE)*, when multiple cameras are involved. Although different cameras would be expected to have different pixel values, the relative order (rank order) are expected to remain the same in an appearance. More experiment results are shown in section 3.4.6.

3.4.3 Bandwidths

In this experiment, we try to find appropriate bandwidths for each variable. We use the feature, (*path-length*, brightness, red, green) in the experiments of this section and following sections unless specified.

To find a good bandwidth for each variable, we change the bandwidth of one variable while the bandwidths of the other variables are fixed. We use 10 models from data set D_1 and D_6 and 40 test images which is the subset of data used



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

Figure 3.25: Result graphs showing the best four feature sets. (a):(*path-length*, brightness, red, green), (b):(*path-length*, rank of brightness, red, green), (c):(*path-length*, RED, GREEN, BLUE), (d):(*path-length*, rank of RED, rank of GREEN, rank of BLUE)

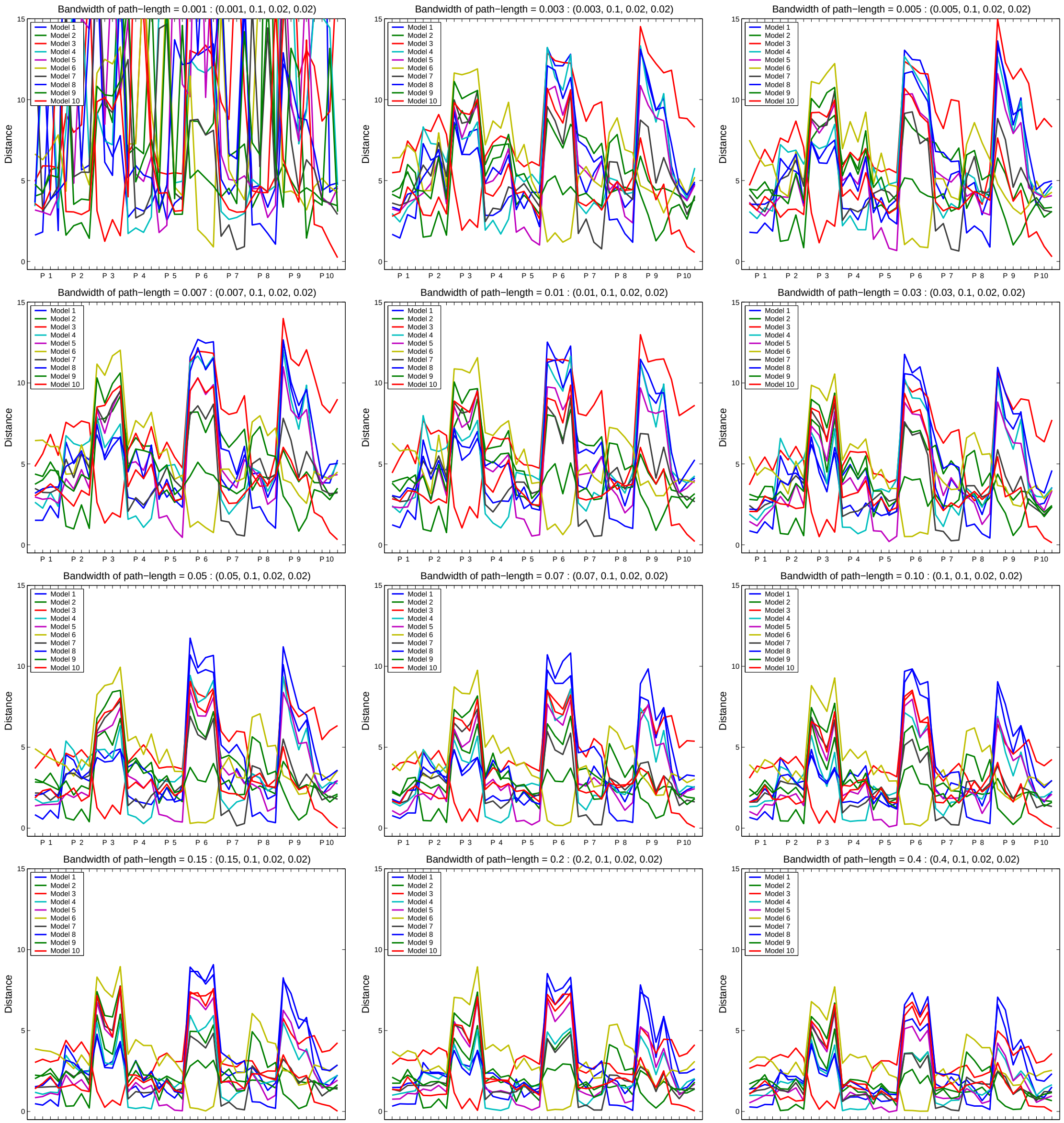
in section 3.4.2. The bandwidths we tested are 0.001, 0.003, 0.005, 0.007, 0.01, 0.03, 0.05, 0.07, 0.1, 0.15, 0.2, and 0.4. Figure 3.26, Figure 3.27, Figure 3.28, and Figure 3.29 are four sets of result graphs that have varying bandwidths for each variable, *path-length*, brightness, red, and green.

Figure 3.26 show results when we change the bandwidth of *path-length* while the bandwidths of the other variables remain unchanged. We can recognize that most cases have almost identical results as long as the bandwidth is greater than 0.005. This is because most appearances in the data set are in the upright pose. We can safely use 0.05 or 0.07 as the bandwidth of *path-length.* Figure 3.27 show results with brightness. Because the brightness is not an intrinsic property of an object [29], it is mainly affected by the illumination condition and exhibits bigger variation. We can predict that the appropriate bandwidth would be greater than other variables, and the graphs show that we need at least a bandwidth of 0.07 to generate acceptable results.

On the contrary, color is an intrinsic characteristics of an object, and a much smaller bandwidth should be appropriate to generate good matching results. As seen in Figure 3.28 and Figure 3.29, 0.01 or 0.03 make the result acceptable.

In conclusion, we summarize the proper bandwidth for each variable as follows:

- *Path-length*: 0.05-0.07
- *Brightness, rank of brightness*: 0.07-0.1
- *Color proportion (red, green)*: 0.02
- *Original color (RED, GREEN, BLUE) or its rank*: 0.07-0.1

 Bandwidth of path−length = 0.001 : (0.001, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.003 : (0.003, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.005 : (0.005, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.007 : (0.007, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.01 : (0.01, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.03 : (0.03, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.05 : (0.05, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.07 : (0.07, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.10 : (0.1, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.15 : (0.15, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.2 : (0.2, 0.1, 0.02, 0.02)

Bandwidth of path−length = 0.4 : (0.4, 0.1, 0.02, 0.02)

Figure 3.26: Varying bandwidth of *path-length* while fixing bandwidths of other variables. Except for 0.001 and 0.005, most graphs have identical results.

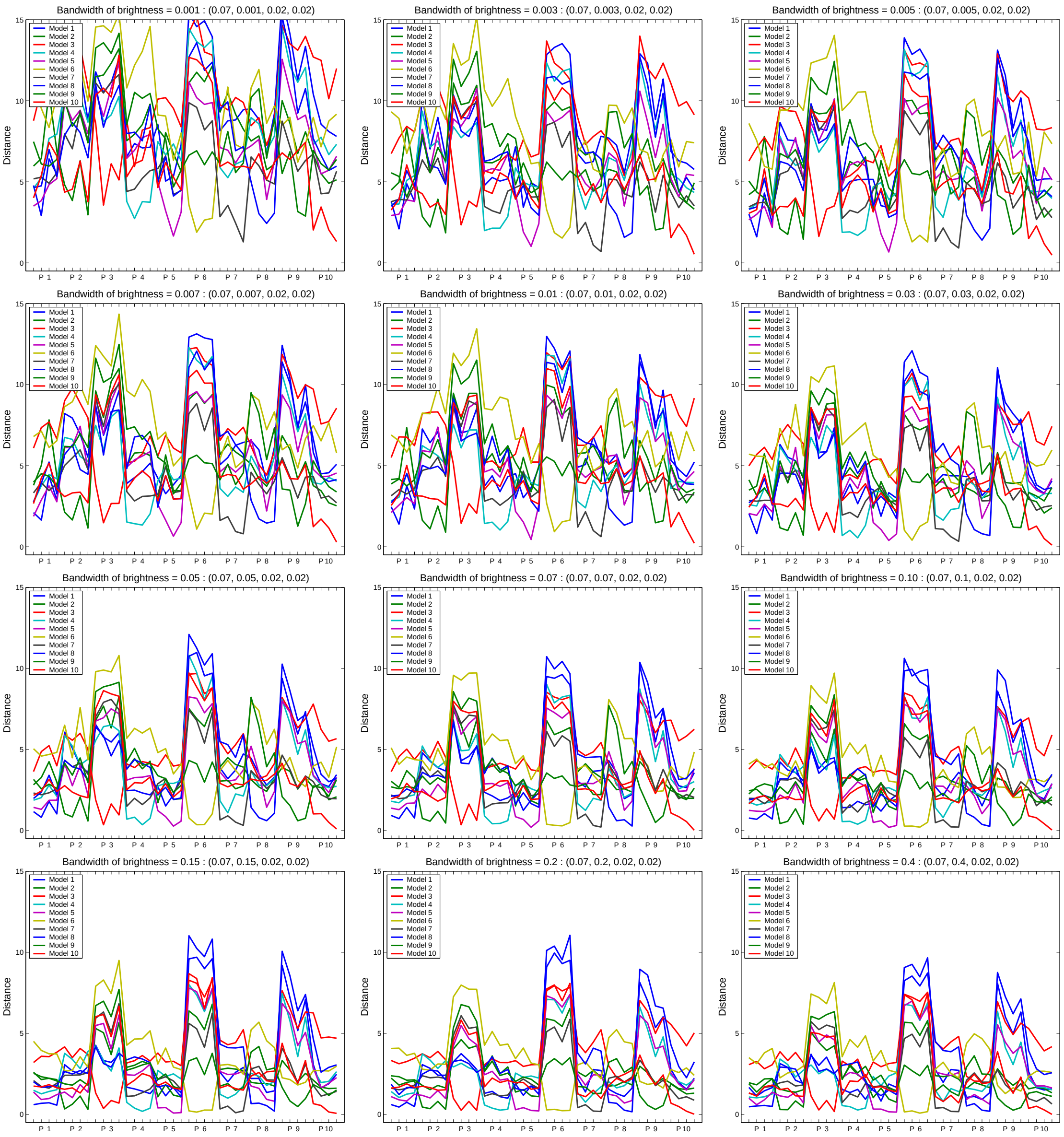
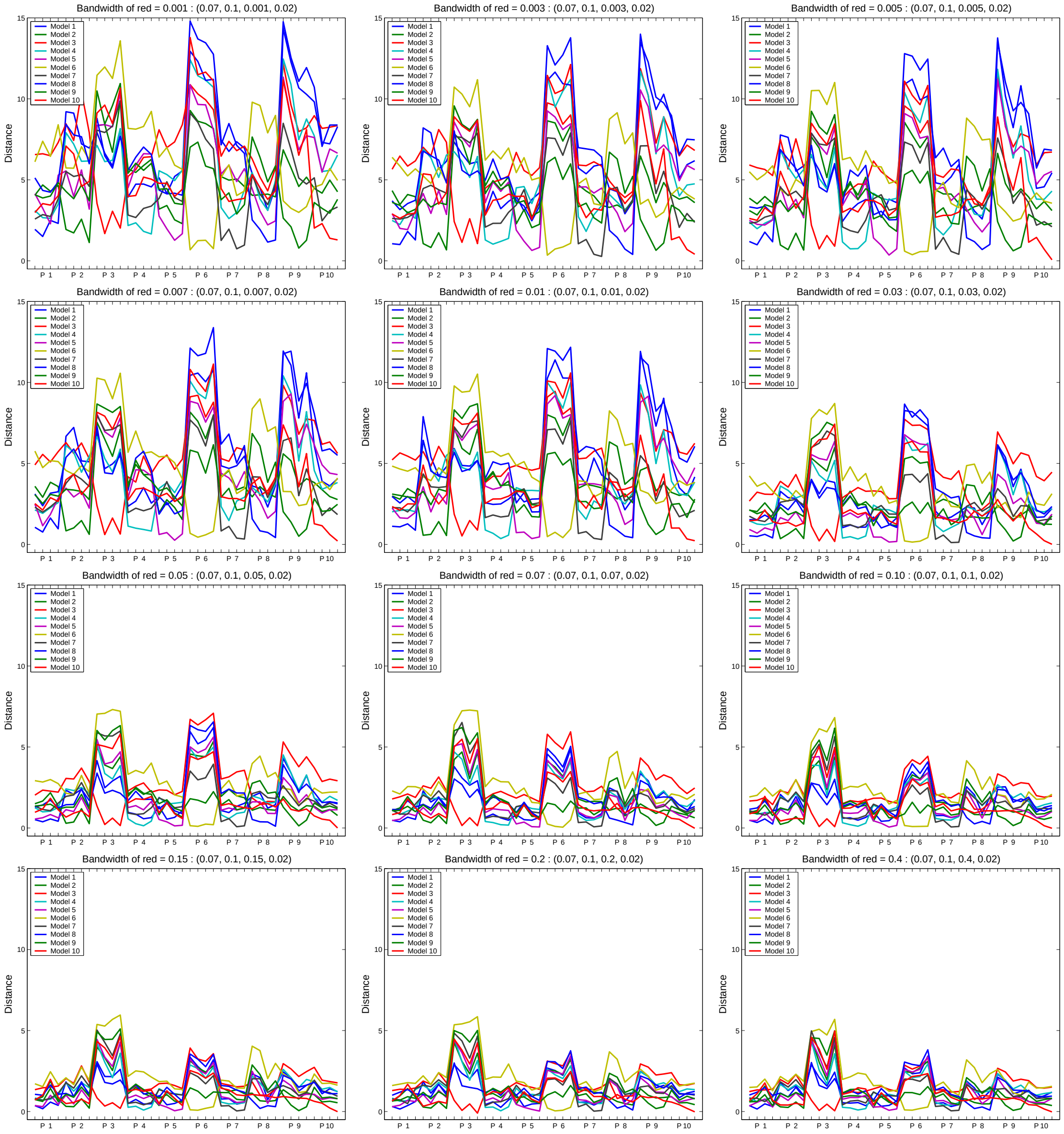


Figure 3.27: Varying bandwidth of *brightness* while fixing bandwidths of other variables. Acceptable results can be acquired at least bandwidth 0.07.



Bandwidth of red = 0.001 : (0.07, 0.1, 0.001, 0.02)

Bandwidth of red = 0.003 : (0.07, 0.1, 0.003, 0.02)

Bandwidth of red = 0.005 : (0.07, 0.1, 0.005, 0.02)

Bandwidth of red = 0.007 : (0.07, 0.1, 0.007, 0.02)

Bandwidth of red = 0.01 : (0.07, 0.1, 0.01, 0.02)

Bandwidth of red = 0.03 : (0.07, 0.1, 0.03, 0.02)

Bandwidth of red = 0.05 : (0.07, 0.1, 0.05, 0.02)

Bandwidth of red = 0.07 : (0.07, 0.1, 0.07, 0.02)

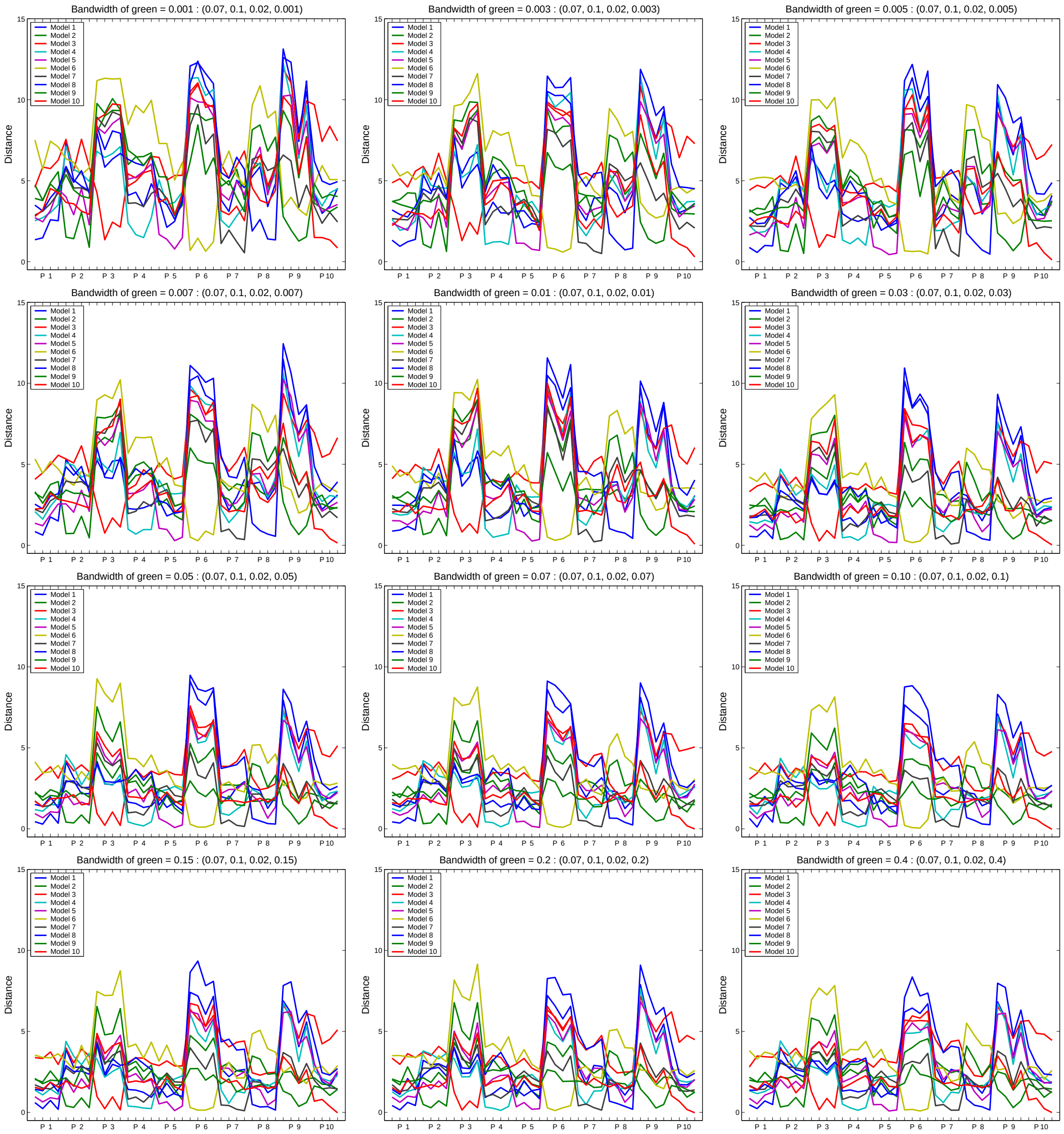
Bandwidth of red = 0.10 : (0.07, 0.1, 0.1, 0.02)

Bandwidth of red = 0.15 : (0.07, 0.1, 0.15, 0.02)

Bandwidth of red = 0.2 : (0.07, 0.1, 0.2, 0.02)

Bandwidth of red = 0.4 : (0.07, 0.1, 0.4, 0.02)

Figure 3.28: Varying bandwidth of *red* while fixing bandwidths of other variables. The best result can be obtained when 0.01 or 0.03 is used.



Bandwidth of green = 0.001 : (0.07, 0.1, 0.02, 0.001)

Bandwidth of green = 0.003 : (0.07, 0.1, 0.02, 0.003)

Bandwidth of green = 0.005 : (0.07, 0.1, 0.02, 0.005)

Bandwidth of green = 0.007 : (0.07, 0.1, 0.02, 0.007)

Bandwidth of green = 0.01 : (0.07, 0.1, 0.02, 0.01)

Bandwidth of green = 0.03 : (0.07, 0.1, 0.02, 0.03)

Bandwidth of green = 0.05 : (0.07, 0.1, 0.02, 0.05)

Bandwidth of green = 0.07 : (0.07, 0.1, 0.02, 0.07)

Bandwidth of green = 0.10 : (0.07, 0.1, 0.02, 0.1)

Bandwidth of green = 0.15 : (0.07, 0.1, 0.02, 0.15)

Bandwidth of green = 0.2 : (0.07, 0.1, 0.02, 0.2)

Bandwidth of green = 0.4 : (0.07, 0.1, 0.02, 0.4)

Figure 3.29: Varying bandwidth of *green* while fixing bandwidths of other variables. When the bandwidth is greater than 0.01, we can have reasonable results.

Table 3.3: Results for different number of sample points. The test data set is D_{16} with 608 frames.

Number of Frames	Number of Sample Points	Incorrect Matching	Matching Rate
608	All Points	2	99.7%
	200	6	99.0%
	100	17	97.2%
	50	37	93.9%
	20	80	86.8%
	10	109	82.1%
	5	135	77.8%

3.4.4 Sampling Method

Because using all the points of the test appearance is computationally prohibitive, we sample points and use them to compute the distance between the given test appearance and model. How the result can vary depending on the number of sample points and sampling method is shown in this experiment.

For the experiment in this section, we use data set, D_{16}. First of all, we changed the number of sample points to see how it affects the results. The tested number of samples are 5, 10, 20, 100, 200, and all points. The result of the experiment is described in Table 3.3.

From Table 3.3, we can see how the number of sample points is related to the accuracy of matching. As long as we have 50 sample points, the matching rate is above 90%. Even when we use all the points in the test appearance, there are 2 mismatched frames. Those two frames with matched models are shown in Figure 3.30. We can notice that although the matched model is not the same person, their appearances look very similar.

The next experiment investigates another aspect of the sampling method. In

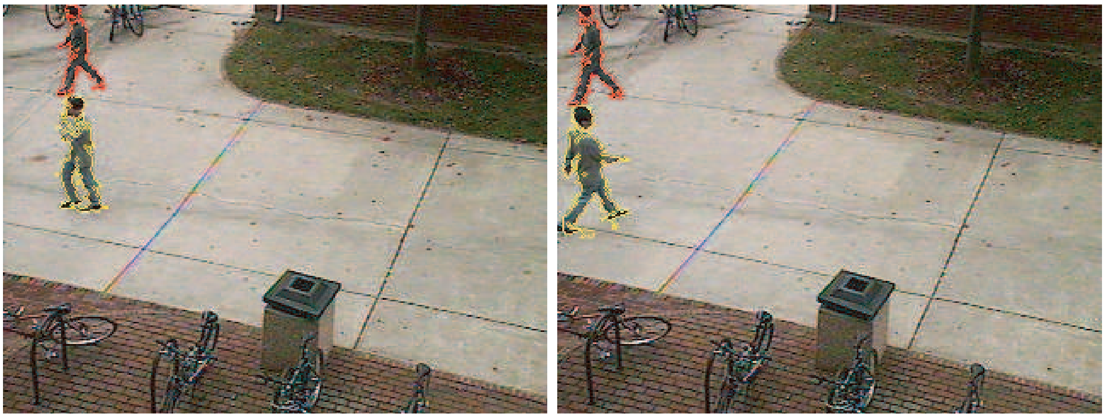


Figure 3.30: Two mismatched frames even when all the points are used to compute the distance.

the previous experiment, sample points are obtained evenly with respect to *path-length*. In other words, when we sample 10 points, we randomly select one sample point for points in each 10% *path-length* interval. This method guarantees that points are evenly sampled along the *path-length*. However, notice that the number of real points in each *path-length* interval is not the same. Hence, we tried two more sampling method as following.

1. Full random sampling: Points are randomly selected all over the area.

2. Stratified random sampling: This is intermediate method between "full random" and "along *path-length*". For each *path-length* interval, points are randomly selected. However it is chosen with the probability proportional to the number of points in the interval. For example, when we need 2 sample points, one point is sampled from the upper part (half of points from the shortest *path-length*) and the other from the lower part (rest of points). Depending on the shape of the appearance, *path-length* dividing the appearance is not

Table 3.4: Results for sampling method. Three sampling methods are compared. PL: Along *path-length*, FR: Full random, ST: Stratified random. The test data set is D_{16} with 608 frames. Number of incorrect matching.

Number of Samples	PL	FR	ST
100	17 (97.2%)	17 (97.2%)	10 (98.4%)
50	37 (93.9%)	33 (94.6%)	23 (96.2%)
20	80 (86.8%)	62 (89.8%)	52 (91.4%)

necessarily 0.5.

Table 3.4 show the results with different sampling method. We have tested all three sampling method with several number of sample points, 20, 50, and 100. In a nutshell, stratified random sampling performs the best. Only with 20 sample points, the matching rate is 91.4%. This is about 5% better than original sampling method.

3.4.5 Comparison to Histogram Method

In this section, we perform comparisons between our approach and another method. The most popular method to compare two distributions is a histogram method. After two separate histograms are built from the distributions,the Bhattacharyya distance can be used to compare the histograms [12, 11].

We used data set D_2 and D_4 for this experiment. The result of the first experiment with D_2 is shown in Table 3.5. For the histogram method, we tested both 3-D and 4-D features. For 3-D histograms, (RED, GREEN, BLUE) was used as the feature vector. In 4-D histogram test, (*path-length*, brightness, red, green) was used. As seen in the table, the 3-D histogram is outperformed by the 4-D histogram and KL distance method. However, there is no significant difference between our

Table 3.5: Comparison between KL and histogram method using data set D_2. 4D histogram method shows better result than our method.

Number of Appearances	Method	Incorrect Matching	Matching Rate
1223	3D Histogram	60	95.1%
	4D Histogram	15	98.8%
	KL distance (100 sample points)	21	98.3%

Table 3.6: Comparison between KL and histogram method using 1/4 size reduced data set D_2. KL distance shows the best result though it uses only 30 sample points.

Number of Appearances	Method	Incorrect Matching	Matching Rate
1223	3D Histogram	158	87.1%
	4D Histogram	43	96.5%
	KL distance (30 sample points)	42	96.6%

method and 4-D histogram approach.

One of reasons would be the sampling. If we use all the points, the performance of KL method will be at least as good as 4-D histogram case. Another more important reason is that when there are many data points in the model and test appearance, there is no benefit to use kernel density estimation. Hence, we performed another experiment on much more difficult data sets. We prepared test data by reducing the size of images in D_2. The result with the new data set (resized D_2) is shown Table 3.6. We resized all the images to a quarter of their original size (the dimension of image in D_2 is 352×240, and the reduced size is 88×60). In this test, though we used only 30 sample points, the performance of our method and 4-D histogram are almost the same.

Another experiment with data set D_4 is shown in Table 3.7 and Table 3.8. We used all points from the resized data set in this experiment. As we can see in Table 3.8, our method clearly outperforms the histogram method.

Table 3.7: Comparison between KL and histogram method using data set D_4.

Number of Appearances	Method	Incorrect Matching	Matching Rate
1359	3D Histogram	61	95.5%
	4D Histogram	47	96.5%
	KL distance (100 sample points)	38	97.2%

Table 3.8: Comparison between KL and histogram method using 1/4 size reduced data set D_4. By using all points in the appearance, we can acquire much better result with KL distance method.

Number of Appearances	Method	Incorrect Matching	Matching Rate
1359	3D Histogram	92	93.2%
	4D Histogram	76	94.4%
	KL distance (All points)	13	99.0%

The last experiment in this section is conducted in the compressed domain. We showed that better result can be acquired by our method based on KDE when we have models for only a small number of data points. This happens very often in the compressed domain. When a video is compressed, because of its quantization process, the distribution of data is very spiky and simple histogram comparison might be unstable. However, KDE has an advantage that it can generate a smooth and stable model distribution even with small number of data points. Hence, we compressed the video clip of data set D_4 with data a rate around 70KBps, and regenerated frame images. The same frames were used as models as in the original setup. The result is shown in Table 3.9 and some frames with result matching and model frames are in Figure 3.31.

Table 3.9: Comparison between KL and histogram method using data set D_4 from compressed video.

Number of Appearances	Method	Incorrect Matching	Matching Rate
1359	3D Histogram	141	89.6%
	4D Histogram	91	93.3%
	KL distance (100 sample points)	69	94.9%

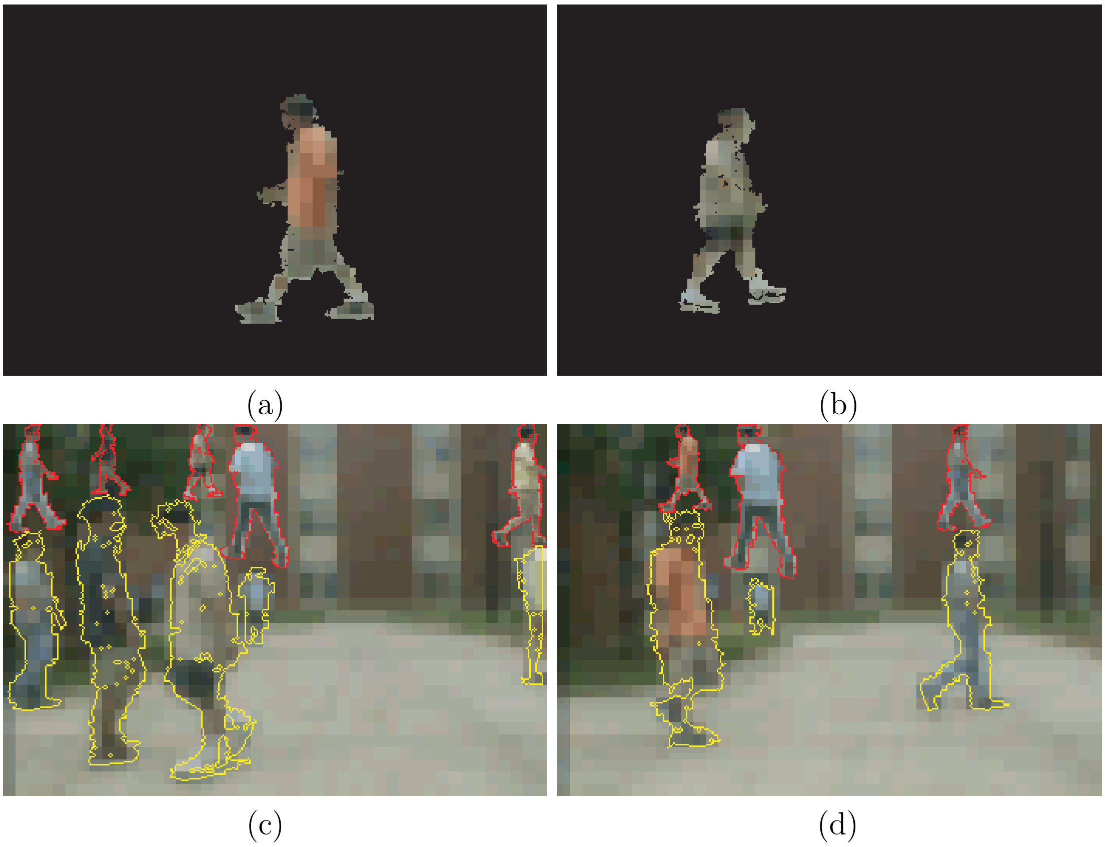


Figure 3.31: Sample images of compressed domain experiments. (a) and (b) are two sample model images. (c) and (d) are examples of matching results.

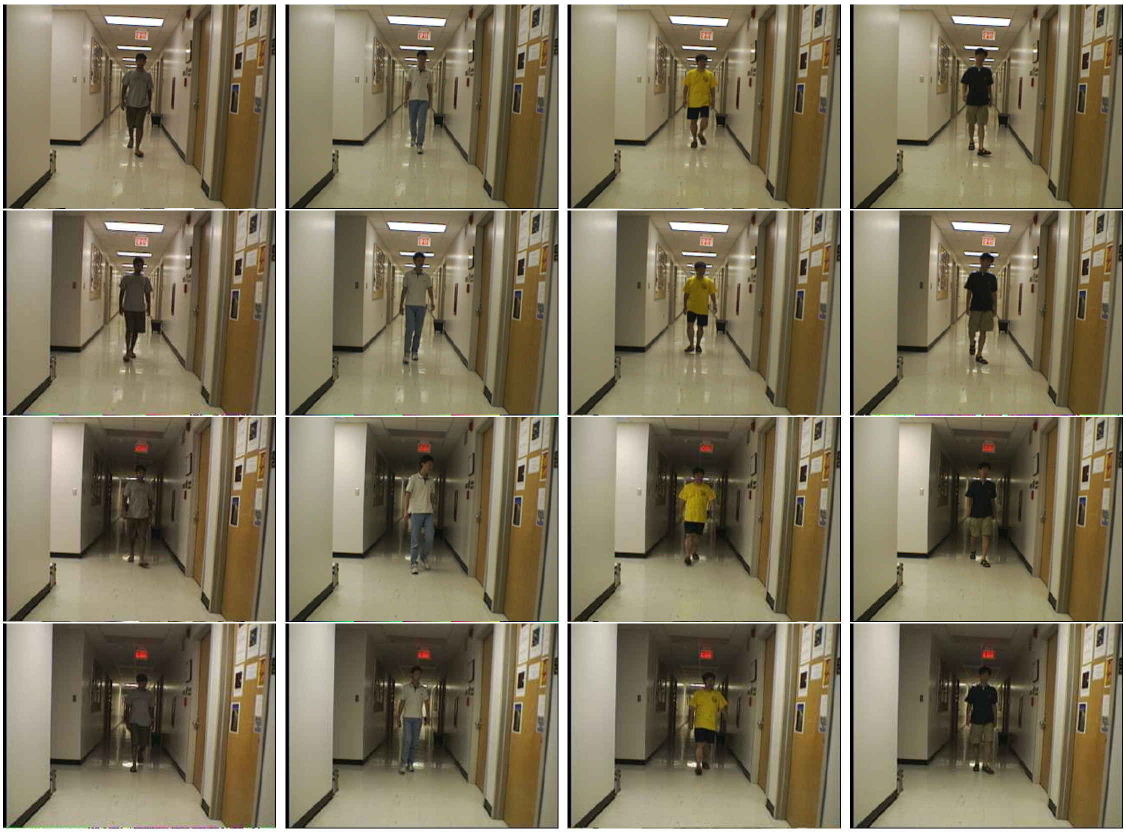


Figure 3.32: Four people under various illumination conditions. All the people have the same front views. Illumination condition is controlled by two sets of lights. Each person stands on the cross-section of two hallways and both hallways have lights on ceiling. Both lights are on in the first row images. One of them is on in the second and the third row, and both off in the last row.

3.4.6 Robustness Test Under Various Conditions

In this section, several test results are presented to show the robustness of the profiles under various environmental conditions. Some conditions we have tested in this section are illumination, viewpoints, multiple cameras, and so on. Also we present more experimental results under one-to-one matching constraint and different gallery management method.

Figure 3.32 shows the sample frames from data set D_1 which have various illumination conditions. Four different people were captured with four different

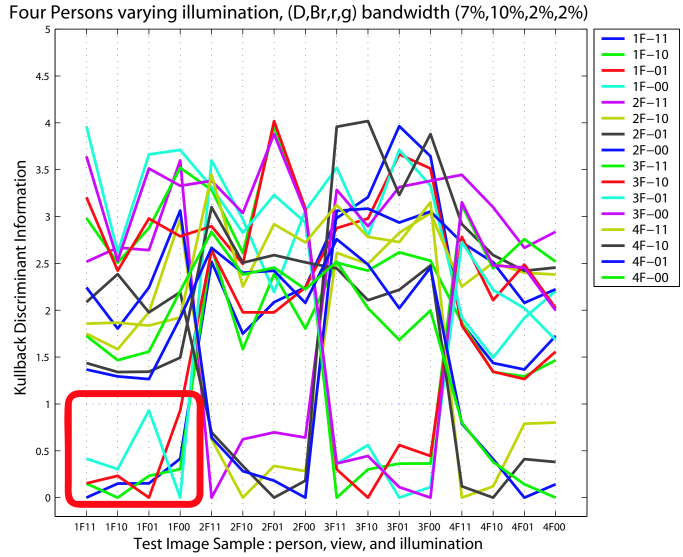


Figure 3.33: Result from data of Figure 3.32. The x-axis labels are appearances. For instance, 2F11 means person 2, frontal image, and both lights on. Graphs were generated using variables (path-length, brightness, red, green).

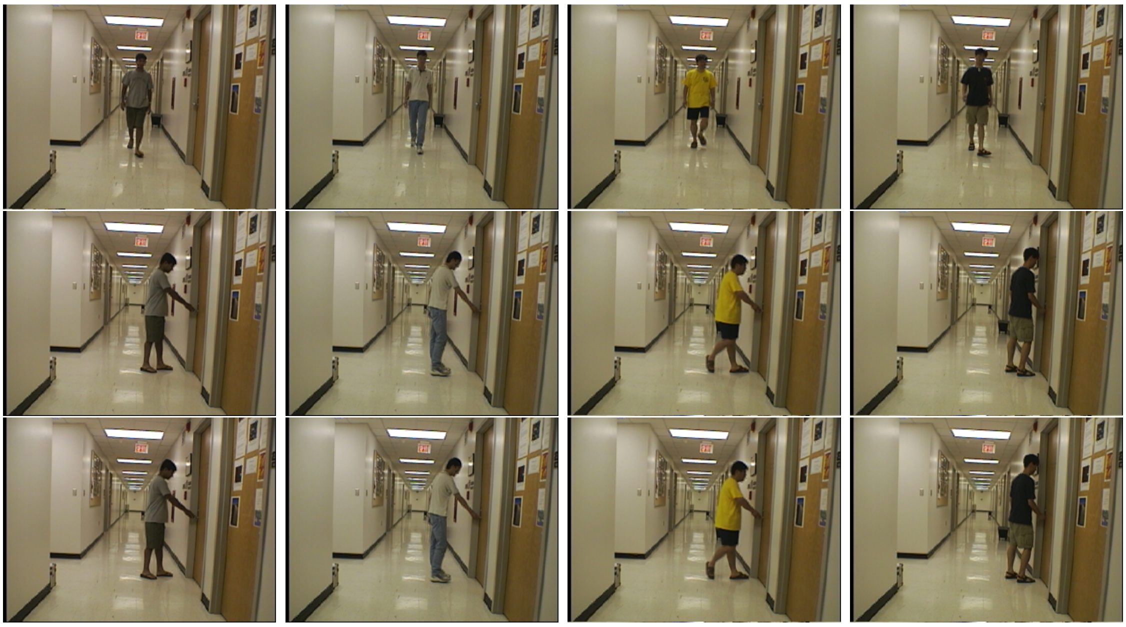
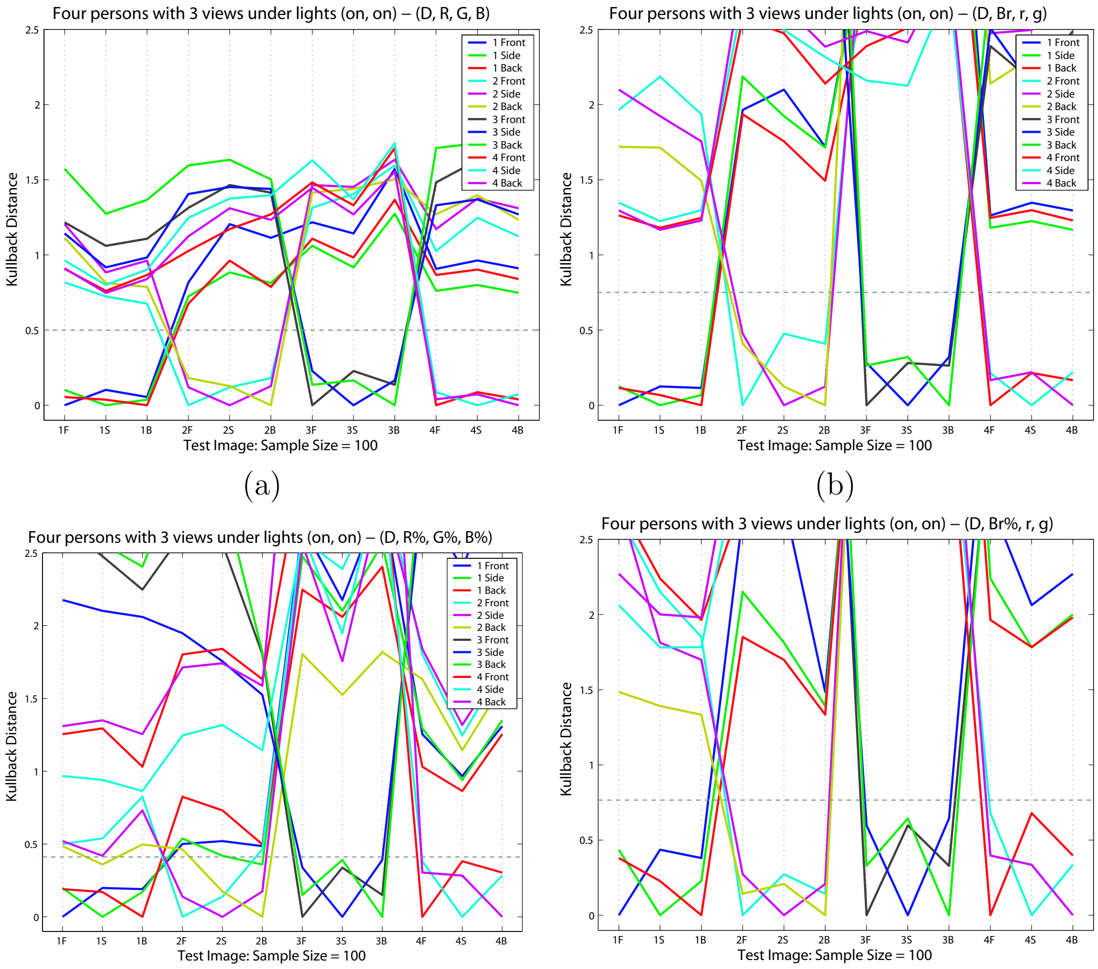


Figure 3.34: Four people with three different views. All the cases have the same illumination conditions.



(a)

(b)

(c)

(d)

Figure 3.35: Four result graphs for four people with different views of Figure 3.34 Four variable sets of section 3.1 are used for (a), (b), (c), and (d) respectively.

illumination conditions. In this example only frontal view was used and Figure 3.33 represents a result from all 16 images in Figure 3.32. Since there are four people, appearances are categorized into four different groups. As seen in the graph, four consecutive images along the x-axis constitute a group and they have lower intra-group distances than inter-group distances.

The first group (person 1) is marked with a rectangle in Figure 3.33. The dotted line in the graph is a tentative threshold value to separate groups. As discussed in section 3.2.2, 7% for path-length, 10% for (rank of) brightness, and 2% for color proportion are used as bandwidths.

The next experiment shows the robustness with respect to view points. Figure 3.34 has 12 frames which have 4 people with 3 different views. The illumination condition is fixed, and each person has three views, frontal view, side view, and back view. When the view point changes, one possible problem is caused by the head part. While frontal view and side view have some skin part, the back view has only hair color. However, the head part is relatively small. The result graphs in Figure 3.35 show that there are no difficulties in matching people. All four variable sets described in section 3.4.2 are used, and they generate similar results except for *(path-length, rank of RED, rank of GREEN, rank of BLUE)* case.

Figure 3.36 has appearance images with varying views and illuminations simultaneously. Two people were captured and each person has three different views under four different illumination conditions. In total there are 24 images of the two people. All four variable sets are applied. The result graphs are shown in Figure 3.37.

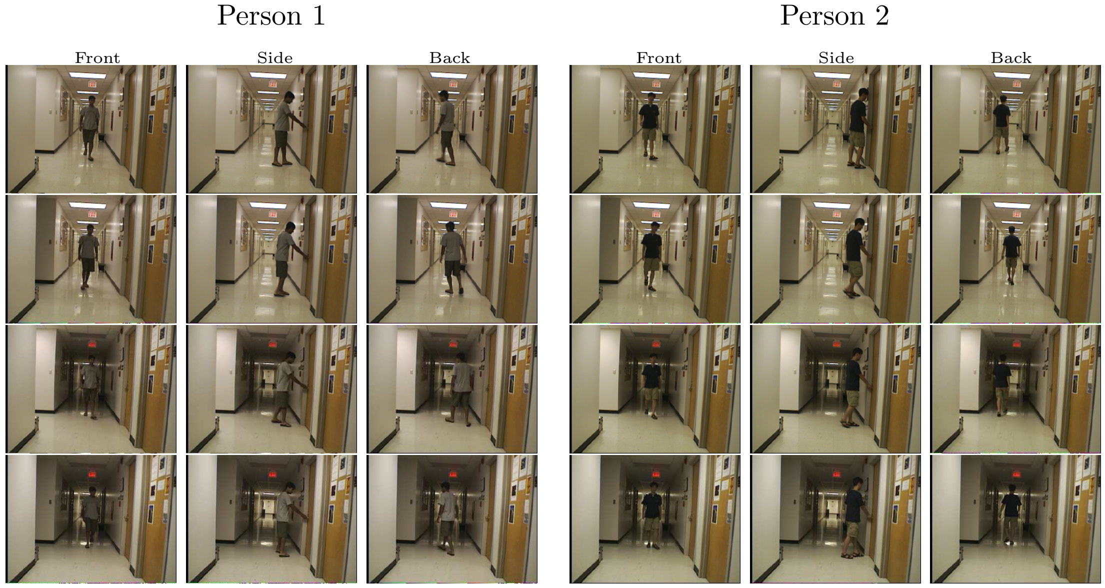
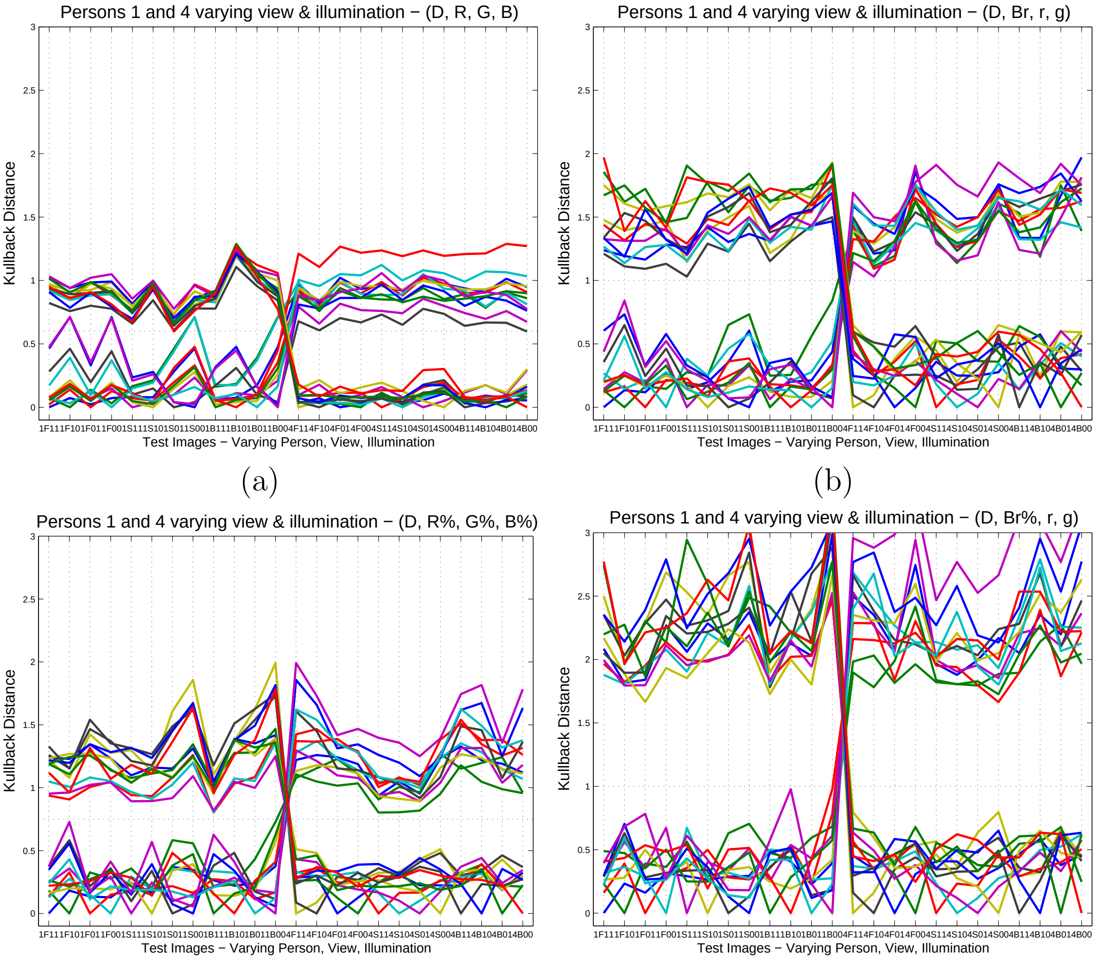


Figure 3.36: Two people with three different views under four varying illuminations.

The next experiment is performed to evaluate the robustness with respect to multiple cameras. Figure 3.38 shows all five different people of data set D_{12} and D_{13} with two different views (front and side) using two different cameras at the same time. Without calibration, the color of different camera has different characteristics. In this case, using the rank order is better than using the original color values. The result graphs are shown in Figure 3.39, and the best result in this case is *(path-length, rank of RED, rank of GREEN, rank of BLUE)*.

A similar experiment is done with D_{12}, D_{13}, D_{14}, and D_{15} more comprehensively. 20 models from the data set are shown in Figure 3.40. For each data set, five people were captured separately with two cameras, and 20 video clips were generated in total. Each person moves from left to right in the scene, and they stop at the middle and turn around to show all the views.

We chose 4280 frames from D_{12}˜D_{15}, where their foreground regions are not



(a) (b)

(c) (d)

Figure 3.37: Four result graphs for two people with different views and illuminations of Figure 3.36. Four variable sets in section 3.1 are used for (a), (b), (c), and (d) respectively.

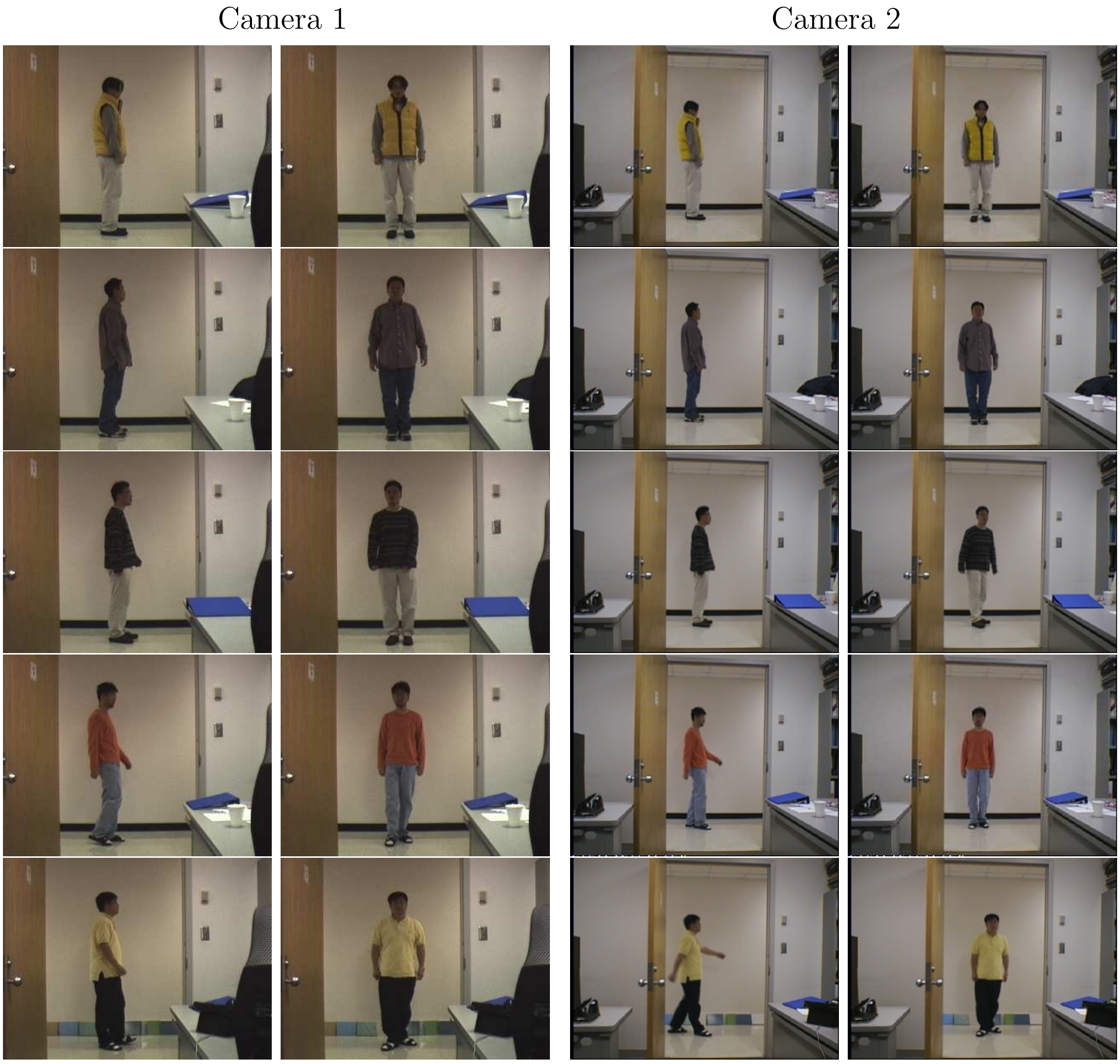


Figure 3.38: Multiple cameras multiple view case data from D_{12} and D_{13}. Five people were captured using two different cameras with two different views. To avoid too much complexity, only two views (front and right side) are used here.

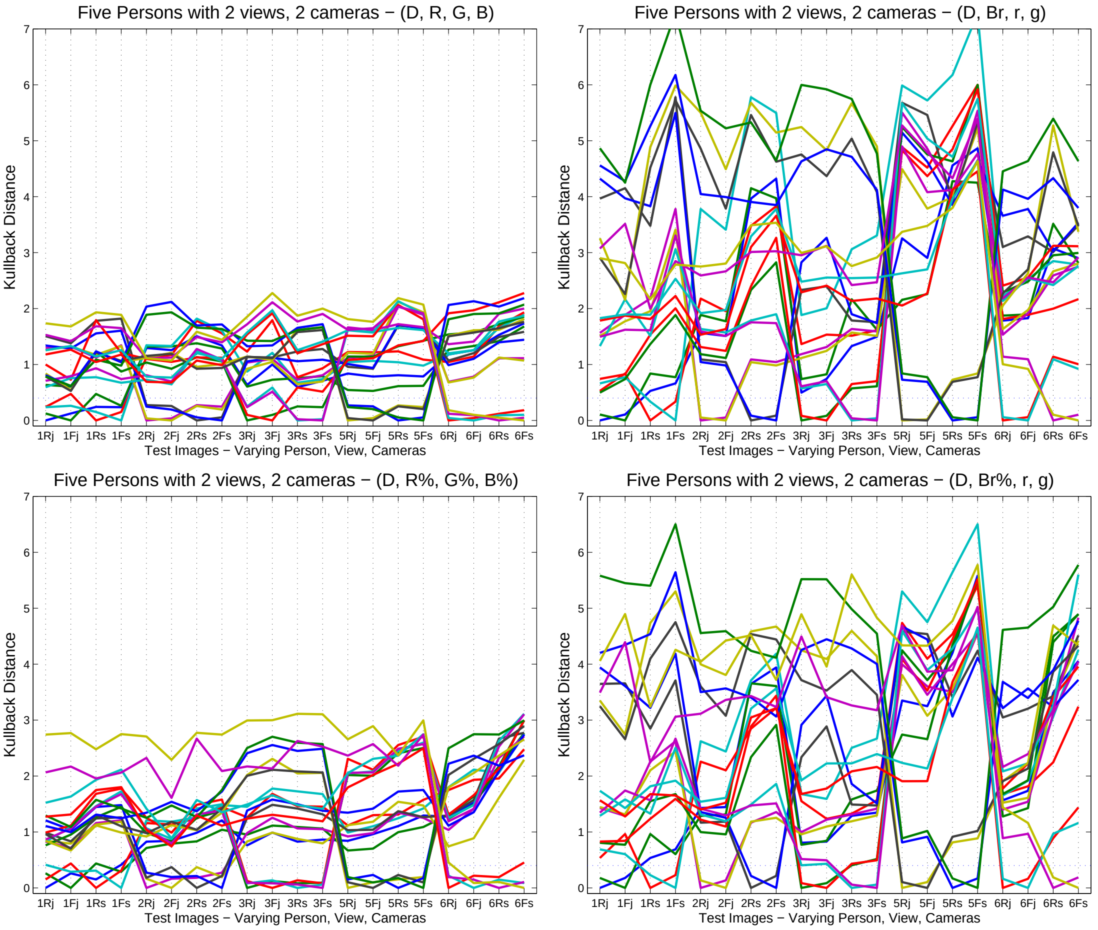


Figure 3.39: The result of multiple cameras multiple view case data (Figure 3.38).

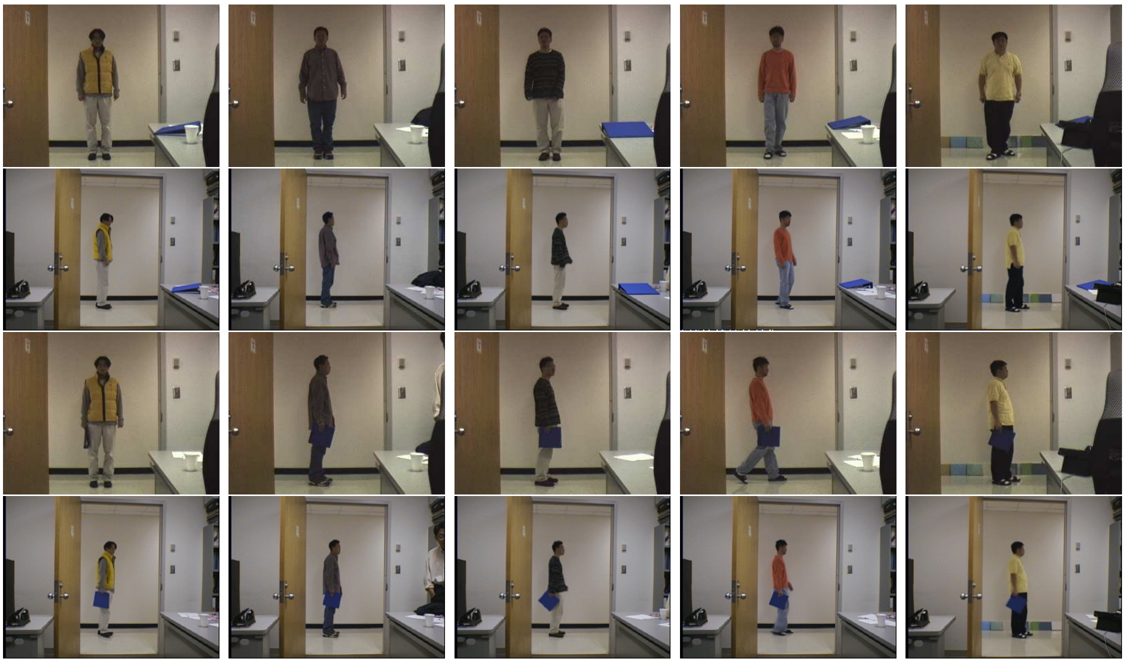


Figure 3.40: 20 frames used for models of D_{12}, D_{13}, D_{14}, and D_{15}. In the first test with static gallery and test with dynamic gallery, only 5 models in the first row were used. In the second test with static gallery all 20 appearances are used as models.

Table 3.10: Experimental results of gallery management and multiple cameras with 4280 frames of D_{12}, D_{13}, D_{14}, and D_{15}. (Figure 3.40)

Gallery	Matched	Mismatched	New	Rates
Static (5 models)	3766	514		87.9
(20 models)	4229	51		98.8
Dynamic	4245	35	15	99.1

clipped by boundaries. The number of appearances is also 4280 because only one person is visible in any frame. Table 3.10 shows the test results. Two tests were performed: the first with a static gallery, and the other with a dynamic gallery. In the first test, only 5 models were kept in the gallery, while all 20 models from 20 clips were used in the second test. The matching rate is much higher in the second test. In the test of dynamic gallery, 15 new models were built while processing frames, and more than 99% of appearances have been matched correctly.

The data set D_7, D_8, D_9, and D_{10} are used for multiple appearances matching.

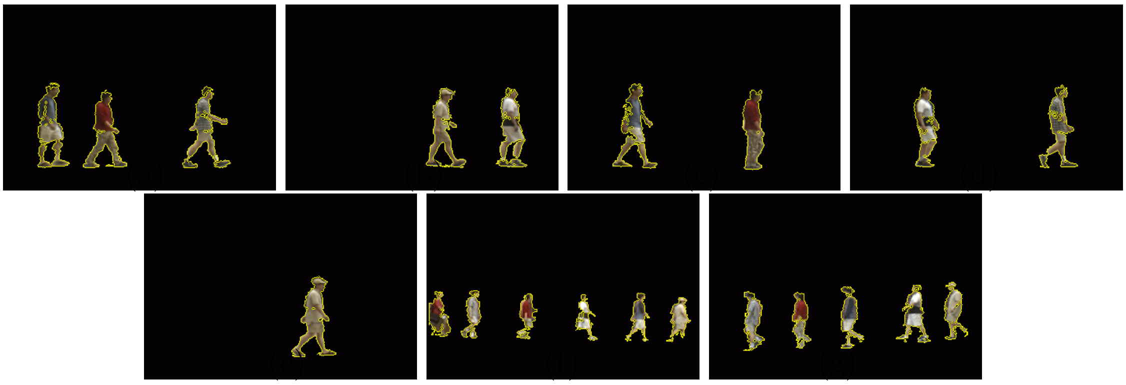


Figure 3.41: Four sets of model appearances. (a), (b) are models of D_7, (c), (d), and (e) are models of D_8. (f) is for D_9, and (g) is for D_{10}.

Five or six people were captured outdoor, and they passed twice through the scene. Also, they can appear in one frame at the same time, and it is necessary to run multiple appearances matching algorithm. Figure 3.41 shows initial models for the four test sets.

Table 3.11 shows the result of the experiment. For each set, four tests were performed. Label 1-1 in the table means that the test uses multiple appearances matching with one-to-one constraint. The number of appearances for each test set is in parenthesis in the first column. The results of data set D_9 show that enforcing one-to-one constraint has better performance because the set has six appearances in one frame at maximum and the foreground regions are relatively small and noisy as seen in Figure 3.41. A collection of randomly chosen result images of tests with D_7-D_{10} and D_{12}-D_{15} are shown in Figure 3.42. In each result image, the matched model appearances are drawn over the top of the person.

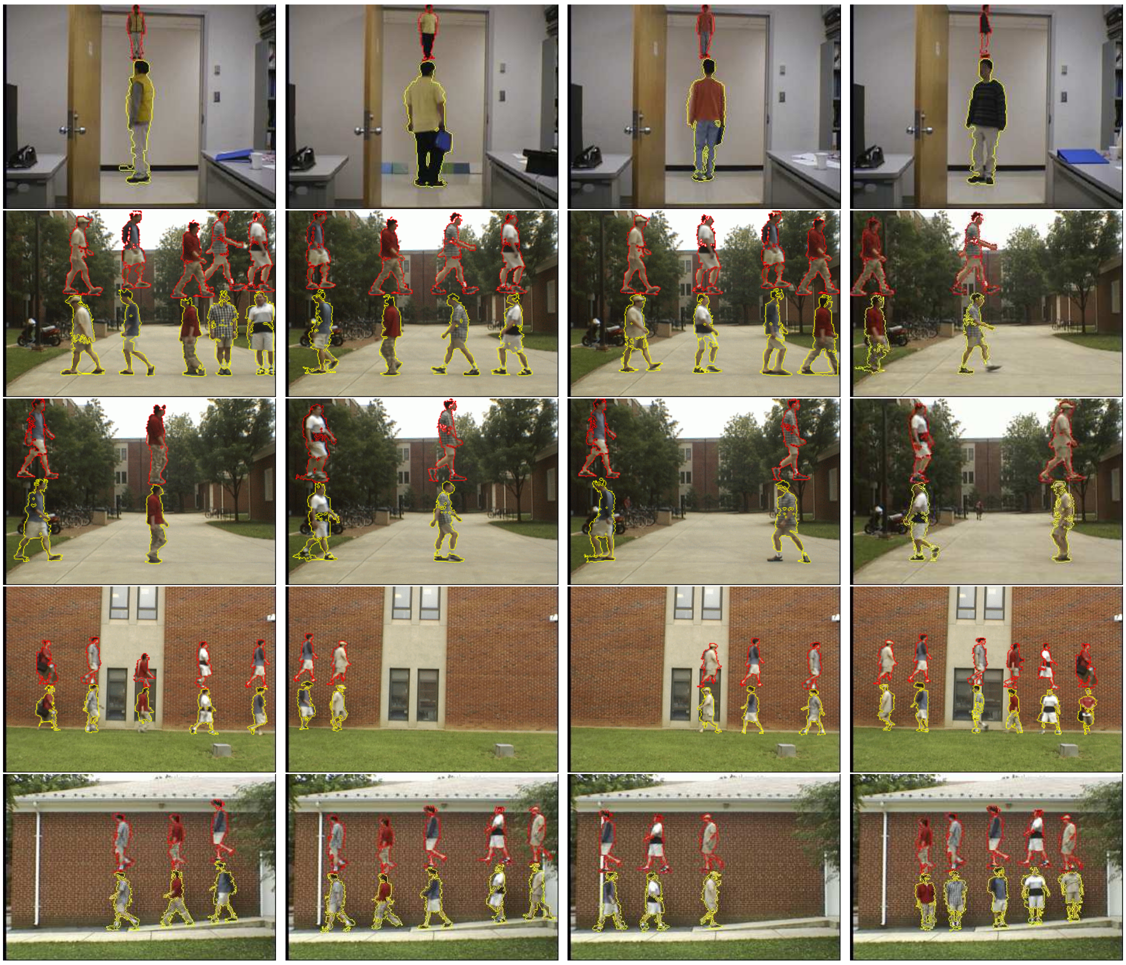


Figure 3.42: Example result images of tests in this section.

Table 3.11: Experimental results four test sets (Figure 3.41)

Test Set	Gallery	1-1	Matched	Mismatched	New	Rates
D_7	Static	No	875	24		97.3
(899)		Yes	874	25		97.2
	Dynamic	No	877	22	4	97.5
		Yes	884	15	4	98.3
D_8	Static	No	743	29		96.2
(772)		Yes	748	24		96.9
	Dynamic	No	756	16	2	97.9
		Yes	756	16	3	97.9
D_9	Static	No	1809	139		92.9
(1948)		Yes	1887	61		96.9
	Dynamic	No	1907	41	84	97.9
		Yes	1916	32	67	98.4
D_{10}	Static	No	1406	16		98.9
(1422)		Yes	1404	18		98.7
	Dynamic	No	1404	18	1	98.7
		Yes	1402	20	0	98.7

3.4.7 Local Change Detection

In this section, the usefulness of local change detection based on appearance is shown with examples. Once an instance is matched to a model, we can localize the difference as described in section 3.3.

The local change is likely caused by a carried package in the case of delivery or pickup event. In Figure 3.43, there are four columns, where the first and second column are two frames for the same person, one without, the other with a package. The snapshot in the first column is used as a model and the second snapshot is used as a test image. Pixel-based probability images are on the third column, and on the last column improved probability images using segmentation are shown. In all the cases, segmentation gives one connected detection area of local changes with clear boundaries.

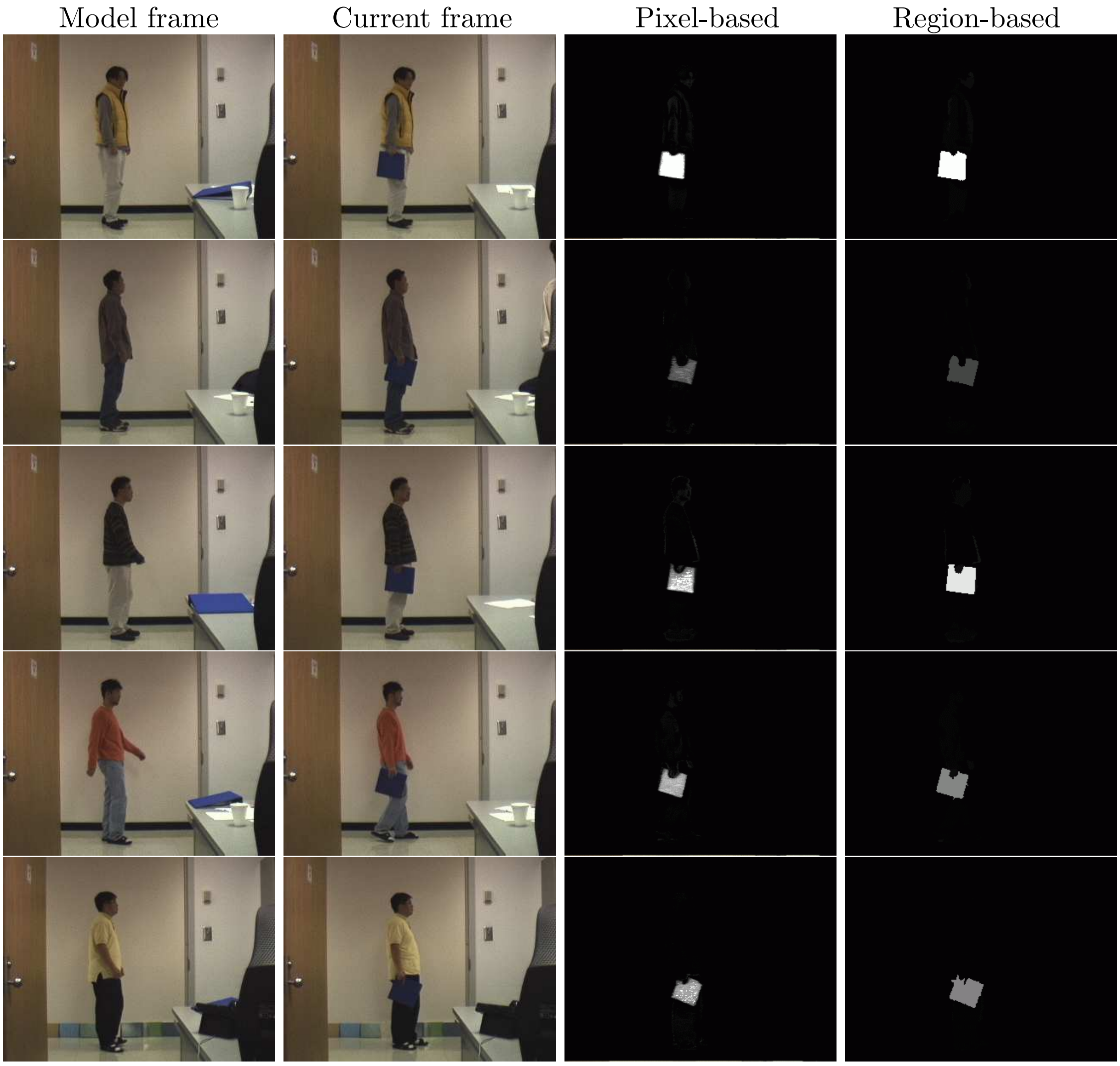


Figure 3.43: Frame images and results for local change detection. In the first and second column are two appearances. The first column is used as a model, and the second column as a test image. In model images people do not have anything, while they carry a package in test images. Two types of result probability images are in the third and fourth columns. The third column has pixel-based probability images, and the segmentation-based results are in the fourth column.

3.5 Conclusion

We have presented a novel way to recognize people by introducing *path-length* as a new feature for spatial information based on profiles of appearance. The KDE and KL distance were used to calculate distance between appearances. Many possible combinations of variables were tested and four variable sets of four dimension turned out the best. Using the variable sets, we can successfully match people in the case of various illuminations, viewpoints, and multiple cameras.

Also, local changes can be detected. We showed that the segmentation of the image can improve the result so that detected areas have one connected blob and clear boundaries. When there is more than one person in a frame, we used multiple appearance matching with a one-to-one constraint, and obtain better performance.

Chapter 4

Gallery Construction using Appearance Model

In many areas of computer vision, databases for reference models are frequently used. Those databases include faces [22, 41, 49, 52], pedestrians [53], vehicles [1, 16, 37], and general objects [46, 55] etc. There have been few studies about how to build a reference database automatically. In this chapter, an approach for constructing a gallery of people observed in a video stream is described. We consider two scenarios that require determining the number and identity of participants: outdoor surveillance and meeting rooms. In these applications face identification is typically not feasible due to the low resolution across the face. The proposed approach automatically computes an appearance model based on the clothing of people and employs this model in constructing and matching the gallery of participants. The appearance model uses *color/path-length* profile and a robust distance measure based on Kernel Density Estimation (KDE) and Kullback-Leibler (KL) distance, to evaluate similarity between people and add models to the gallery as described in Chapter 3. A one-to-one constraint is enforced to correctly match instances to models at each frame. In the meeting room scenario we exploit the fact that the relative locations of subjects are likely to remain unchanged for the whole sequence.

4.1 Introduction

One aspect of video surveillance of indoor meetings involves matching a person against a gallery of known people. Such a gallery is tedious to construct manually. We describe an approach to automatically construct a gallery of participants based on clothing-appearance. The gallery directly supports the human identification task but it can also be used to answer questions such as how many people were observed, when each has appeared and how people interacted in video sequences.

We assume that people do not change clothing, although our method does tolerate localized appearance changes such as a person holding a package at one time, but not at another. We employ well-known approaches for human detection in video and focus on the modelling and matching of human appearance.

We consider two application areas: surveillance and meetings video. Here, it is difficult to employ faces for identification since the resolution across the face is too small and faces typically appear in off-frontal poses or profile views. Instead, we model the clothing of people and acquire quantitative models that support matching.

We represent appearances of people from silhouettes obtained by background subtraction or from torso areas computed based on detected faces. Over short periods of time, we assume that the appearance of the person remains unchanged, except for small, local changes, for instance due to carried packages or illumination variation.

We model human appearance based on clothing-color and spatial information as in Chapter 3.

4.2 Robust Distance Measure

The foreground region representing a person is used to construct an appearance model that is compared to models in the gallery. The distance between the current appearance and existing appearance models in the gallery determines if a new model should be added to the gallery or the current appearance is an instance of a model already present in the gallery.

Human appearance in video streams varies over time. In outdoor scenes, lighting, human pose variation and carried objects may lead to changes in the foreground region. Similarly, in the meeting scenario people move their arms in front of their torso as well as handle objects that may occlude parts of their torso. To cope with such variations we employ a robust estimation norm that adjusts the weighting of points within the distance metric based on whether points are inliers or outliers.

For the robust estimation, we employ the general M-estimator, which minimizes the objective function [32]

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - \mathbf{x_i}^T \mathbf{b}) \tag{4.1}$$

where $\mathbf{x_i}$'s are independent variables, y_i's are data points, $\mathbf{b}$ is a coefficient vector, ρ is the influence function, and n is the number of data points.

Let $\psi = \rho'$ be the derivative of ρ. We need to solve the following equation to minimize (4.1).

$$\sum_{i=1}^{n} \psi(y_i - \mathbf{x_i}^T \mathbf{b}) \mathbf{x_i^T} = 0$$

If we define the weight function $\omega(e) = \psi(e)/e$, and let $\omega_i = \omega(e_i)$. Then the equation can be written as

$$\sum_{i=1}^{n} \omega_i (y_i - \mathbf{x_i^T b}) \mathbf{x_i^T} = 0 \tag{4.2}$$

For each sample point, we define a new feature, δ_i using p_i and q_i (defined in (3.9)):

$$\delta_i = \frac{|q_i - p_i|}{max\,(p_i, q_i)}$$

When the current instance is correctly matched to a model, most p_i's are similar to q_i's leading the δ_i's to be close to 0. On the other hand, when the instance and model are mismatched, most δ_i's will be greater than 0. The mean of δ_i will roughly represent how much the current instance is matched to the model correctly. We apply the robust fitting (4.2) to compute the robust mean of the δ_i's, μ; it can be written as

$$\sum_{i=1}^{n} \omega_i (\delta_i - \mu) = 0$$

Notice that weights are designed to minimize the influence of outliers. In other words, the weight of each data point depends on how far the point is from the mean. Data points near to the estimated mean get high weight. Points that are far from the mean have smaller weights. When points are farther from the mean than would be expected by random chance, they get zero weight.

The weights depend upon the residuals and the residuals depend upon the estimated mean, and the estimated mean depend upon the weights. The iteratively re-weighted least square (IRLS) method is employed to get a robust mean [9, 20].

The bisquare weight function used in our approach is defined as in [9, 20]:

$$\omega_B(u) = \begin{cases} \left[1 - (\frac{u}{B})^2\right]^2 & |u| \leq B \\ 0 & |u| > B \end{cases}$$

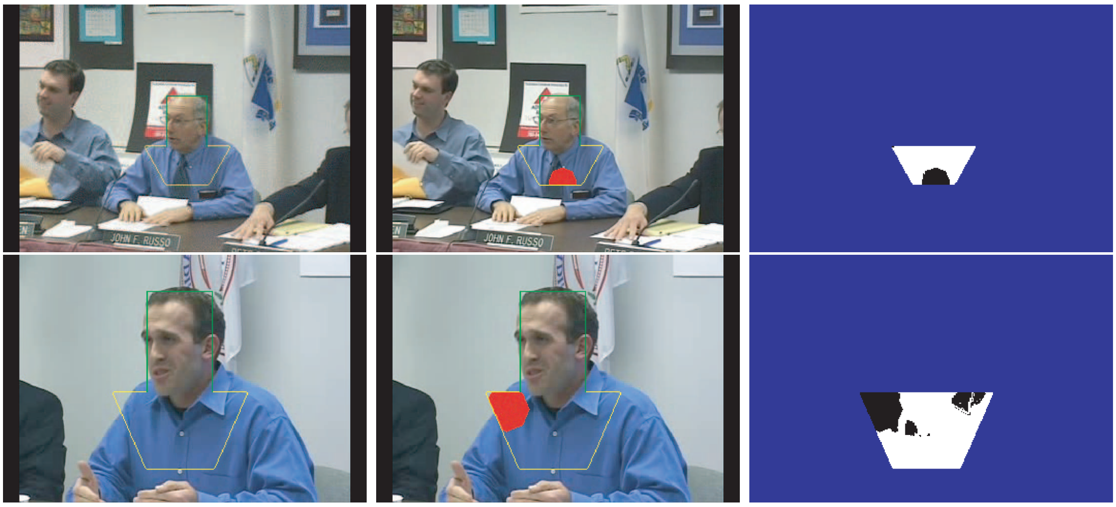


Figure 4.1: Detection of outliers. The image in the first column is the model image. Second column images are used as instances. To synthesize outliers, a 15% size block with red color pixels is created. In the third column the inliers and outliers are shown as white and black points, respectively.

where the default tuning cost $B = 4.685$, and u is the scaled residual.

We use the final weights at the last iteration after the estimated mean converges as the degree of inliers. Only data points with the weight greater than a certain threshold value, e.g. 0.1, are regarded as inliers. The KL distance is recomputed only using inliers. Figure 4.1 shows examples of outliers and inliers as determined using robust fitting method for a sample region that has been manually altered by changing its color. About 15% of the points in the region have been manually changed but the match between the model region and the instance remained strong.

4.3 Multiple Matching and Dynamic Gallery

The uniqueness constraint (e.g. two people seen simultaneously must be different) is important in the gallery construction process. The gallery is built starting from an empty set while frames are processed. Whenever the system processes a frame, it tries to match all the instances to models in the current gallery. When it fails to find a matched model for an instance, a new model is added to the gallery. This matter is described in section 3.2.4.

4.4 Spatial Analysis

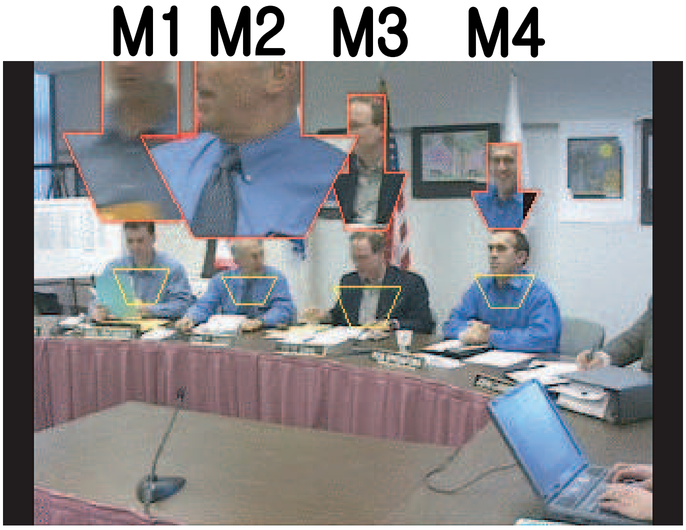
In meeting room videos, the spatial order of people's positions is likely to remain stable over the whole sequence because participants are mostly seated and do not walk around. It is possible to improve the accuracy of the models in the gallery and the identification performance by utilizing the relative order of participants. We do this as follows.

For each model, M_i, we compute an adjacency matrix, F_i that captures the frequency of spatial ordering among models. An adjacency matrix, F_i is $m \times n$, where n is the number of models and m indexes relative positions. For example, if N is the maximum number of people in one frame and people are arranged in a "linear" configuration, then $m = 2 * (N - 1)$.

To build the adjacency matrix F_i, all the frames which have a person matched to model M_i are employed. The (j, k)-th element of F_i is the number of occurrences of model M_k at the relative horizontal position, $pos(j)$. $pos(.)$ is defined as

$$pos(j) = \begin{cases} j - \frac{m}{2} - 1 & \text{if } j < \frac{m}{2} \\ j - \frac{m}{2} & \text{otherwise} \end{cases} \tag{4.3}$$

The upper half of an adjacency matrix, F_i, represents the frequencies of models



(a)

$$F_1 = \begin{bmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{bmatrix} \quad F_2 = \begin{bmatrix} 0&0&0&0\\ 0&0&0&0\\ 1&0&0&0\\ 0&0&1&0\\ 0&0&0&1\\ 0&0&0&0 \end{bmatrix} \quad F_3 = \begin{bmatrix} 0&0&0&0\\ 1&0&0&0\\ 0&1&0&0\\ 0&0&0&1\\ 0&0&0&0\\ 0&0&0&0 \end{bmatrix} \quad F_4 = \begin{bmatrix} 1&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0 \end{bmatrix}$$

(b)

Figure 4.2: Simple example of constructing adjacency matrices. If there is only one frame with four people like in (a), we can have four adjacency matrices in (b). In this example, the number of models, n, is 4, and m is 6 since N is 4.

to the "left" of M_i; the bottom the "right" side. Figure 4.2 shows an example of adjacency matrices from a single frame.

The difference between adjacency matrices represents how similar two models are to each other. To compute the distance between adjacency matrices the sum of absolute differences is used. Before computing d_{ij}, each F_i is normalized by the $max_{j,k}((F_i)_{j,k})$, so we have

$$d_{ij} = \sum_{k=1}^{n} \sum_{l=1}^{n} |(F_i)_{k,l} - (F_j)_{k,l}| \tag{4.4}$$

Figure 4.3 shows the adjacency matrices from the experiment described in

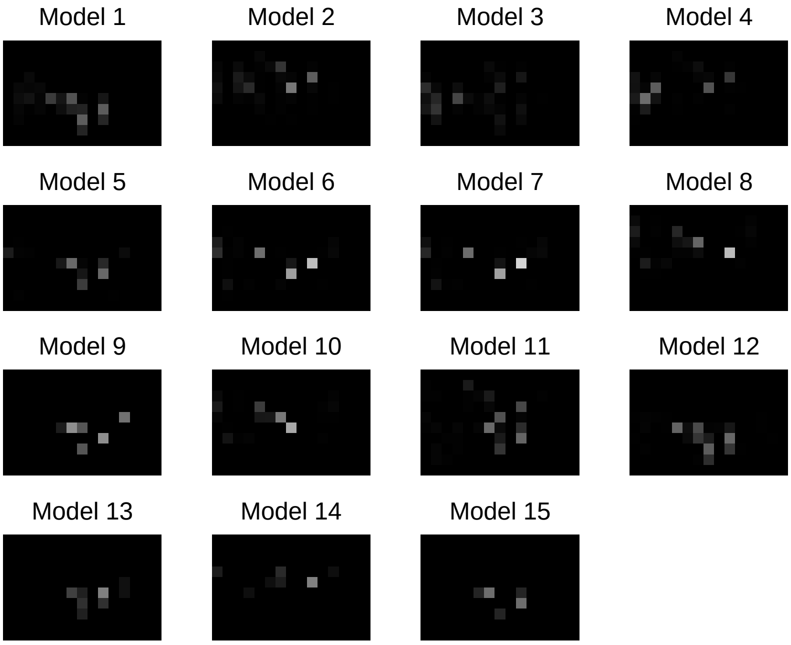


Figure 4.3: Adjacency matrices for 15 models in the experiment of section 4.5.2

section 4.5.2. 15 models were found after the first pass. Distances between adjacency matrices are computed and are plotted in Figure 4.4. The pairs with distance less than threshold (1.5) are circled in the figure.

In the example in Figure 4.3 and Figure 4.4, we can easily see that M_6 and M_7 must be the same model. Similarly pairs such as (M_1, M_{12}) and (M_5, M_{11}) appear to be the same model. More detailed result are discussed in section 4.5.2.

4.5 Experiments

We present two experiments. The first was conducted on a video set of 1212 frames from four video clips (D_6, D_7, D_9, and D_{10} in section 3.4) collected at different locations and under different illumination conditions. The second experiment

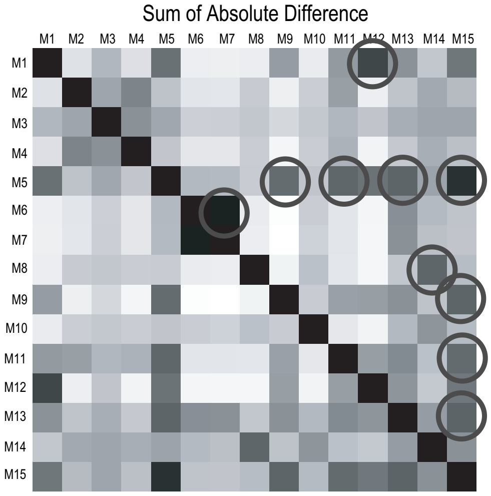


Figure 4.4: Differences between adjacency matrices in Figure 4.3. When two models are similar, the distance is very low and it is plotted dark. Low distance values are circled.

analyzes an 18 minute long video clip[1] of a meeting. In this meeting room video experiment, a face detection algorithm was used to determine an approximate torso area. In each experiment, we show the final gallery and the matching results based on the gallery.

The gallery construction process consists of two passes.

1. **Construct an initial gallery.** From an empty set, a gallery is built while processing all the frames. After this pass, the gallery has all the tentative models.

2. **Refine the gallery.** Using the gallery built in the first pass, all the frames are processed again. The gallery is not extended in this pass. Using the matching result, redundant models are removed based on frequency and spatial analysis (for meeting videos) to build a more compact and accurate gallery.

[1] Test video clip from BAE systems

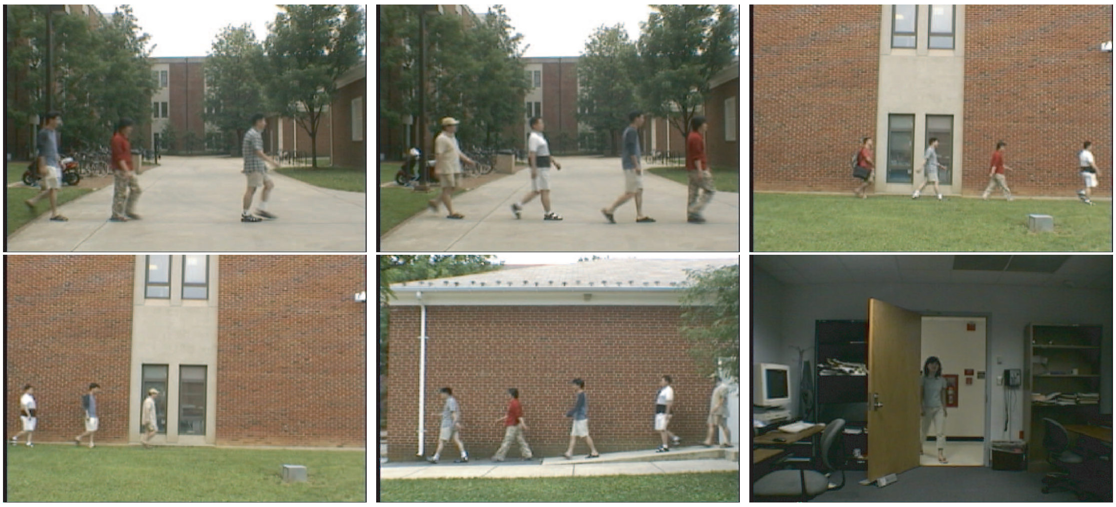


Figure 4.5: Sample frames of the full body gallery test.

4.5.1 Full Body Gallery - Experiment 1

For this experiment, 1212 frames were collected from four different video clips. Three clips were outdoor video, and one clip was captured in a room monitoring people coming and going. The number of people in the test set is 12. We employed a background subtraction algorithm to detect the foreground regions, and the detected regions are considered as full-body appearance of human. Figure 4.5 shows some of the images in this test set.

After the first pass, we have 24 models in the gallery. The second pass uses the static gallery of the 24 models. In this experiment, most redundancy comes from the inaccuracies of human silhouettes created by background subtraction. After the second pass, we have a final gallery of 16 models as shown in Figure 4.6. All 12 people were modelled, however 2 people have two models and 1 person has 3 models respectively. Models (a), (g), and (h) were constructed based on one person. Also, ((c), (i)) and ((n), (o)) are redundant model pairs.

Table 4.1: Matching result - Full body

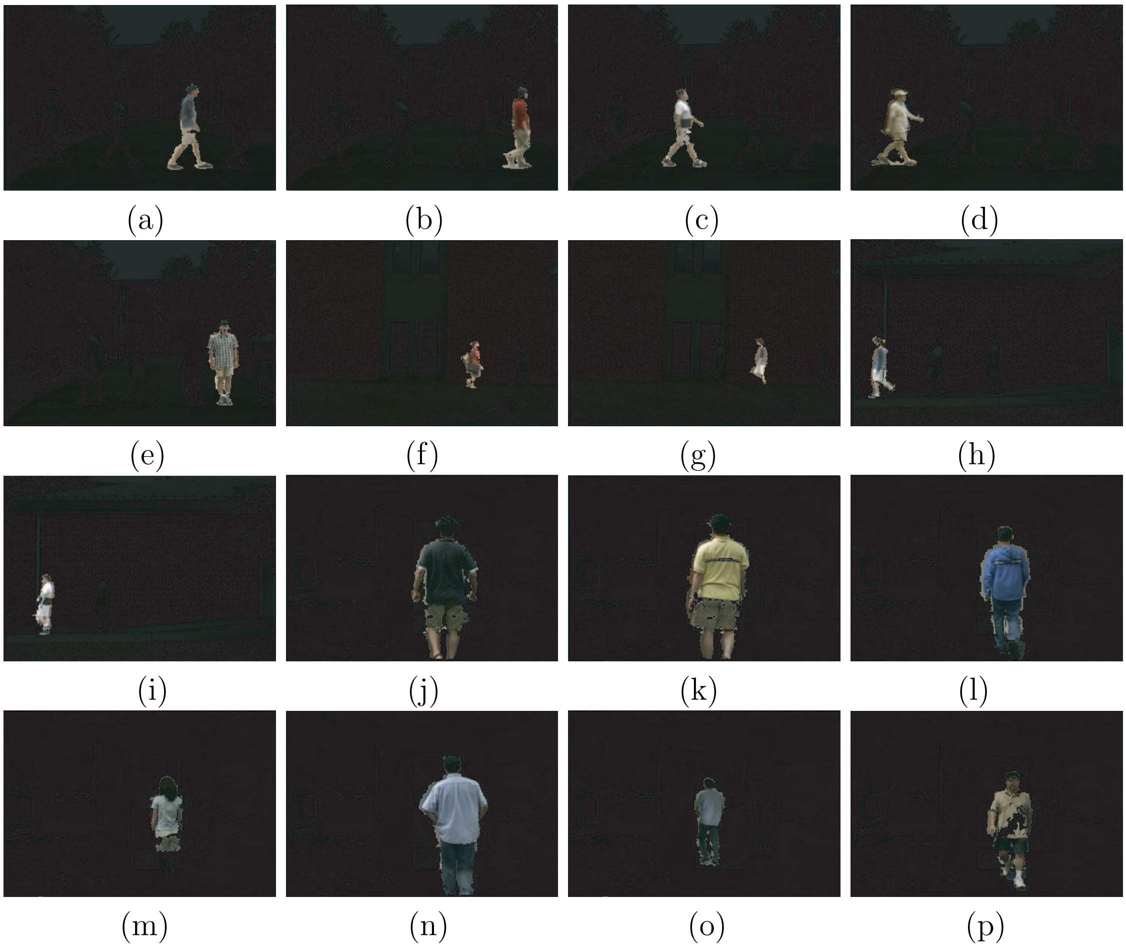
	Number	Correct	Incorrect	Matching
Gallery	of Models	Matching	Matching	Rate
Initial	24	1609	291	85.1%
Refined	16	1583	307	83.7%

In this data set, 1890 foreground areas are detected from the 1212 frames. Using the the final gallery with 16 models, we could match 1583 regions correctly, while 307 are mismatched (83.7% success). When we use the 24 model gallery before removing redundant models, the number of correct matches is 1609 and 291 regions are not matched correctly (85.1% success). The representation power of the gallery is dependent on data set and foreground segmentation results. When using the same segmentation results, the final gallery has similar representation power compared to the gallery before redundant model removal (Table 4.1).

4.5.2 Upper Body Gallery - Experiment 2

An 18 minute long video clip which has 8 people is used for this experiment. Although the number of total frames is 32,400, only one frame out of every five frames were processed. This video clip was captured in a meeting room, and people remain seated without position changes. The cameras pan and tilt as the meeting progresses, so that at any one time we see a different subset of the participants. Only the upper bodies of people are seen.

We employ a face detection algorithm to locate people [57]. Based on the detected face areas, the torso areas were computed as in Figure 4.8 and appearance model is built using the torso areas. Since the relative positions between people



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

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 4.6: The final gallery built with a test set of Figure 4.5. The number of models in the gallery is 16.

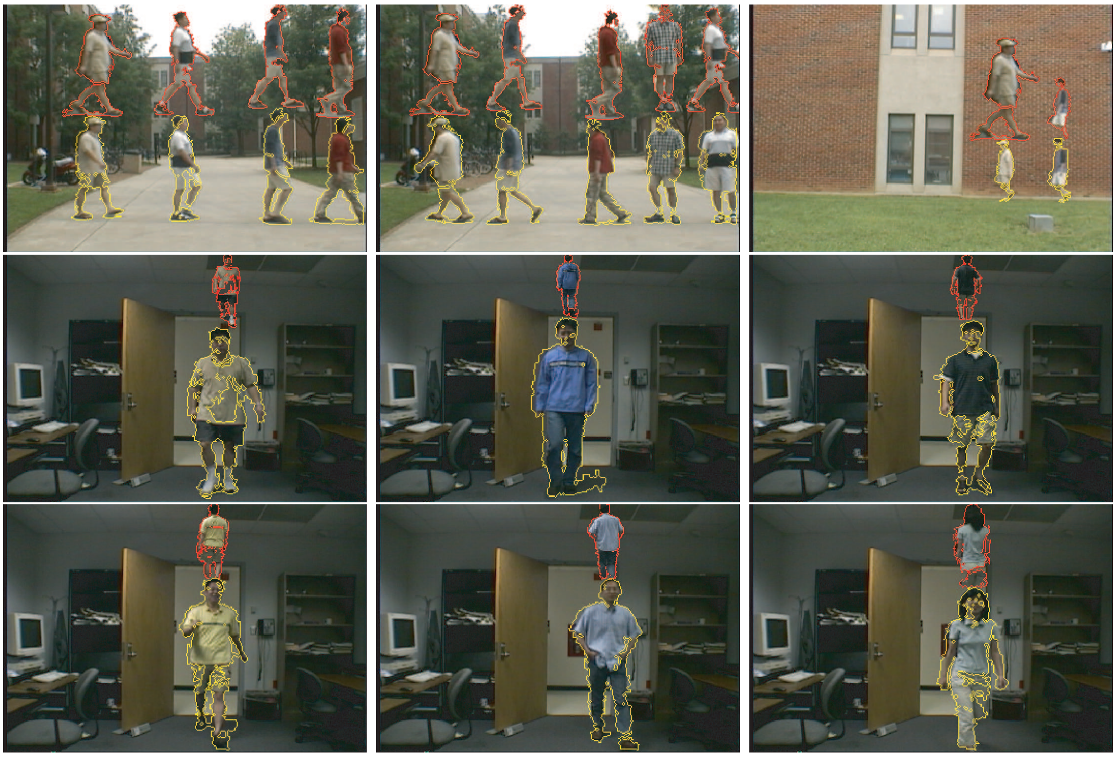


Figure 4.7: Some frames showing matching results with the final gallery.

remain unchanged for the entire clip, we perform the spatial analysis described in section 4.4.

Several frames are shown in Figure 4.9. The first pass constructed a 15 model gallery excluding false alarms from the face detector. Figure 4.10 shows these models.

In the second pass, the spatial analysis of relative horizontal positions was carried out. The adjacency matrices of the 15 models were shown in Figure 4.3 and the difference between the adjacency matrices was shown in Figure 4.4.

Before calculating the differences between adjacency matrices, the total frequency for each model is used to eliminate some models. The total number of face occurrences is 13,709, and some models have very low frequency. Table 4.2 shows

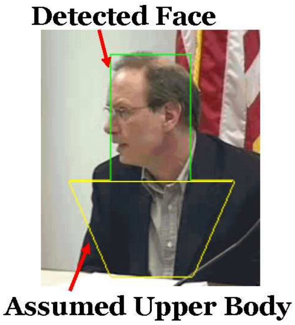


Figure 4.8: A torso area can be assumed based on detected face area. Experimentally, 2× (width of face) for top width of torso, the same with for bottom width, and 3/4 of face height for torso height are used.

Figure 4.9: Sample frame images from the video clip used in upper body gallery test

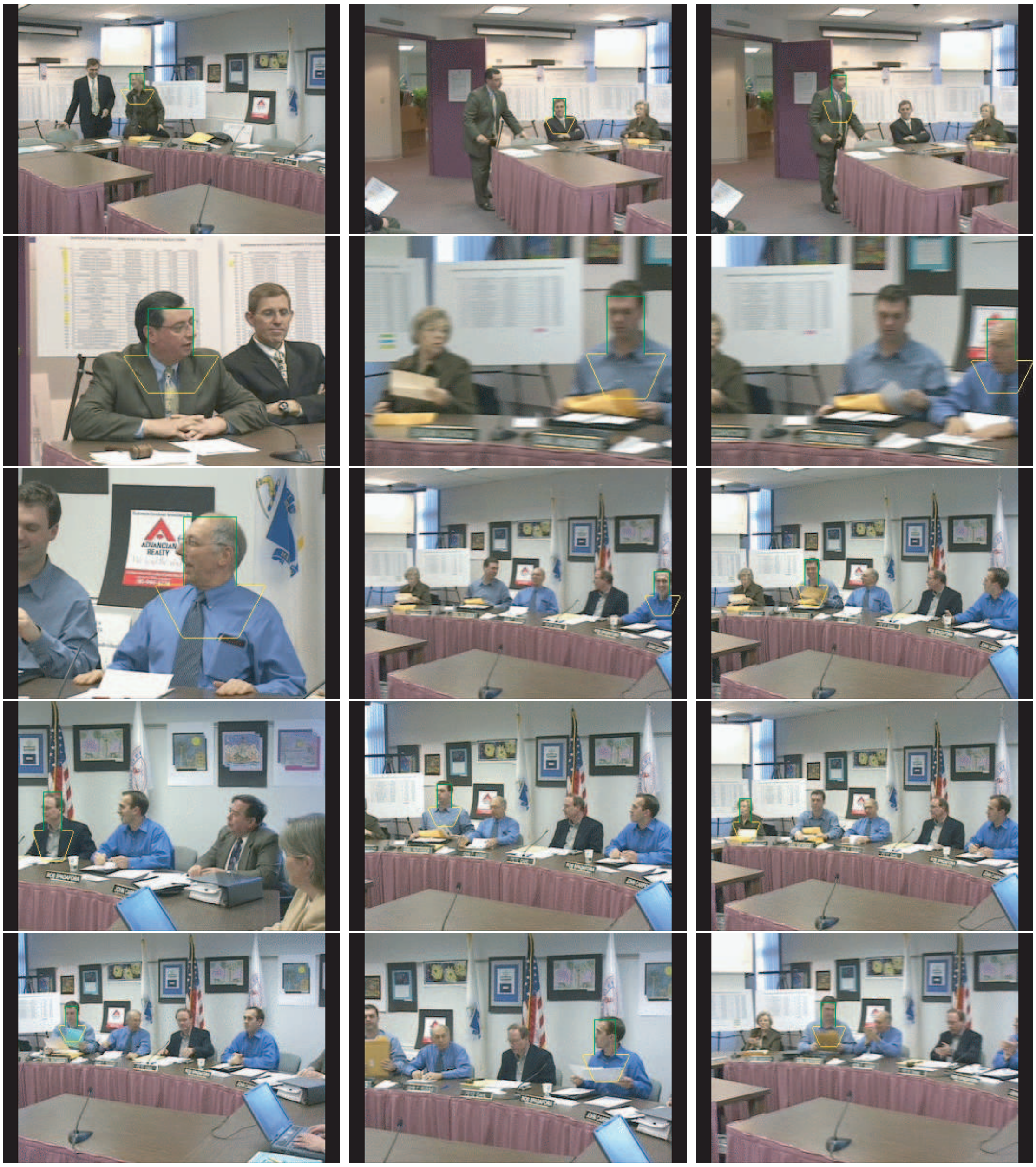


Figure 4.10: 15 models after the first pass.

Table 4.2: Frequency of each model

Model	M_1	M_2	M_3	M_4	M_5
Freq.	910	703	370	277	1945
Model	M_6	M_7	M_8	M_9	M_{10}
Freq.	426	1892	2997	9	3359
Model	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}
Freq.	97	221	16	18	7

the frequency of each model.

As seen in Table 4.2, M_9, M_{13}, M_{14}, M_{15} can be eliminated since their frequencies are very low. Most of the dropped models were built separately due to folders and hands which occlude the torso area. Next, by thresholding the differences between adjacency matrices, we select pairs of models, which can be merged into one.

(M_6,M_7), (M_5,M_{15}), (M_1,M_{12}), (M_{11},M_{13}), (M_5,M_{13})

(M_9,M_{15}), (M_5,M_{11}), (M_8,M_{14}), (M_{11},M_{15}), (M_5,M_9)

The final gallery has 8 models. In the video clip, although there are nine people appearing, the ninth person shows only side view and she was not detected by the face recognition algorithm. The eighth person was not included in the gallery, and two models were found for the first person. Table 4.3 shows the gallery we acquired. The merged models are shown in parentheses.

Figure 4.12 shows some of the matching results using the final gallery. To investigate the identification accuracy of matching, we randomly chose 100 frames which were found to have 210 face areas. The total number of faces is 13709 and 210 is around 1.5% of the data. Table 4.4 summarized the result. Just like in the

Table 4.3: Final Gallery

Person	Model
P_1	M_3,M_4
P_2	M_2
P_3	(M_1,M_{12})
P_4	(M_5,M_9,M_{11},M_{13},M_{15})
P_5	(M_6,M_7)
P_6	M_{10}
P_7	(M_8,M_{14})
P_8	NONE

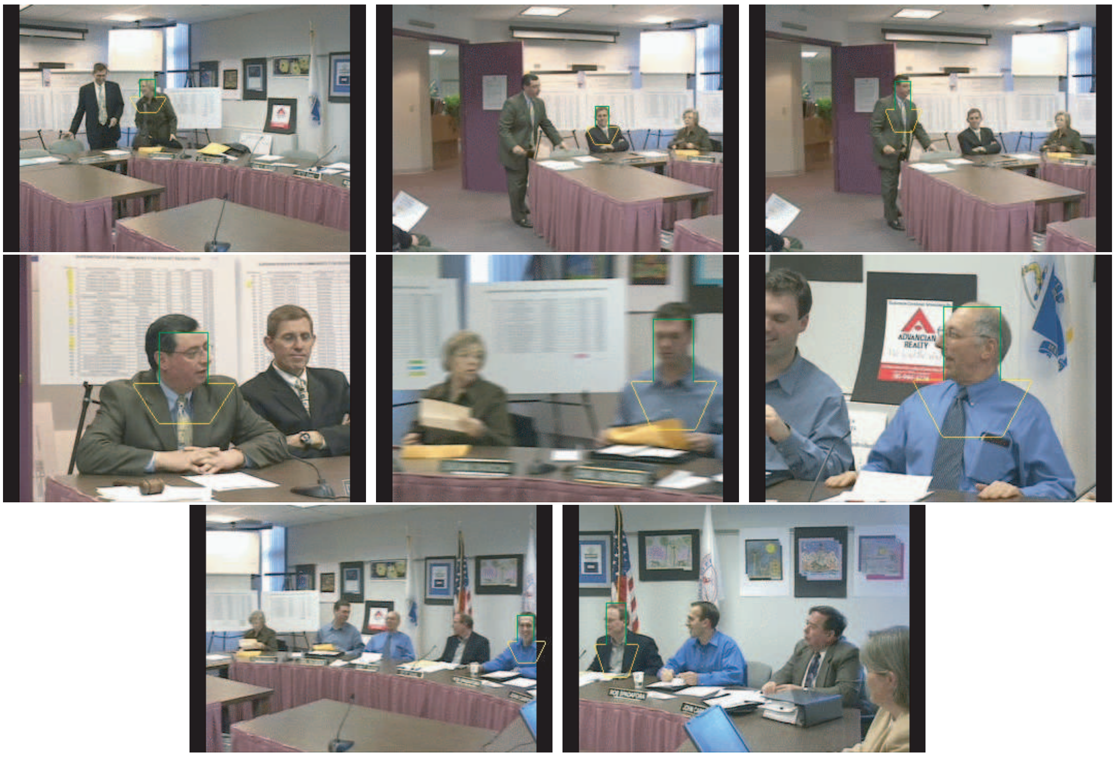


Figure 4.11: 8 models in the final gallery after the spatial analysis.

Table 4.4: Matching result - Upper body

Gallery	Number of Models	Correct Matching	Incorrect Matching	Matching Rate
Initial	15	198	12	94.3%
Refined	8	194	16	92.4%

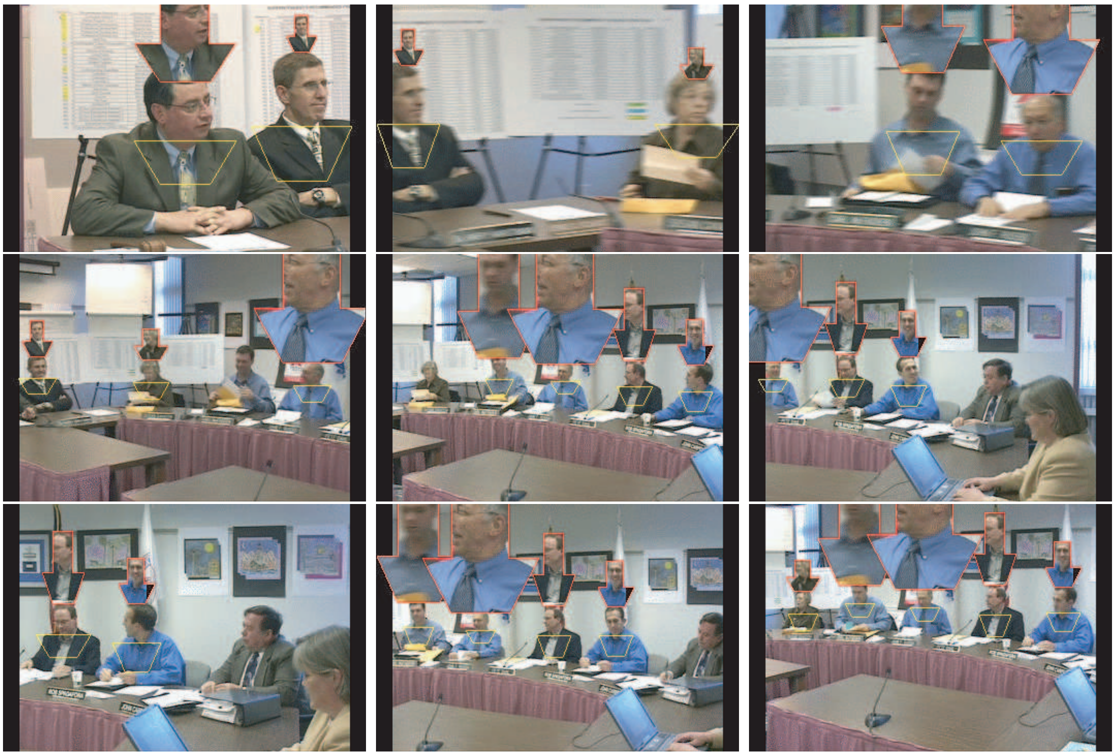


Figure 4.12: Some frames showing matching results with the final gallery in the second experiment.

experiment in Section 4.5.1, even with the smaller number of models the gallery shows the similar performance.

4.6 Conclusion

We proposed an approach for constructing a dynamic gallery of people from a video clip or a set of frame images based on appearance model using color/path-length profile. Kullback-Leibler distance is used to robustly compare models and a one-to-one constraint is enforced when more than one instance is present and matched at a frame. When the order of people rarely changes, the relative spatial order is analyzed and used to reduce the redundant models from the gallery.

Chapter 5

Conclusion and Future Work

5.1 Summary

We presented a region-based BGS which improves the performance of pixel-based BGS by employing spatial information in the detection phase. For recognition of people, we introduced a new feature, *path-length*, and used KDE and KL distance as the distance measure between appearances. We showed that the appearance model can be used to locate the local changes which might be caused by carried packages. Through a set of experiments, we showed the robustness of our recognition method. Lastly, we presented a method to construct a human gallery from a given video clip using the appearance model and dynamic gallery management.

5.2 Future Directions

For region-Based BGS, more systematic and comprehensive performance evaluation of segmentation based BGS should be done. Also, there is no established metric that guarantees best segmentation. We will study more about various metrics to obtain a better segmentation hierarchy, and it could use two separate metrics in generating basic segmentation and in building the hierarchy of segmentation. The selection of best segmentation from the hierarchy and finding better aggregation method are other issues that should be considered.

In appearance-based recognition and gallery construction, the following is the list of open issues which should be studied in the future.

1. Using a better color models in appearance model
2. Automatic estimation of threshold values in appearance comparison
3. Multiple frame based model - how can we combine appearance information as a person is tracked continuously?
4. Multiple frame based comparison - how can we improve matching as a person is tracked continuously?
5. Better use of segmentation information in local change detection
6. More effective gallery management, using the methods developed in 3 and 4

Appendix A

Symmetric Neighborhood Filter (SNF)

We use SNF as the first step to segment an image. The SNF is an image enhancement filter first introduced by Harwood et. al in [28]. It is designed to smooth the interiors of homogeneous regions while simultaneously enhancing blurred edges. Although it has been employed in several approaches [27, 58, 3, 47], it has not been described in detail. Hence, we summarize the algorithm here.

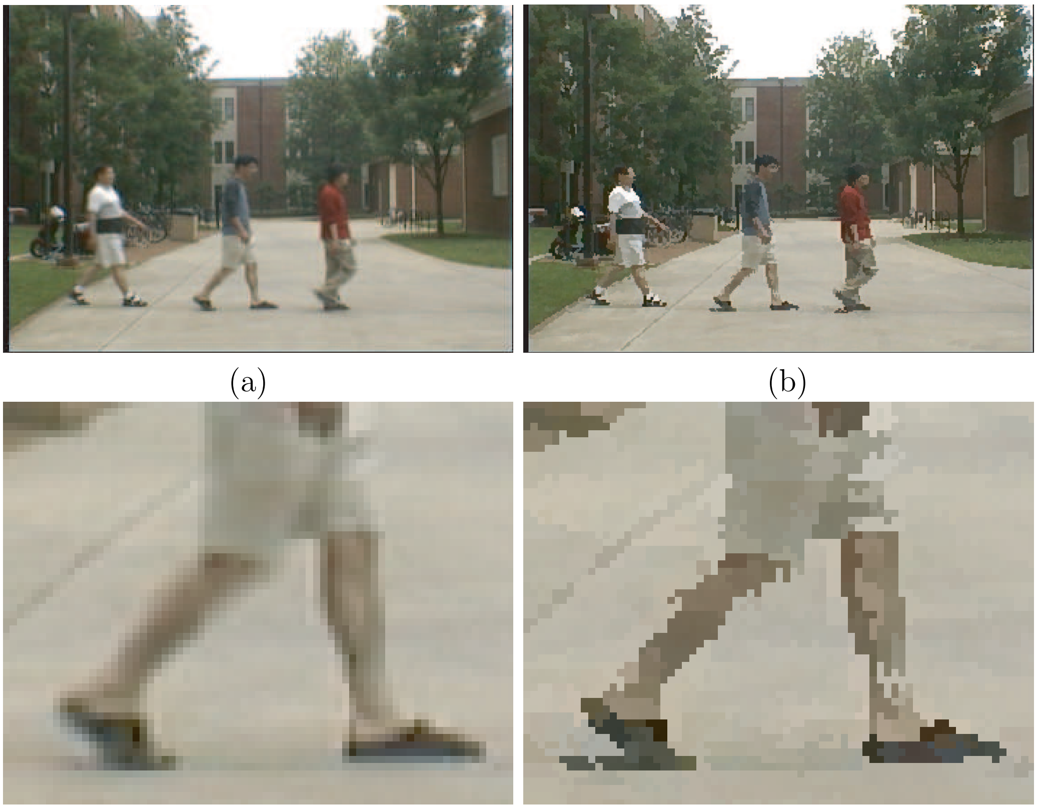
For each pixel, the SNF selects half the number of pixels in its neighborhood by selecting one pixel nearest in gray level to the center pixel from each pair of pixels located symmetrically opposite the center. When two pixels are equidistant to the center pixel or the nearest distance is greater than the given parameter (ϵ), the center pixel is selected. The collection of four pixels are averaged together, and finally the center pixel is replaced by the mean of the center pixel value and this average.

The following is the SNF ($(2n+1) \times (2n+1)$ size) algorithm for one iteration.

```
Algorithm: SNF-One-Iteration
for each pixel $(x, y)$
    $s \leftarrow 0$
    for each pair of pixels $\{(x+i, y+j), (x-i, y-j)\}$ where $-n \leq i, j \leq n$
        $d = min(|g(x+i, y+j) - g(x, y)|, |g(x-i, y-j) - g(x, y)|)$;
        if $d > \epsilon$
            $s \leftarrow s + g(x, y)$
        else if $(|g(x+i, y+j) - g(x, y)| < |g(x-i, y-j) - g(x, y)|)$
            $s \leftarrow s + g(x+i, y+i)$
        else if $(|g(x+i, y+j) - g(x, y)| > |g(x-i, y-j) - g(x, y)|)$
            $s \leftarrow s + g(x-i, y-j)$
        otherwise
            $s \leftarrow s + g(x, y)$
        endif
    endfor
    $m \leftarrow ((2n+1) \times (2n+1) - 1)/2$
    $g(x, y) \leftarrow (s/m + g(x, y))/2$
endfor
```

$g(x, y)$ is the pixel value of pixel (x, y), and ϵ is a single parameter of SNF filter. Although SNF can be applied with any $n \times n$ neighborhood, we use 3×3 SNF in our approach.

Three steps of the SNF is applied with different parameters. The first step runs with $\epsilon = 0$ for four iterations to preserve edges. The second step runs to flatten the interior of regions with $\epsilon = \kappa\sigma$ for 200 iterations. If the all the pixels are fixed, it stops even before 200 iterations. Here, σ is the noise level of the image, which is the median of the standard deviation of all $(2n+1) \times (2n+1)$ neighborhoods. κ is a constant, usually 2.0. To sharpen the borders of regions, the third step runs with $\epsilon = 0$ and maximum number of iterations is 200.



(a)

(b)

(c)

(d)

Figure A.1: An example image of SNF. (a) is the original image, and the result of SNF is (b). (c) and (d) show the difference in detail.

Appendix B

Algorithm for *Path-length*

We compute *path-length* of pixels in the given foreground region using a queue. The following algorithm computes *path-length* of all the points, sorts and stores them in a list.

```
Algorithm: Path-Length Computation
Q.insert(head point)
$L \leftarrow 0$
repeat
    $p \leftarrow Q.remove()$
    $L.insert(p)$
    for each neighbor of $p$, $q$
        if ($q$ is in the silhouette) and ($q$ is unvisited)
            compute path-length of $q$ using path-length of $p$
            $Q.insertAtCorrectPosition(q)$
        endif
    endfor
until $Q$ is empty
```

Q is a temporary queue, and L is the result list with all the points in the silhouette sorted with respect with *path-length.* $Q.insertAtCorrectPosition$ finds the correct position from Q, and guarantees that points in Q are in order all the time.

BIBLIOGRAPHY

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, November 2004.

[2] D. C. Alexander and B. F. Buxton. Statistical modeling of colour data. *International Journal of Computer Vision*, 44(2):87–109, 2001.

[3] David A. Bader, Joseph JaJa, David Harwood, and Larry S. Davis. Parallel algorithms for image enhancement and segmentation by region growing with an experimental study. In *IPPS '96: Proceedings of the 10th International Parallel Processing Symposium*, pages 414–423, Washington, DC, USA, 1996. IEEE Computer Society.

[4] Chiraz BenAbdelkader, Larry S. Davis, and Ross Cutler. Motion-based recognition of people in eigengait space. In *FGR*, pages 267–274, 2002.

[5] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *CVPR '98: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 232, Washington, DC, USA, 1998. IEEE Computer Society.

[6] J.L. Blue, G.T. Candela, P.J. Grother, R. Chellappa, and C.L. Wilson. Evaluation of pattern classifiers for fingerprint and OCR applications. *Pattern Recognition*, 27(4):485–501, April 1994.

[7] T. H. Chalidabhongse, K. Kim, D. Harwood, and L. Davis. A perturbation method for evaluating background subtraction algorithms. In *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, October 2003.

[8] Rama Chellappa, Charles L. Wilson, and Saad Sirohey. Human and machine recognition of faces: A survey. *Proceedings of the IEEE*, 83(5):705–740, May 1995.

[9] David Coleman, Paul Holland, Neil Kaden, Virginia Klema, and Stephen C. Peters. A system of subroutines for iteratively reweighted least squares computations. *ACM Trans. Math. Softw.*, 6(3):327–336, 1980.

[10] Robert Collins, Ralph Gross, and Jianbo Shi. Silhouette-based human identification from body shape and gait. In *Intl' Conference on Face and Gesture*, pages 351–356, October 2002.

[11] D. Comaniciu and V. Ramesh. Robust detection and tracking of human faces with and active camera. In *IEEE Int'l Workshop on Visual Surveillance,* Dublin, Ireland, pages 11–18, 2000.

[12] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* Hilton Head, SC, volume II, pages 142–149, June 2000.

[13] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *CVPR*, pages 2142–, 2000.

[14] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* New York: Wiley, 1991.

[15] M. Cristani, M. Bicego, and V. Murino. Integrated region- and pixel-based approach to background modelling. In *Proc. IEEE Workshop on Motion and Video Computing*, 2002.

[16] Carnegie Mellon Image Database. http://vasc.ri.cmu.edu/idb/.

[17] R. O. Duda, D.G. Stork, and P. E. Hart. *Pattern Classification.* John Wiley and Sons Inc., 2000.

[18] 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*, 90(7):1151–1163, July 2002.

[19] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *Proc. European Conf. on Computer Vision,* Dublin, Ireland, volume II, pages 751–767, June 2000.

[20] J. Fox. Robust regression: Appendix to an r and s-plus companion to applied regression, 2002.

[21] J. Freixenet, X. Munoz, D. Raba, J. Marti, and X. Cufi. Yet another survey on image segmentation: Region and boundary information integration. In *ECCV 2002, LNCS 2352*, pages 408–422, 2002.

[22] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[23] U. Halici, L. C. Jain, and A. Erol. Introduction to fingerprint recognition. pages 1–34, 1999.

[24] I. Haritaoglu, R. Cutler an D. Harwood, and L.S. Davis. Backpack: detection of people carrying objects using silhouettes. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 102–107, September 1999.

[25] I. Haritaoglu, D. Harwood, and L. Davis. Ghost: A human body part labeling system using silhouettes. In *Proceedings of Fourteenth International Conference on Pattern Recognition*, volume 1, pages 77–82, August 1998.

[26] M. Harville. A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models. In *European Conf. Computer Vision*, volume 3, pages 543–560, 2002.

[27] D. Harwood, S. Chang, and L. S. Davis. Interpreting aerial photographs by segmentation and search. In *Proc. of the Image Understanding Workshop*, pages 507–520, Los Angeles, CA, 1987.

[28] David Harwood, Muralidhara Subbarao, Hannu Hakalahti, and Larry S. Davis. A new class of edge-preserving smoothing filters. *Pattern Recogn. Lett.*, 6(3):155–162, August 1987.

[29] H.G.Barrow and J.Tenenbaum. Recovering intrinsic scene characteristics from images. pages 3–26, 1978.

[30] T. Horprasert, D. Harwood, and L. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proceedings of IEEE ICCV'99 FRAME-RATE Workshop*, September 1999.

[31] T. Horprasert, D. Harwood, and L. Davis. A robust background subtraction and shadow detection. In *Proceedings of ACCV'2000*, January 2000.

[32] Peter J. Huber. *Robust Statistical Procedures*. Society for Industrial and Applied Mathematics, 1977.

[33] J. N. Kapur and H. K. Kesavan. *Entropy Optimization Principles with Applications*. Academic Press, 1992.

[34] Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood, and Larry Davis. Real-time foreground-background segmentation using codebook model. 11:172–185, June 2005.

[35] H. W. Kuhn. The hungarian method for the assignment problem. 2(83), 1955.

[36] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

[37] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Proceedings of the Workshop on Statistical Learning in Computer Vision*, Prague, Czech Republic, May 2004.

[38] X. F. Li, R. J. Logan, and R. E. Pastore. Perception of acoustic source characteristics: Walking sounds. 90(6):3036–3049, 1991.

[39] H. Ling and D. Jacobs. Using the inner-distance for classification of articulated shapes. In *International Conference on Computer Vision and Pattern Recognition*, 2005.

[40] Kaj Mäkelä, Jaakko Hakulinen, and Markku Turunen. The use of walking sounds in supporting awareness. In *Proceedings of the 2003 International Conference on Auditory Display,* Boston, MA, USA, July 2003.

[41] E. Marszalec, B. Martinkauppi, M. Soriano, and M. Pietikäinen. A physics-based face database for color research. *Journal of Electronic Imaging*, 9(1):32–38, 2000.

[42] Jérôme Martin, Vincent Devin, and James L. Crowley. Active hand tracking. In *FG*, pages 573–578, 1998.

[43] Stephen J. McKenna, Yogesh Raja, and Shaogang Gong. Object tracking using adaptive color mixture models. In *ACCV (1)*, pages 615–622, 1998.

[44] Anurag Mittal and Larry S. Davis. M_2Tracker: A multi-view approach to segmenting and tracking people in a cluttered scene. 51(3):189–203, 2003.

[45] Chikahito Nakajima, Massimiliano Pontil, Bernd Heisele, and Tomaso Poggio. Full-body person recognition system. *Pattern Recognition*, 36(9):1997–2006, 2003.

[46] Amsterdam Library of Object Images. http://staff.science.uva.nl/ aloi/.

[47] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29:51–59, 1996.

[48] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 1988.

[49] Ferdinando Samaria and Andy Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision,* Sarasota FL, December 1994.

[50] D. W. Scott. *Multivariate Density Estimation.* Wiley Interscience, 1992.

[51] B. W. Silverman. *Density estimation for statistics and data analysis.* Chapman & Hall, New York, 1986.

[52] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression (pie) database. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, May 2002.

[53] P. Sinha, E. Osuna, M. Oren, C. Papageorgiou, and T. Poggio. Pedestrian detection using wavelet templates. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition,* Puerto Rico, pages 193–199, June 1997.

[54] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition,* Fort Collins, CO, pages 246–252, 1999.

[55] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 762–769, Washington, DC, June 2004.

[56] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Int. Conf. Computer Vision*, pages 255–261, 1999.

[57] Paul A. Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR (1)*, pages 511–518, 2001.

[58] T. Westman, D. Harwood, T. Laitinen, and M. Pietikainen. Color segmentation by hierarchical connected component analysis with image enhancement by symmetric neighborhood filters. In *Proceedings of the 10th International Conference on Computer Vision and Pattern Recognition Systems and Applications*, pages 769–802, 1990.

[59] R.P. Wildes. Iris recognition: an emerging biometric technology. In *Proc. IEEE 85*, pages 1348–1363, 1997.

[60] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):780–785, 1997.

[61] W. Zhao, R. Chellappa, A. Rosenfeld, and P. Phillips. Face recognition: A literature survey, 2000.

[62] Shaohua Zhou and Rama Chellappa. Probabilistic human recognition from video. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part III*, pages 681–697, London, UK, 2002. Springer-Verlag.

[63] Yong Zhua, Tieniu Tan, and Yunhong Wang. Biometric personal identification based on iris patterns. 02(2):2801, 2000.