

AN INVESTIGATION INTO VIEWPOINT INVARIANT PEDESTRIAN
RECOGNITION FOR SURVEILLANCE SYSTEMS

BY

BRIAN MATTHEW STIEBER

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Professor Thomas S. Huang

ABSTRACT

This is a study of viewpoint invariant appearance models for pedestrian recognition. The study investigates basic models, color histograms and region based histograms, to gain intuition about the appearance models as well as create new baseline results for evaluating future appearance models. This insight is then used to create an appearance model based on large-margin nearest-neighbor classification. This model significantly outperforms the current state of the art. Finally, the IFP Image Processor, a basic framework for implementation of surveillance systems algorithms, is introduced.

To my parents and sisters for their love, support and encouragement

ACKNOWLEDGMENTS

I would like to thank my adviser, Prof. Thomas S. Huang, who gave me the opportunity to pursue my interests in computer vision. Prof. Huang provided me with the guidance and support needed to successfully complete the thesis process. He also introduced me to an intelligent and helpful Image Formation and Processing group. This group of individuals was invaluable in the research process.

I also owe a great debt of gratitude to my lab mate Mert Dikmen. Merts expertise in computer vision, programming and knowledge of the vast literature corresponding to the field, proved to be instrumental to my success.

I would like to thank my father, Peter J. Stieber, for mentoring me through several obstacles in the creation of the IFP Image Processor.

This project used the implementation of the GMM background model by Zoran Zikovic and Kilian Weinberger's Matlab implementation of large-margin nearest-neighbor classification, as well as the VIPeR dataset from the University of California Santa Cruz.

TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION	1
1.1	Literature Review	2
1.2	Contributions	5
CHAPTER 2	APPEARANCE MODELING	7
2.1	VIPeR Dataset	7
2.1.1	Viewpoint Variation	7
2.1.2	Other Variations	8
2.2	Performance Evaluation and Methodology	9
2.3	Color Histograms	10
2.3.1	Distance Measures	11
2.3.2	Results	12
2.4	Regional Histogram Models	15
2.4.1	Hand Localized Histogram	16
2.4.2	Block Histograms	19
2.5	Large-Margin Nearest-Neighbor Distance Learning	19
2.5.1	Background	19
2.5.2	Experiments and Results	24
CHAPTER 3	IFP IMAGE PROCESSOR	27
3.1	Fusion Center	27
3.1.1	Tracking	28
3.1.2	Association Across Non-Overlapping Fields of View	30
3.2	Image Processor	30
3.2.1	Background Modeling	31
3.2.2	Detection and Feature Extraction	33
3.3	Interface	35
3.3.1	Creating a Case	35
3.3.2	Editing Fusion Center	36
3.3.3	Editing Image Processor	37
3.3.4	Display	39
REFERENCES	40

CHAPTER 1

INTRODUCTION

Video surveillance systems have been researched and developed for several decades. Recently, imaging technology has been developed to the point where it is economical to deploy surveillance systems with several sensors [1]. Such systems are now in place in many public places such as airports, banks, shopping malls, public buildings, government bases, prisons and hospitals [2]. However, as the number of sensors in these system increases, so does the number of human operators needed to monitor their activity. This creates additional costs for the system and introduces human errors caused by fatigue, inattention, and the inability to continuously monitor all sensors. In order to eliminate these costs and errors, there has been an effort to create automatic surveillance systems that can monitor human activity with fewer human operators [3]. These systems aim to detect threatening activity of objects and alert human operators.

Automatically detecting threatening behavior requires long term tracking of objects in the environment. This is particularly important when trying to learn patterns of behavior and then detect anomalous activity. In these situations, it is necessary to track through multiple fields of view (FOVs). This is due to gaps in sensor coverage caused by an inadequate number of cameras or obstruction from environmental constraints. These gaps in coverage create the need to solve the target reacquisition problem. That is, when a new object is detected in a camera's FOV, one must determine whether that object is new to the system, and thus a new track, or associated with previous track that has left a different camera's FOV. This

becomes increasingly difficult when dealing with pedestrians due to large variations in pose. The focus of this thesis is to investigate methods for resolving these track ambiguities. This is done by evaluating the models used to represent people and creating a system capable of using these models to track pedestrians across cameras with non-overlapping FOVs.

1.1 Literature Review

Proposed surveillance systems vary significantly based on scenario. This includes the type of objects being tracked and known information about the environment, camera locations and camera parameters. Depending on the scenario, a combination of spatial, temporal and appearance information is used to match objects across cameras. Some of the earliest work on target reacquisition [4] used Gaussian probability distribution functions to model vehicles based on velocity, size and hue-saturation-value (HSV) color information, and subsequently match cars between two camera views.

In scenarios with known camera and environment information, it is possible to incorporate this information to solve the human reacquisition problem. For example, Loke et al. [5] used known environmental information to project pedestrians onto a common ground plane view. New pedestrians were then matched using a fuzzy logic system based on their position and motion in the common ground plane view, as well as shape information. Similarly, Lim et al. [6] accomplished matching using a map of the surveillance area, which provides information of possible path trajectories across non-overlapping regions. These trajectories are combined with appearance models in a likelihood framework to determine possible matches. Finally, Pflugfelder and Bischof [7] proposed a method for tracking in 3-D space without using appearance. Rather, the geometry between cameras is used by

expanding linear inhomogeneous triangulation by a Gaussian random walk model. Tracking is then cast as triangulation followed by a re-projection.

In scenarios with unknown environment and camera information, it becomes necessary to include appearance models for target reacquisition. Appearance models are found extensively in the literature due to their suitability in image retrieval, tracking, and recognition. The models also vary depending on the context in which they are used.

Islam et al. [8] proposed the use of shape and color templates to model the appearance of a pedestrian. However, this method is extremely simplistic and fails when pose or viewpoint changes occur. Flexible alignment and matching are possible solutions to the viewpoint changes [9], but this only shows promise for ridged objects. Several researchers have attempted subspace [10] and manifold [11] methods to alleviate errors caused by pose and viewpoint variations. However, local deformations of nonrigid objects are complex and difficult to represent accurately.

Local region descriptor methods have also been proposed to combat variations in pose and viewpoint. These methods select regions or points in an image and create a numerical descriptor to represent each region. Region descriptors are then compared across images to develop a correspondence and matching score. Wang et al. [12] use histograms of oriented gradients as descriptors of different body parts. Similarly, Gheissari et al. [13] describe salient body segments by combining color and edge histograms. Lowe [14] presented the popular scale invariant feature transform (SIFT) that creates descriptors invariant to scale, rotation, illumination, and 3D viewpoint changes. Some of the most recent methods [15] use a cascade of region descriptors.

An extension of local descriptors is the bag-of-features approach. In this method, descriptors belong to a closed set, or dictionary. A histogram of occurrences of these descriptors are used as a representation of the object. The closed set can be learned

by clustering local region features over a large database. These features include those previously described. A slight deviation is constellation methods, which include relative spatial information in features to improve performance [16].

There are also methods that do not belong to any of the standard categories. For example, Yu et al. [17] created a model based on color and path-length features of the pixels of a detected person. Path-length is the length of the shortest path from the top of the head to a point entirely within the body. It attempts to describe structural information about the person. It is invariant to 2D-articulations. This makes path-length feature less sensitive to human motion than features based on spatial positions. Path-length is combined with robust color features in a probabilistic framework.

Despite the large volume of literature related to appearance models, the most common appearance model used is color histograms [18, 19]. Color histograms are robust to extreme non-rigid deformations. However, the lack of geometric information makes it impossible to discriminate similar color distributions with dramatic structure differences. Several attempts have been made to integrate spatial information into the histograms. Huang et al. [20] used color correlograms, which save color correlation as a function of distance. Birchfield and Rangarajan [21] introduced the concept of a spatiogram, which is a generalization of a histogram that includes spatial means and covariances for each histogram bin. Gheissari et al. [13] apply a geometric transform to parts of the human body, to create a pose invariant model. However, this requires a solution to segmenting a body into known parts.

One of the main sources of errors when comparing any appearance model across viewpoint changes is the illumination change. This can be caused by a change in scene locations and camera models. Javed et al. [22] proposed a possible solution, which learns a brightness transfer function across views and compensates accordingly. This may be well suited for surveillance systems, but the lack of

camera information corresponding to the image pairs in a dataset makes the method inapplicable. In those cases, methods such as histogram equalization [23] and greyworld normalization [24] have been proposed [25] to alleviate illumination errors.

Until recently, it has been difficult to evaluate appearance models for pedestrian reacquisition across non-overlapping fields of view. Models typically cater to the particular dataset used by the researcher which are biased to particular views. However, Gray and Tao [26] recently created a dataset designed specifically for viewpoint invariant pedestrian recognition (VIPeR). The VIPeR dataset contains 632 pedestrian image pairs from arbitrary viewpoints. Gray then proposed a viewpoint invariant appearance model [27] by training an ensemble of localized features (ELF) created with the AdaBoost algorithm. Alahi et al. [15] have since used the dataset to select optimal parameters in a novel appearance model used to track objects across cameras with non-overlapping fields of view. However, the authors significantly reduce the number of samples used in the data set, making it difficult to evaluate their model relative to the baseline methods.

1.2 Contributions

Despite the creation of a challenging and useful VIPeR dataset, a full investigation into baseline models has not been conducted. The original study by Gray was limited to a single color space and distance metric. This work serves as an investigation into a larger variety of appearance models including histograms and spatial histograms. A new feature is presented that exploits the dataset using large-margin nearest-neighbor (LMNN) distance metric learning and performs better than the current state of the art.

Additionally, the IFP Image Processor software application is introduced. The IFP Image Processor is an open source application for developing and testing

surveillance algorithms. It was developed to fill the need for a simple application that can be used to evaluate pedestrian tracking across cameras with non-overlapping fields of view. The IFP Image Processor is architected to allow developers to alter each stage of the surveillance system processing sequence and currently utilizes well known libraries and algorithms.

CHAPTER 2

APPEARANCE MODELING

As previously discussed, appearance models are essential to matching pedestrians across cameras with non-overlapping fields of view. In this chapter, several possible appearance models are evaluated for use as viewpoint invariant pedestrian features. Before describing the appearance models and their results, the VIPeR dataset and performance evaluation techniques are introduced.

2.1 VIPeR Dataset

The ViPeR dataset is used to develop and test appearance models. The dataset was created at the University of California Santa Cruz and is available at <http://vision.soe.ucsc.edu/?q=node/178>. The VIPeR dataset is the most challenging and complete dataset currently available for testing pedestrian appearance models with viewpoint variations. While previous datasets [28, 13] have been biased to a particular view, the VIPeR dataset aims to test appearance models that claim viewpoint invariance. This is done by collecting images of 632 pedestrians from two arbitrary viewpoints. Example image pairs are illustrated in Figure 2.1.

2.1.1 Viewpoint Variation

The dataset is quantized into 45 degree viewpoint segments. Thus a total of 8 possible viewpoints and 28 possible viewpoint pairs exist. Possible pairs can be reduced to 10, if symmetry is considered. Ideally these viewpoint pairs would be



Figure 2.1: Example pairs of images in the VIPeR dataset.

sampled uniformly to provide a viewpoint invariant dataset. Unfortunately, the VIPeR dataset does not satisfy this condition. The VIPeR dataset is biased towards frontal and side views, which also biases the viewpoint change to 90 degrees. However, this bias is weak compared to previously introduced datasets. A complete description of viewpoint sampling and viewpoint disparity sampling can be found in Tables 2.1 and 2.2 respectively.

2.1.2 Other Variations

The image pairs were collected over a period of weeks at varying locations and times of day. This combined with the fact that no illumination constraints were imposed between cameras, allowed for significant lighting variations in the image pairs. The images were originally part of video sequences and image quality varies significantly

Table 2.1: Distribution of viewpoint angles

Viewpoint Angle	0	45	90	135
45	16			
90	241	47		
135	43	72	4	
180	102	53	50	3

Table 2.2: Distribution of angle disparity

Viewpoint Angle Disparity	Examples
45	70
90	363
135	96
180	103

due to artifacts caused by compression and interlacing. The images were also cropped and scaled to remove biometric information, which introduced distortion.

2.2 Performance Evaluation and Methodology

The performance of an appearance model used for recognition can be difficult to quantify due to dependence on the size of the dataset. As the dataset size increases, the probability of recognition of a particular sample tends towards zero. Thus, to evaluate the models more effectively the recognition rate is modeled as a ranking problem rather than a same/different problem. In this study, the cumulative matching characteristic (CMC) curve and synthetic reacquisition rate (SRR) proposed by Gray and Tao [26] are used for evaluation. The CMC is found by first computing the distance from a sample image in a primary view to all the images in the secondary view. These distances are then ranked in order of increasing distance with the rank of the correct match defined as R_c . The CMC for a given rank, R , is the percentage of samples whose correct match, R_c , is less than or equal to R . This probability is plotted over the size of the data set. The SRR is found by transforming the CMC to represent a reacquisition problem. The SRR assumes that the set of M pedestrians in the camera network are i.i.d. samples from the testing set of size N . Suppose M pedestrians cross a coverage gap from one field of view to another. For reacquisition it is necessary to identify the correct matching

configuration. Ignoring the one to one matching constraint, the probability that any of the M best matches is correct is computed as

$$SRR(M) = CMC\left(\frac{N}{M}\right) \quad (2.1)$$

In order to evaluate the appearance models, the dataset was divided into two sets of 316 image pairs. Image pairs for each set are chosen at random. The first set is used to train models when necessary. The second set is used for model evaluation. In order to quantify the results for numerical comparison, the normalized area under the CMC curve is calculated.

2.3 Color Histograms

Color histograms are the simplest form of appearance model. Color histograms are also the most commonly used appearance model due to their ability to represent objects regardless of changes in rotation and scale. For this reason, they are used to generate baseline results when evaluating new models. When given an $n \times m$ image, I , quantized in to b colors, $c_1 \dots c_b$, a histogram can formally be written as follows:

$$H_I(c_i) = \sum_{x=1}^m \sum_{y=1}^n (I(x, y) = c_i) \text{ for } i = 1, \dots, b \quad (2.2)$$

Color histograms can exist in several forms that vary in color space, binning and structure. In these experiments, baseline results are developed by investigating the red-green-blue (RGB), HSV and luminance-blue chrominance-red chrominance (YCbCr) color spaces. The structure of the histograms is created by considering each channel marginally and jointly. In the marginal histograms, channels are quantized as powers of two, 2^n , for $n = 1, 2, \dots, 8$. Joint histograms are restricted to 2^n , for $n = 1, 2, 3, 4$, due the dramatic increase in dimensionality as n increases.

2.3.1 Distance Measures

Along with histogram parameters, there are also several measures used to compare the similarity of histograms. These include correlation, intersection, chi-square, earth mover's distance (EMD) and Bhattacharyya distance. Gray found that the Bhattacharyya distance, equation 2.3, was consistently the most reliable measure. However, his study makes no reference to the earth mover's distance.

$$d(H_1, H_2) = \sqrt{1 - \frac{\sum_i \sqrt{H_1(i)H_2(i)}}{\sqrt{\sum_i H_1(i) \sum_i H_2(i)}}} \quad (2.3)$$

The earth mover's distance [29] is based on the minimal cost to transform one distribution into the other. This has been shown to match perceptual similarity better than other distances used for image retrieval. Intuitively, given two distributions, one is seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the EMD is defined to be the least amount of work needed to fill the holes with earth. Here, a unit of work corresponds to transporting one unit of earth, one unit of distance. Computing the EMD is based on a solution to the well-known transportation problem and can be formalized as the following linear programming problem.

Let $P = \{f(p_1; w_{p_1}), \dots, (p_m; w_{p_m})\}$ be the first histogram with m bins, where p_i is the bin location and w_{p_i} is the weight of the bin. Let $Q = \{f(q_1; w_{q_1}), \dots, (q_n; w_{q_n})\}$ be the second histogram with n bins and $D = [d_{ij}]$, the distance matrix where d_{ij} is the distance between bins p_i and q_j . We wish to find the flow $F = [f_{ij}]$ that minimizes the cost

$$C(P, Q, F) = \sum_{i=1}^m \sum_{j=1}^n d_{ij} f_{ij} \quad (2.4)$$

subject to the following constraints:

$$\begin{aligned}
f_{ij} &\geq 0 & 1 \leq i \leq m, 1 \leq j \leq n \\
\sum_{j=1}^n f_{ij} &\leq w_{p_i} & 1 \leq i \leq m \\
\sum_{i=1}^m f_{ij} &\leq w_{q_j} & 1 \leq j \leq n \\
\sum_{i=1}^m \sum_{j=1}^n f_{ij} &= \min(\sum_{i=1}^m w_{p_i}, \sum_{j=1}^n w_{q_j})
\end{aligned}$$

The first constraint enforces unidirectional movement of the earth. The second constraint limits the amount of earth moved to the weight of a bin. The third constraint forces the maximum amount of earth moved to a bin to be equal to the bin's weight. Finally, the fourth constraint limits the maximum amount of earth that may be moved. The transportation problem can be solved using the Hungarian algorithm [30] and the EMD is given by normalized total flow:

$$EMD(P, Q) = \frac{\sum_{i=1}^m \sum_{j=1}^n d_{ij} f_{ij}}{\sum_{i=1}^m \sum_{j=1}^n f_{ij}} \quad (2.5)$$

One of the drawbacks of the earth mover's distance is the computational complexity $O(N^3 \log N)$. To reduce computation time when running experiments, the weighted wavelet transform [31] is computed. This is approximately as fast as computing the Euclidean distance for a one-dimensional histogram and a good approximation to the EMD.

2.3.2 Results

Our baseline results consider the previously mentioned RGB, HSV and YCbCr color spaces, both marginally and jointly, and both the Bhattacharyya distance and EMD. The results are presented over all parameters in tabular form in Table 2.3 and Table 2.4, but only the best results graphically in Figure 2.2.

In general, the HSV color space performed significantly better than the RGB and YCbCr color spaces. The average normalized area under the CMC curve using the

Table 2.3: Normalized area under the CMC curve for color histograms using the Bhattacharyya distance

Bins	Color Space					
	RGB		YCbCr		HSV	
	1D	3D	1D	3D	1D	3D
2	0.5494	0.6488	0.5933	0.6083	0.6952	0.6931
4	0.6327	0.7376	0.6037	0.6072	0.7803	0.7575
8	0.5832	0.7201	0.6357	0.6304	0.7769	0.7484
16	0.5895	0.7313	0.6806	0.6681	0.7736	0.7427
32	0.5887	—	0.6923	—	0.7633	—
64	0.5900	—	0.6927	—	0.7584	—
128	0.5934	—	0.6991	—	0.7585	—
256	0.5934	—	0.6968	—	0.7568	—
Average	0.5900	0.7094	0.6623	0.6285	0.7579	0.7354

Table 2.4: Normalized area under the CMC curve for color histograms using the EMD

Bins	Color Space					
	RGB		YCbCr		HSV	
	1D	3D	1D	3D	1D	3D
2	0.5522	0.5564	0.6041	0.6084	0.6638	0.5627
4	0.6483	0.6725	0.6040	0.6147	0.7519	0.7132
8	0.6304	0.6539	0.6086	0.6226	0.7662	0.7071
16	0.6295	0.6536	0.6158	0.6329	0.7705	0.6989
32	0.6274	—	0.6476	—	0.7725	—
64	0.6263	—	0.6504	—	0.7533	—
128	0.6257	—	0.6504	—	0.7508	—
256	0.6255	—	0.6494	—	0.7578	—
Average	0.6207	0.6341	0.6288	0.6196	0.7483	0.6705

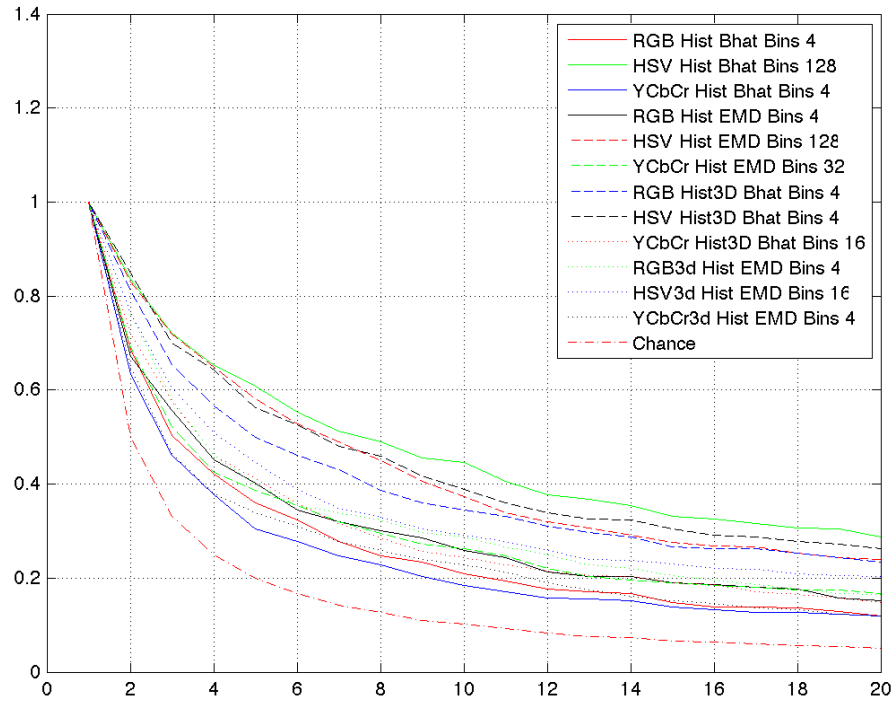
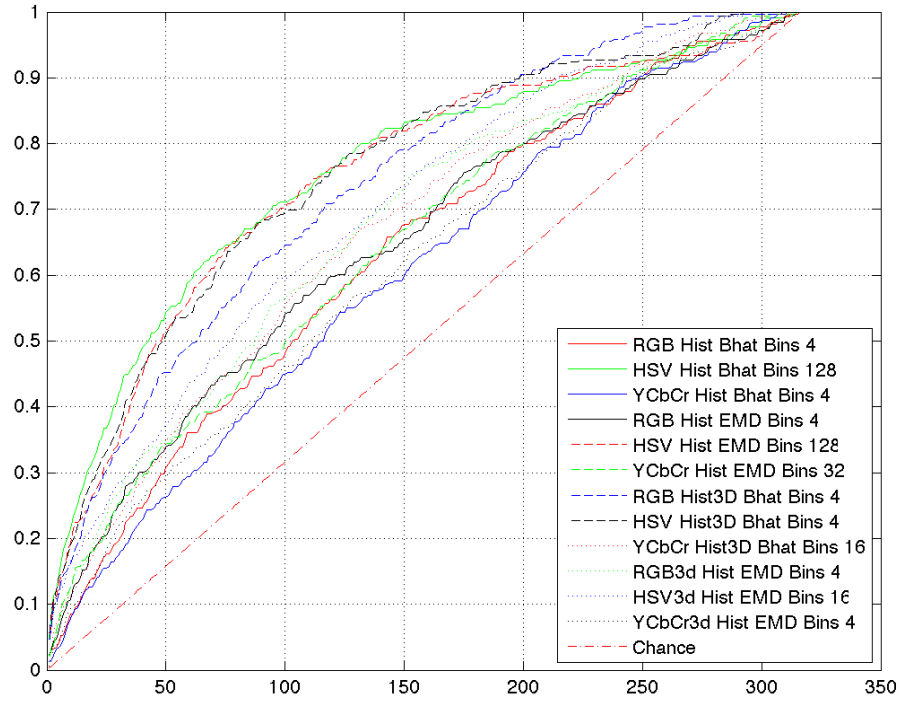


Figure 2.2: (Top): CMC curve for the highest performing histograms. (Bottom): SRR curve for the highest performing histograms.

Bhattacharyya distance is 0.6298, 0.6510 and 0.7504 for the RGB, YCbCr and HSV color spaces respectively. Similarly, when using the EMD, the average normalized CMC area is 0.6252, 0.6257 and 0.7224 for RGB, YCbCr and HSV respectively. Considering the channels jointly only improves performance in the RGB color space, but the improvement does not outperform the marginal HSV histograms. This is also advantageous for storage, as the size of a marginal histogram is $3n$ rather than n^3 , where n is the number of bins per channel. Generally the Bhattacharyya distance performs better than the EMD, the exception being marginal histograms in RGB space. It is also important to note that computing the EMD of joint histograms takes considerably longer than marginal histograms.

2.4 Regional Histogram Models

Although color histograms are invariant to rotation and scale, their complete lack of spatial information can lead to errors. For example, a person wearing blue pants and a black shirt will have a color histogram very similar to a person wearing black pants and a blue shirt. Thus to improve performance, we investigate the use of regional histograms models. One important note is that the dataset contains background, while a video surveillance system is likely to remove background using background modeling. Thus, performance would likely improve upon integration into a full system. In this study we consider two spatial models: hand-localized histograms and block histograms.

Hand localized histograms are a simple attempt to capture head, torso and legs as separate features. Ideally an algorithm would be capable of segmenting each of these regions, but the current state of the art does not support this. Instead, the fact that all of the images in the VIPeR dataset are scaled to a uniform height is exploited. The analysis used the top $\frac{1}{5}$, middle $\frac{2}{5}$ and bottom $\frac{2}{5}$ of the image to

represent the head, torso and legs.

Block histograms are created by partitioning the image into rectangular regions of equal size called blocks. A histogram is then computed for each block in the image. Several variations of block histograms can be achieved by varying block size, allowing block overlap and sub-blocks. For both hand-localized and block histograms the distance between any two images is found by summing the distances between the histograms of corresponding regions.

2.4.1 Hand Localized Histogram

For hand-localized histogram models, all of the same parameters as color histograms were considered. Once again, results are presented over all parameters in tabular form in Tables 2.5 and 2.6, but only the best results are graphically represented in Figure 2.3.

Many of the conclusions drawn from the color histogram results are confirmed in hand-localized histograms. The HSV color space performed significantly better than the RGB and YCbCr color spaces for both the Bhattacharyya distance and the EMD. The average normalized area under the CMC curve using the Bhattacharyya distance is 0.6556, 0.7281 and 0.7615 for the RGB, YCbCr and HSV color spaces respectively. Similarly, for the EMD, the average normalized CMC area is 0.6466, 0.7019 and 0.7650 for RGB, YCbCr and HSV respectively. Considering the channels jointly only improves performance in the RGB color space and generally the Bhattacharyya distance performs better than the EMD. However, the best results are found using HSV marginal histograms and the EMD.

There is significant improvement in overall performance with all parameters when using hand-localized histograms instead of color histograms. The most dramatic performance increase is in the YCbCr color space, while the HSV color space maintains the best performance.

Table 2.5: Normalized area under the CMC curve for hand-localized histograms using the Bhattacharyya distance

	Color Space					
	RGB		YCbCr		HSV	
Bins	1D	3D	1D	3D	1D	3D
2	0.5794	0.6764	0.6846	0.7101	0.7276	0.7059
4	0.6460	0.7588	0.6915	0.7117	0.7966	0.7728
8	0.5952	0.7384	0.7091	0.7127	0.7839	0.7655
16	0.6005	0.7561	0.7488	0.7210	0.7687	0.7583
32	0.5976	—	0.7641	—	0.7660	—
64	0.5986	—	0.7639	—	0.7624	—
128	0.6005	—	0.7617	—	0.7645	—
256	0.5999	—	0.7576	—	0.7661	—
Average	0.6022	0.7324	0.7351	0.7139	0.7670	0.7506

Table 2.6: Normalized area under the CMC curve for hand-localized histograms using the EMD

	Color Space					
	RGB		YCbCr		HSV	
Bins	1D	3D	1D	3D	1D	3D
2	0.5802	0.5864	0.6864	0.6905	0.6913	0.6917
4	0.6763	0.6952	0.6988	0.6869	0.7827	0.7564
8	0.6552	0.6672	0.7068	0.6728	0.7949	0.7580
16	0.6505	0.6646	0.7112	0.6654	0.7933	0.7102
32	0.6473	—	0.7220	—	0.7886	—
64	0.6461	—	0.7257	—	0.7754	—
128	0.6454	—	0.7274	—	0.7707	—
256	0.6452	—	0.7283	—	0.7803	—
Average	0.6433	0.6534	0.7133	0.6789	0.7722	0.7291

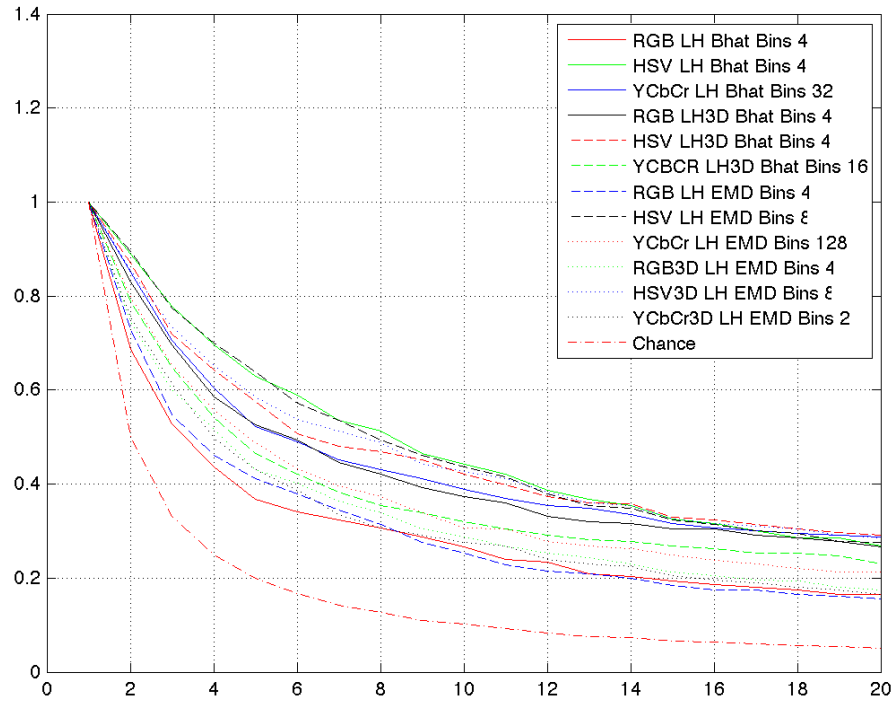
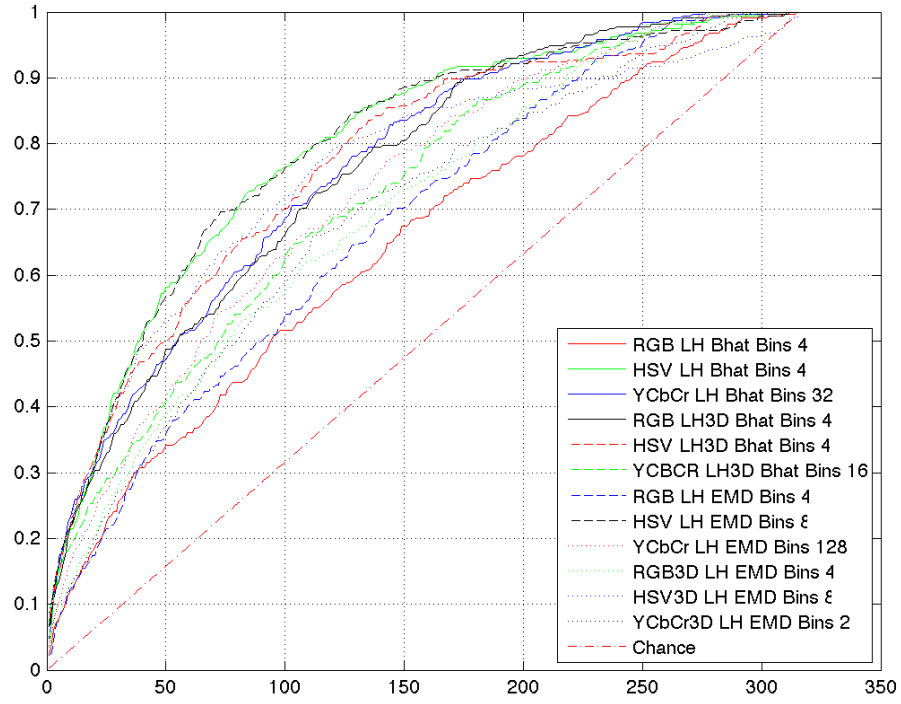


Figure 2.3: (Top): CMC curve for the highest performing local histograms. (Bottom): SRR curve for the highest performing local histograms.

2.4.2 Block Histograms

As mentioned previously, block histograms can exist in several forms by varying color space, block size, and block overlap. In our experiments, we consider the same color spaces RGB, YCbCr, and HSV; block sizes of 8×12 , 8×24 and 16×24 ; and overlapped and non-overlapped blocks. In cases where block overlap is used, a block overlap of 50% is used in both the x and y directions of the image. One drawback of block regions is the increased size of features and computation times. For this reason, histograms are constructed marginally over each channel and bins of 2, 4, 8, and 16 were considered.

Block histograms perform better than both color histograms and hand-localized histograms. Results using the Bhattacharyya distance in the RGB, YCbCr and HSV color spaces can be found in Tables 2.7, 2.8 and 2.9, while results using the EMD can be found in Tables 2.10, 2.11 and 2.12. In our experiments the 8×24 overlapping blocks maintain the best performance across all color spaces and distance metrics. In particular, the normalized area under the curve, 0.8398, in the HSV color space with overlapping blocks of size 8×24 and 4 bins, outperformed all baseline methods including the those presented by Gray. The tables indicate overlapping blocks improved performance in all situations. It can also be seen that EMD performance is better in the RGB color space while the Bhattacharyya distance is more robust in the YCbCr and HSV color spaces.

2.5 Large-Margin Nearest-Neighbor Distance Learning

2.5.1 Background

Distance metric learning for large-margin nearest-neighbor (LMNN) classification is a technique that learns a Mahalanobis distance metric for k-nearest neighbors

Table 2.7: Normalized area under the CMC curve for hand RGB block histograms using the Bhattacharyya distance

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.6366	0.6192	0.6354	0.6027	0.6265	0.5987
4	0.7132	0.6978	0.7173	0.6904	0.7073	0.6837
8	0.6781	0.6654	0.6791	0.6576	0.6678	0.6488
16	0.6830	0.6712	0.6821	0.6642	0.6712	0.6544
Average	0.6777	0.6634	0.6785	0.6537	0.6682	0.6463

Table 2.8: Normalized area under the CMC curve for hand RGB block histograms using the Bhattacharyya distance

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.8045	0.7867	0.8190	0.7553	0.8110	0.7424
4	0.8137	0.7946	0.8240	0.7662	0.8150	0.7554
8	0.8083	0.7905	0.8201	0.7619	0.8126	0.7534
16	0.8081	0.7904	0.8274	0.7776	0.8239	0.6544
Average	0.8086	0.7906	0.8226	0.7652	0.8156	0.7564

Table 2.9: Normalized area under the CMC curve for hand HSV block histograms using the Bhattacharyya distance

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.7640	0.7534	0.7728	0.7530	0.7682	0.7498
4	0.8310	0.8208	0.8398	0.8240	0.8363	0.8198
8	0.8329	0.8221	0.8368	0.8206	0.8319	0.8152
16	0.8299	0.8160	0.8344	0.8129	0.8285	0.8067
Average	0.8145	0.8031	0.8210	0.8026	0.8162	0.7979

Table 2.10: Normalized area under the CMC curve for hand RGB block histograms using the EMD

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.6239	0.5934	0.6353	0.5924	0.6208	0.5723
4	0.7263	0.7008	0.7399	0.7003	0.7170	0.6743
8	0.7157	0.6948	0.7285	0.6929	0.7121	0.6616
16	0.6952	0.6751	0.7022	0.6734	0.6914	0.6406
Average	0.6903	0.6660	0.7015	0.6648	0.6853	0.6372

Table 2.11: Normalized area under the CMC curve for hand YCbCr block histograms using the EMD

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.7918	0.7386	0.8066	0.7334	0.7791	0.7047
4	0.8161	0.7584	0.8319	0.7531	0.8022	0.7163
8	0.8151	0.7536	0.8299	0.7495	0.8003	0.7031
16	0.7978	0.7275	0.8148	0.7258	0.7803	0.6837
Average	0.8052	0.7445	0.8208	0.7405	0.7904	0.7019

Table 2.12: Normalized area under the CMC curve for hand HSV block histograms using the EMD

	Block Size					
	8×12		8×24		16×24	
Bins	Overlap	No Overlap	Overlap	No Overlap	Overlap	No Overlap
2	0.7640	0.7423	0.7676	0.7405	0.7565	0.7187
4	0.8165	0.7995	0.8211	0.7985	0.8086	0.7799
8	0.8162	0.8006	0.8223	0.8006	0.8114	0.7814
16	0.8196	0.7966	0.8245	0.7963	0.8101	0.7738
Average	0.8041	0.7847	0.8089	0.7804	0.7966	0.7635

classification. The technique was first introduced in 2006 by Weinberger et al. [32] and applied to face, spoken letter, and text recognition. The technique has since been applied to several computer vision problems and received 76 citations.

We start by defining the set (x_i, y_i) for $i = 1 \dots n$ as a training set where x_i and y_i denote the feature vector and class label of training sample i . n is the number of samples in the training set. The binary matrix y_{ij} defines the correspondence match between class labels. The goal of LMNN is to optimize kNN with respect to a linear transformation, T , with the following distance metric:

$$D(x_i, x_j) = \|T(x_i - x_j)\|^2 \quad (2.6)$$

Target neighbors are defined as the k nearest inputs with the same class label y_i . To improve classification, we want these target neighbors to have a minimal distance to x_i when computed using eq. 2.6. Let the binary matrix η_{ij} indicate whether x_j is a target neighbor of x_i . We can now define the cost function as

$$C(T) = \sum_{ij} \eta_{ij} \|T(x_i - x_j)\|^2 + c \sum_{ijl} \eta_{ij}(1 - y_{il}) [1 + \|T(x_i - x_j)\|^2 - \|T(x_i - x_l)\|^2]_+ \quad (2.7)$$

where in the second term $[z]_+ = \max(z, 0)$ denotes the standard hinge loss and $c > 0$ is a positive constant. The cost function has two competing terms. The first term, 2.8, is the pull term and penalizes large distances between features and its target neighbors. The second term, 2.9, is the push term and penalizes small distances between features and features from another class. The push term incorporates a margin by enforcing a hinge loss on differently labeled features that do not exceed the distance between a feature and its target neighbors by at least one unit distance. The concept is illustrated in Figure 2.4 from [32].

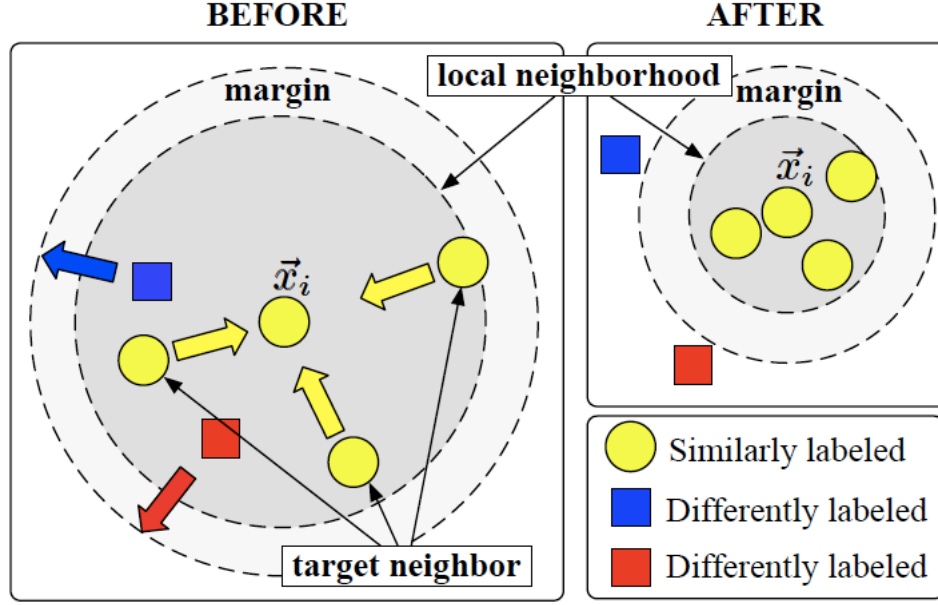


Figure 2.4: Illustration of LMNN for $k=3$

$$\sum_{ij} \eta_{ij} \|T(x_i - x_j)\|^2 \quad (2.8)$$

$$\sum_{ijl} \eta_{ij}(1 - y_{il}) [1 + \|T(x_i - x_j)\|^2 - \|T(x_i - x_l)\|^2]_+ \quad (2.9)$$

Optimization of the cost function can be solved by reformulation as semidefinite programming. Since semidefinite programming is convex, it is then possible to find the global minimum of the cost function. By defining the Mahalanobis distance, $M = T^T T$, the distance function can be reformulated as follows:

$$D(x_i, x_j) = (x_i, x_j)^T M(x_i, x_j) \quad (2.10)$$

The first term of the cost function is linear in M and the hinge loss can be written as slack variables ζ_{ijl} . The semidefinite programming problem can now be solved by minimizing equation 2.11 subject to the constraints 2.12, 2.13 and 2.14.

$$\sum_{ij} \eta_{ij}(x_i, x_j)^T M(x_i, x_j) + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \zeta_{ijl} \quad (2.11)$$

$$(x_i, x_j)^T M(x_i, x_j) - (x_i, x_j)^T M(x_i, x_j) \geq 1 - \zeta_{ijl} \quad (2.12)$$

$$\zeta_{ijl} \geq 0 \quad (2.13)$$

$$M \geq 0 \quad (2.14)$$

2.5.2 Experiments and Results

In order to learn the Mahalanobis distance, M , one must first define a feature to represent pedestrians in an arbitrary view. The results from the previous sections were used to select these models. It is clear from the region based histograms that spatial information is vital to an appearance model. In our experiments, a feature was created by concatenating the block based regional histograms. In order to reduce the dimensionality of the feature, principal component analysis (PCA) was performed on all training samples. Our experiments utilized a block size of 8×24 in the YCbCr and HSV color spaces and bin counts of 4 and 16 respectively. These two parameter sets had the best performance of the simple appearance models. Features were also concatenated from both color spaces. The dimension was reduced to 20, 30, and 40 for each method. MATLAB experiments were conducted using code provided by Kilan Weinberger at <http://www.cse.wustl.edu/~kilian/kqw/Code.html>. Distance measure learning took less than five minutes on the training set and the low dimensionality of the features allowed for nearly instantaneous distance computations between appearance models. The results can be found in tabular form in Table 2.13 and graphically in Figure 2.5.

The result of LMNN distance learning outperformed the current state of the art, ELF using AdaBoost as reported by Gray and Tao [27]. Notably, there was over a

Table 2.13: Normalized area under the CMC curve using LMNN

Dimension	HSV	YCbCr	HSV/YCbCr
20	0.9306	0.9084	0.9458
30	0.9409	0.9137	0.9526
40	0.9441	0.9085	0.9547

10% performance increase for matches with rank 50 or less. The results indicated the HSV and HSV-YCbCr feature performance increased with dimensionality. However, the YCbCr features yielded the best performance when the dimensionality is reduced to 30. When the features from both the HSV and YCbCr color space were combined, the performance increased for any dimensionality.

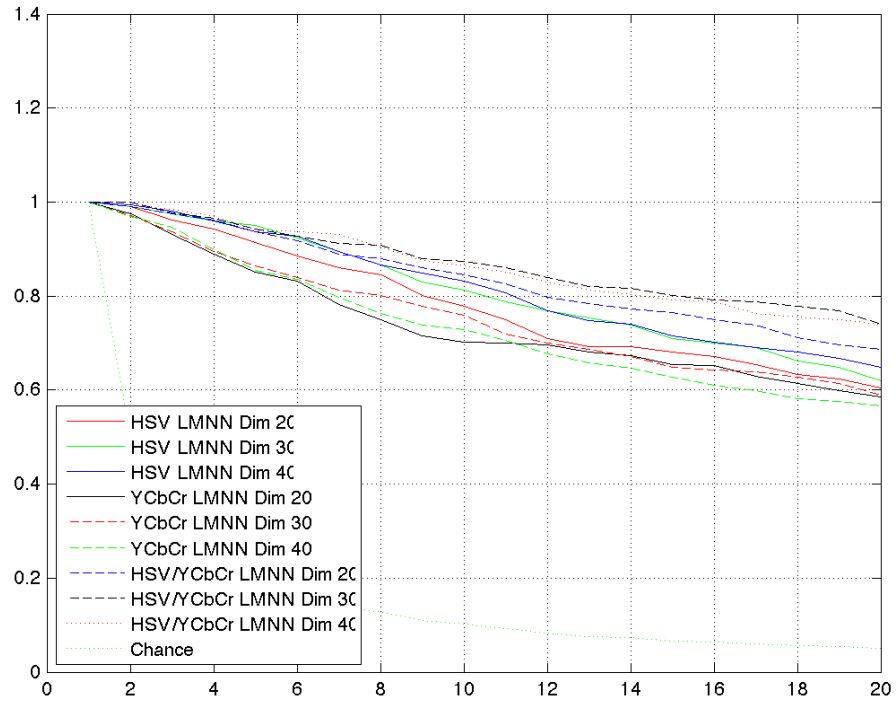
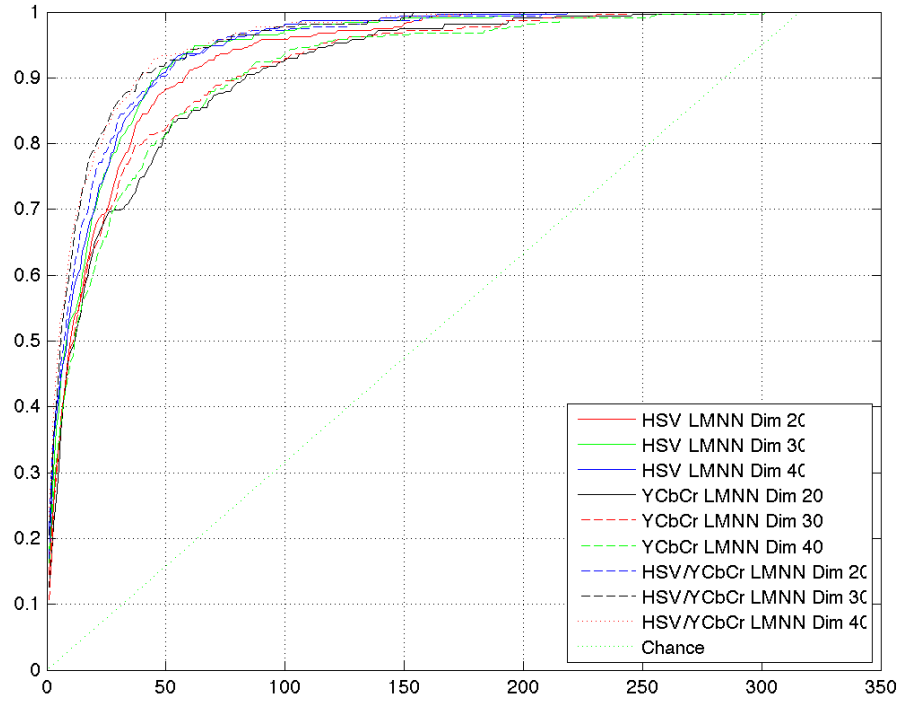


Figure 2.5: (Top): CMC curve for the highest performing histograms. (Bottom): SRR curve for the highest performing histograms.

CHAPTER 3

IFP IMAGE PROCESSOR

Although the focus of this thesis is to evaluate and design appearance models for pedestrian surveillance, there is no known framework to test these models in real scenarios. The IFP Image Processor is a cross-platform GUI application built using the wxWidgets [33] library. The IFP Image Processor was designed to solve the problem of tracking pedestrians across cameras with non-overlapping fields of view, but can be used to develop algorithms for nearly any surveillance system. The IFP Image Processor also utilized the OpenCV library [34]. Any OpenCV-based algorithm can be integrated into the IFP Image Processor. The basic structure of the IFP Image Processor can be found in Figure 3.1.

3.1 Fusion Center

The IFP Image Processor’s main module is the Fusion Center. The Fusion Center contains multiple Image Processors, each of which is responsible for extracting information from a video source. The Fusion Center uses information from multiple sensors to drive algorithms such as cross-camera association, activity recognition and scene understanding. It is also possible for the Fusion Center to contain a single Image Processor and extract information from a single field of view. For the problem of cross-camera tracking, the Fusion Center holds the track parameters and appearance features for the targets in each sensor’s field of view. When a target dismount occurs, the track information is stored in a possible target pool. These

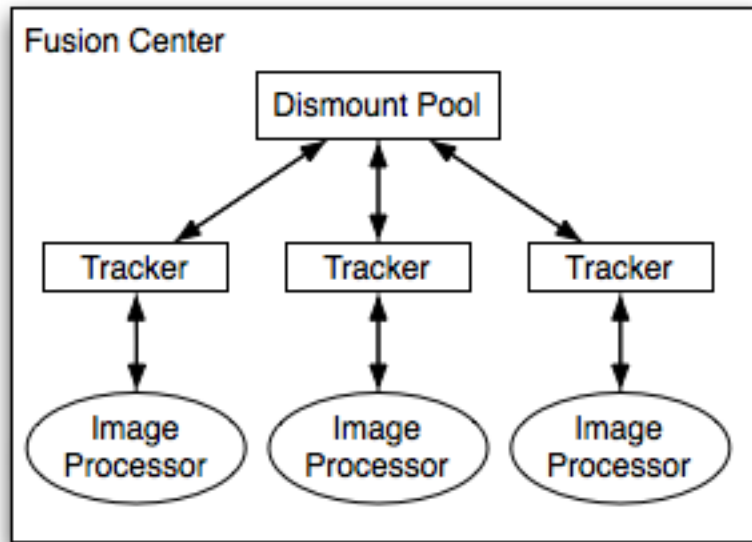


Figure 3.1: Structure of the Fusion Center.

possible targets are then matched with new tracks that appear in other fields of view. The Fusion Center also controls all Image Processors. When an update in the Fusion Center occurs, it cycles through each Image Processor and updates the information associated with each tracker.

3.1.1 Tracking

The Fusion Center uses a Kalman filter to track people in the image plane. The Kalman filter implements a predictor-corrector type estimator that is optimal in the sense that it minimizes the estimated state error covariance. The Kalman filter relies on the underlying dynamics of the state and observations given by two equations:

$$\text{State:} \quad \mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k + w \text{ for } k \geq 0$$

$$\text{Observation:} \quad \mathbf{y}_k = \mathbf{H}^T \mathbf{x}_k + v_k \text{ for } k \geq 0 \quad (3.1)$$

$$\text{Where} \quad w \sim N(0, \mathbf{Q}) \text{ and } v \sim N(0, \mathbf{R})$$

People are tracked using a nearly constant velocity dynamics model. This assumes that people walk at a nearly constant velocity with random perturbations in acceleration. The state vector \mathbf{x} contains the target's position and velocity in pixel space. The transition matrix \mathbf{F} and the noise covariance matrix \mathbf{Q} are generated from the well known kinematic equations $s = vt + \frac{1}{2}at^2$ and $v_{i+1} = v_i + at$.

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix} \quad \mathbf{F} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

$$\mathbf{Q} = \begin{bmatrix} \frac{T^4}{4} & 0 & \frac{T^3}{2} & 0 \\ 0 & \frac{T^4}{4} & 0 & \frac{T^3}{2} \\ \frac{T^3}{2} & 0 & T^2 & 0 \\ 0 & \frac{T^3}{2} & 0 & T^2 \end{bmatrix} \quad (3.3)$$

Note that T can be selected based on the frame rate of the camera and known limitations of a target, or it is possible to simply track in units of pixels per frame. The latter is currently implemented in the IFP Image Processor. The measurements used in image-plane tracking are the target location in the current frame in pixel coordinates. The observation matrix, \mathbf{H} , and measurement noise matrix, \mathbf{R} , are realized as follows:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (3.4)$$

3.1.2 Association Across Non-Overlapping Fields of View

When the Fusion Center updates, each Image Processor returns a set of detection locations and corresponding features. Detections are associated with tracks using the Mahalanobis distance. The appearance feature is selected by the user and corresponds to one of the methods tested in the previous chapter.

When a track leaves one field of view, it is placed in the possible target pool with an ID corresponding to its previous Image Processor. This track is compared to new tracks created in individual trackers. If a new track is a possible match, based on location and temporal information, the distance between the appearance features of the possible target and the new target is computed. A match can be computed by either thresholding the distance or associating the most likely match with the new track. The latter would be used in cases where known environment constraints force a new target to come from the possible target pool.

3.2 Image Processor

The Image Processor implements algorithms used to extract information from a single sensor. The Image Processor currently extracts features by learning a background model, detecting pedestrians, and extracting their features. The basic structure, as currently implemented, can be found in Figure 3.2.

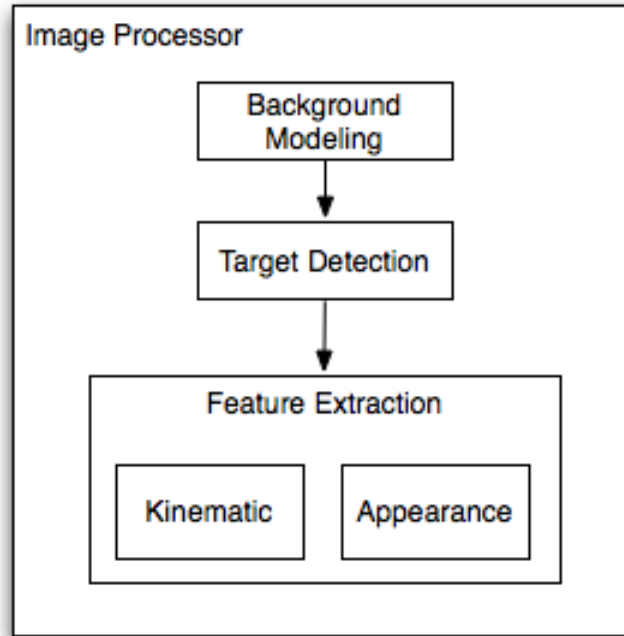


Figure 3.2: Structure of the Image Processor.

3.2.1 Background Modeling

Background modeling is used to segment an image into background and foreground components. The IFP Image Processor uses the Gaussian Mixture Model (GMM) background subtraction algorithm introduced by Zivkovic [35, 36]. The algorithm is a pixel-level background subtraction that learns a probability density function for each pixel individually. A pixel from a new image is assigned a background pixel if its intensity is well described by its density function.

Zivkovic’s algorithm assumes the probability density function for each pixel can be modeled as a GMM. The GMM is learned using the pixels from the previous T frames and adaptively adjusts the number of modes in the distribution. The squared Mahalanobis distance is used to determine whether a new pixel is well defined by the GMM. An example of the background subtracted image can be found in Figure 3.3.

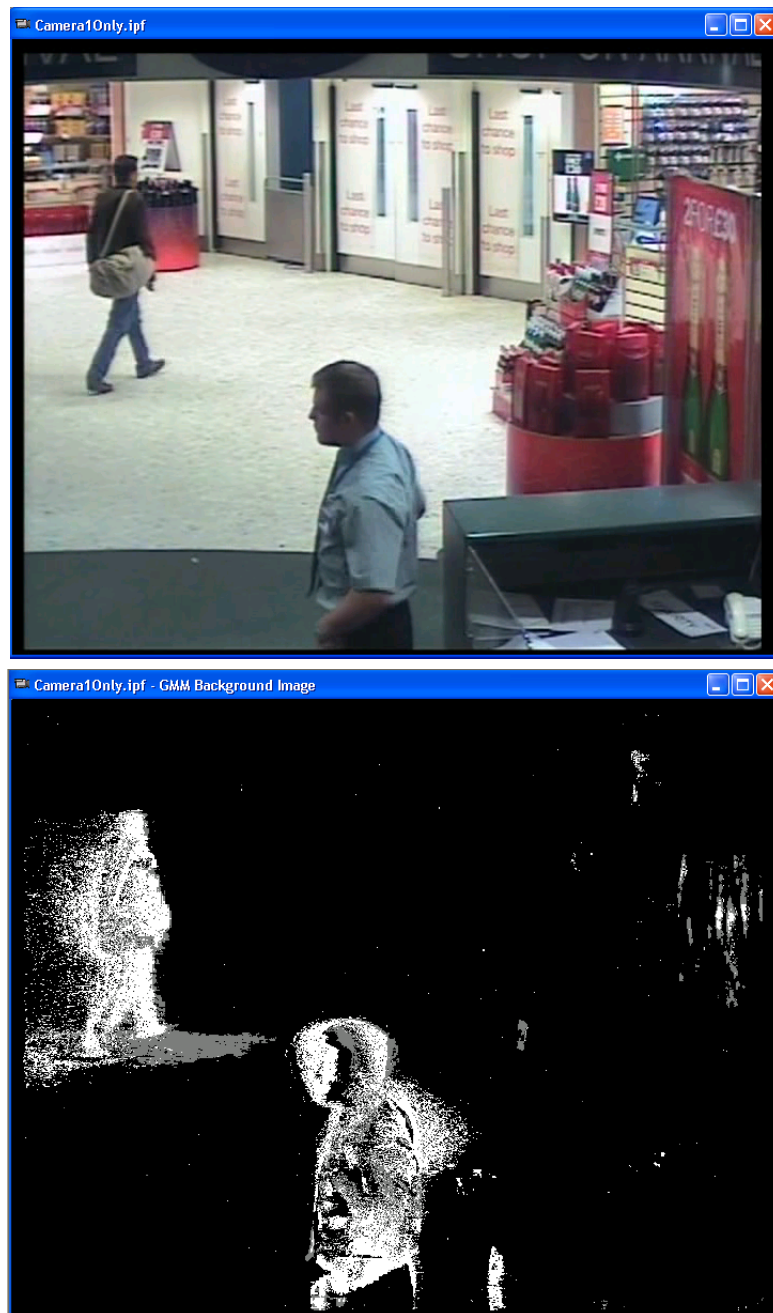


Figure 3.3: Original frame (Top) and corresponding GMM background subtracted image (Bottom).

3.2.2 Detection and Feature Extraction

The detection algorithm is used to cluster foreground pixels from the background subtraction image into detections that represent objects in the image. The IFP Image Processor cascades a median filter, morphological closing operation and maximum separation clustering to detect multiple object regions.

From Figure 3.3, we see that the background image contains salt and pepper noise. The median filter removes this noise. A second source of error is caused by uniform object color. When an object of uniform color moves slowly across a pixel, the probability density function of that color value will increase. This causes foreground detections to be more likely along the edges of objects. The morphological closing operation is used to close holes caused by these errors. Examples of both images can be found in Figure 3.4.

The morphologically closed image is used as the basis for creating detections. A detection is a cluster of foreground pixels, in which every pixel is separated by a distance less than D_s from a pixel in the same cluster. Once the clusters are created, the total number of pixels in each cluster is compared to a minimum and maximum cluster size. If the pixel count is within range, the cluster is considered a detection. An example detection can be found in Figure 3.5.

In order to track images in a single field of view and across cameras, kinematic and appearance features are extracted from each detection. The Kalman filter, previously discussed, requires a measurement of the target location in the image. The location of a target is found by calculating the centroid of the detection. The appearance model is created by computing features from pixels only detected as foreground or by creating a bounding box around the detection and creating features from the bounding box region. The dataset used to test the models in the previous chapter contains pairs of detections based on a bounding box. Thus, the latter approach is used in IFP Image Processor.



Figure 3.4: Median filtered image (Top) and morphologically closed image corresponding to GMM image in Figure 3.3.

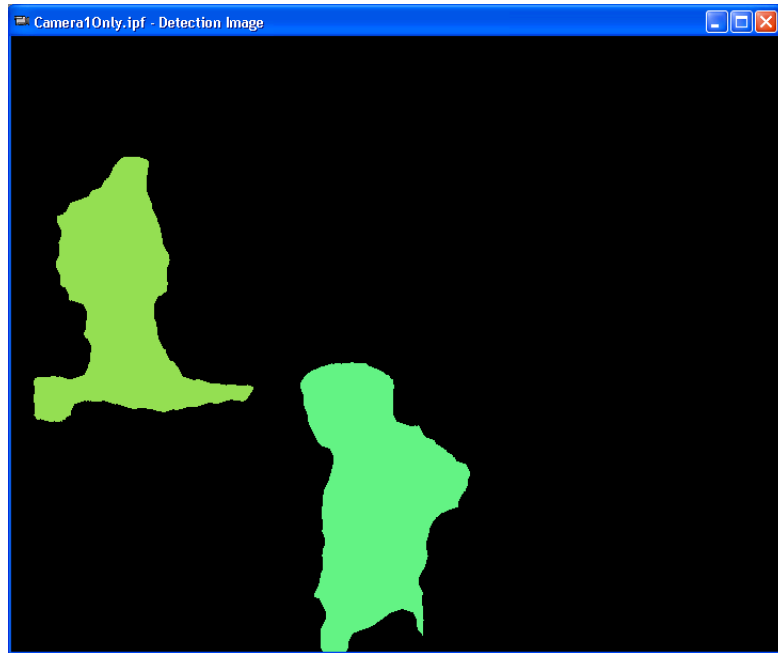


Figure 3.5: Detection image corresponding to frame in Figure 3.3.

3.3 Interface

The IFP Image Processor is designed with a simple user interface to change surveillance system parameters. This section briefly discusses how to take advantage of the IFP Image Processor user interface.

3.3.1 Creating a Case

To create a new case, select **File | New**. Then select **Image Processor Files** from the dialog box. At this point, a new Fusion Center and a single Image Processor have been created. The newly created window is blank because no video source has been assigned to the Image Processor. To edit a case select **Edit | Case . . .**. The dialog box in Figure 3.6 will appear.

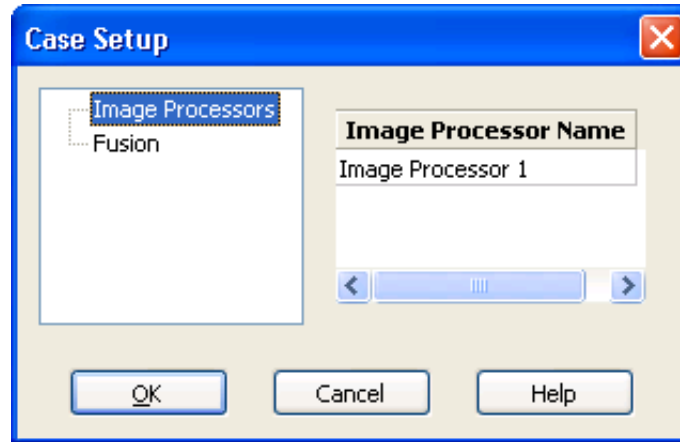


Figure 3.6: IFP Image Processor Case Setup dialog box.

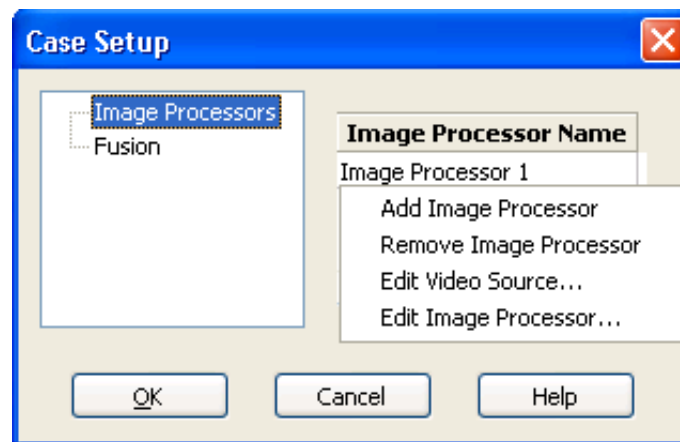


Figure 3.7: Adding Image Processors.

3.3.2 Editing Fusion Center

The IFP Image Processor initially has a single Image Processor. To add an Image Processor, right-click on the grid control and select **Add Image Processor**. To remove an Image Processor, right-click on the row in the grid control that corresponds to the Image Processor, and select **Remove Image Processor** as shown in Figure 3.7.

To select the appearance model used in the Fusion Center, click on **Fusion** in the tree control on the left side of the Case Setup dialog box. Select the preferred model from the panel as show in Figure 3.8.

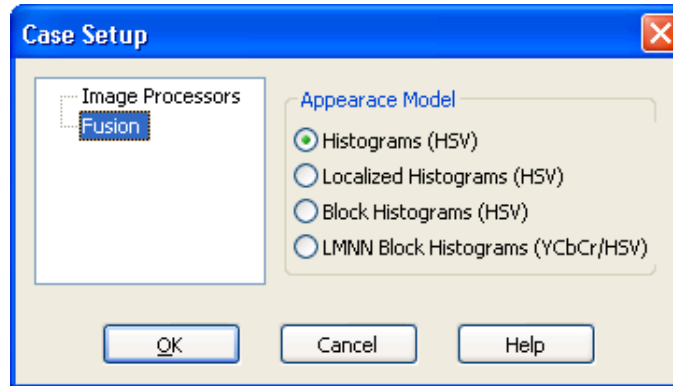


Figure 3.8: Selecting an appearance model in the Fusion Center panel.

3.3.3 Editing Image Processor

Each Image Processor is assigned a video source. The video source can be a sequence of files, an IP camera, or an AVI file. To set the video source for an Image Processor, right-click on the Image Processor you wish to edit and select **Edit Video Source...** as shown in Figure 3.7. In the Image Source Parameters dialog box, select the appropriate source type (**Sequence of Files**, **IP Camera**, or **AVI File**), and the corresponding parameters as shown in Figure 3.9.

To edit an Image Processor, right-click on the row in the grid control that corresponds to the Image Processor, and select **Edit Image Processor...**. The Image Processor Properties dialog box in Figure 3.10 will appear. This dialog box holds panels that allow the user to edit the Image Processor's configuration, GMM background model, target detection and Kalman filter parameters. The configuration parameters allow the user to select the level of processing (only display video, GMM background modeling, smoothing, morphological filtering, target detection, Kalman filtering) that is performed by the Image Processor.

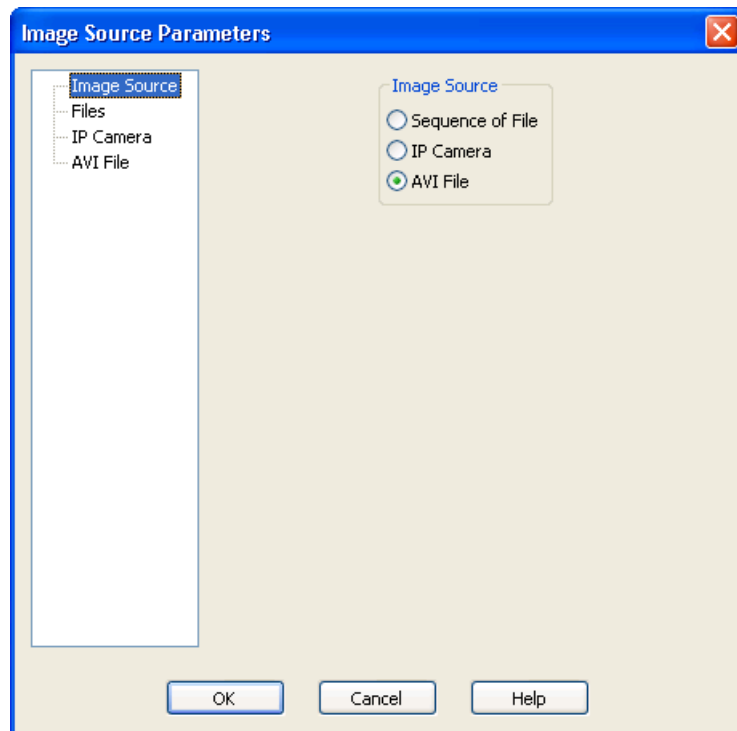


Figure 3.9: Changing the video source of an Image Processor.

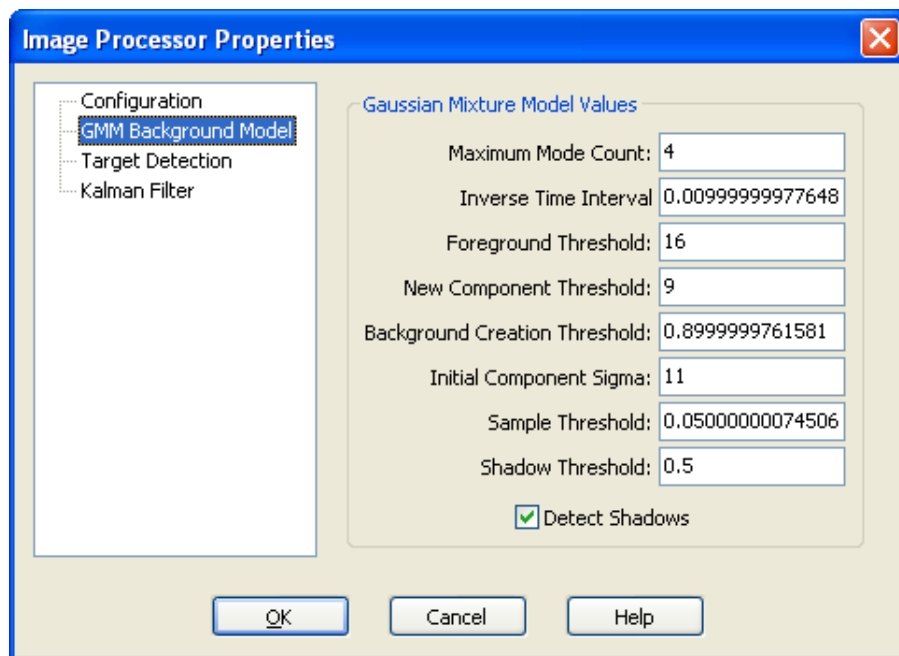


Figure 3.10: Image Processor Properties dialog box.

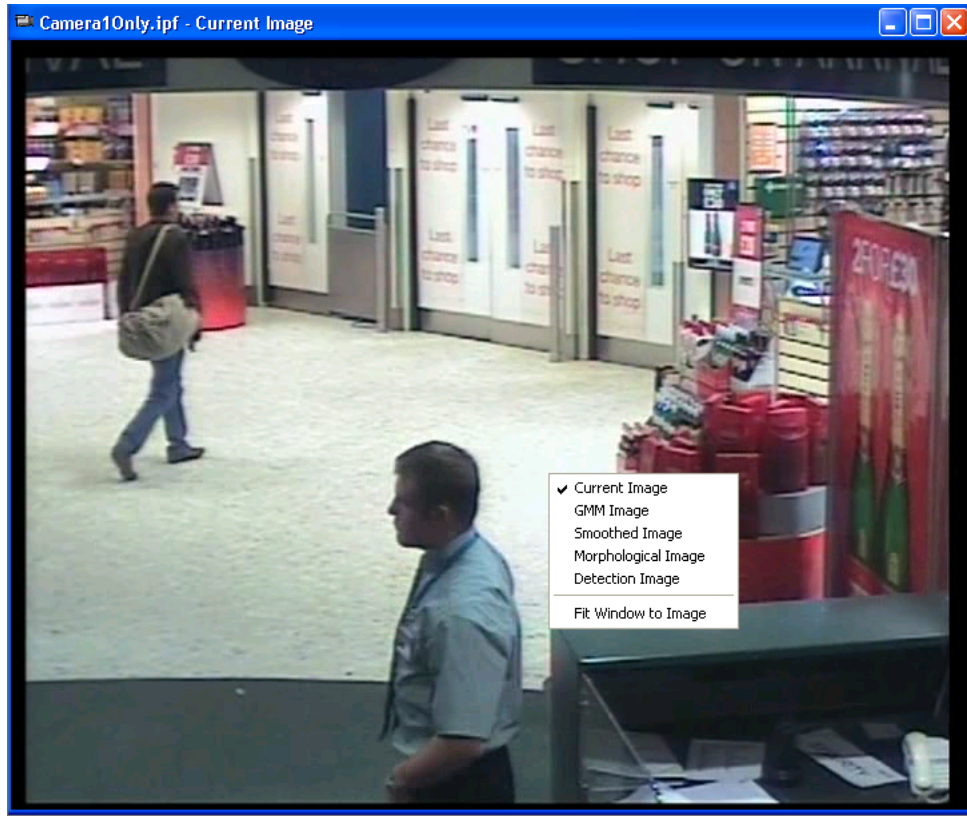


Figure 3.11: Changing the current displayed image type.

3.3.4 Display

It is possible to change the displayed image type, `Current Image`, `GMM Image`, `Smoothed Image`, `Morphological Image`, `Detection Image`, by right-clicking on the image window and selecting the appropriate image. This is illustrated in Figure 3.11. An additional menu entry, `Fit Window to Image`, can be utilized to resize the client area of the image window to match the image size. Additional views can be displayed by selecting the `View | Add View` menu entry. This can be repeated to display multiple views, with each view displaying a different image type. Each window is updated with the specified image type as the application processes each frame.

REFERENCES

- [1] L.-Q. Xu, J. L. Landabaso, and B. Lei, "Segmentation and tracking of multiple moving objects for intelligent video analysis," *BT Technology Journal*, vol. 22, no. 3, pp. 140–150, 2004.
- [2] G. Foresti, C. Micheloni, L. Snidary, P. Remagnino, and T. Ellis, "Active video-based surveillance systems," *IEEE Signal Processing Magazine*, vol. 22, pp. 25–37, 2005.
- [3] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 34, pp. 334–352, 2004.
- [4] T. Huang and S. Russell, "Object identification: A Bayesian analysis with application to traffic surveillance," *Artificial Intelligence*, vol. 103, no. 1-2, pp. 77–93, 1998.
- [5] Y. Loke, P. Kumar, S. Ranganath, and W. Huang, "Object matching across multiple non-overlapping fields of view using fuzzy logic," *Acta Automatica Sinica*, vol. 32, no. 6, pp. 978–987, 2006.
- [6] F. L. Lim, W. Leoputra, and T. Tan, "Non-overlapping distributed tracking system utilizing particle filter," *Journal of VLSI Signal Processing Systems*, vol. 49, no. 3, pp. 343–362, 2007.
- [7] R. Pflugfelder and H. Bischof, "Tracking across non-overlapping views via geometry," in *IEEE International Conference on Pattern Recognition*, 2008, pp. 1–4.
- [8] Z. Islam, C. Oh, and C. Lee, "New integrated framework for video based moving object tracking," in *International Conference on Human-Computer Interaction*, 2009, pp. 423–432.
- [9] Y. Guo, S. Hsu, Y. Shan, H. Sawhney, and R. Kumar, "Vehicle fingerprinting for reacquisition and tracking in videos," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, pp. 761–768.
- [10] M. Black and A. Jepson, "EigenTracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

- [11] H. Murase and S. Nayar, “Visual learning and recognition of 3-D objects from appearance,” *International Journal of Computer Vision*, vol. 14, no. 1, pp. 5–24, 1995.
- [12] X. Wang, G. Doretto, T. Sebastian, J. Rittscher, and P. Tu, “Shape and appearance context modeling,” in *IEEE International Conference on Computer Vision*, 2007, pp. 1–8.
- [13] N. Gheissari, T. Sebastian, and R. Hartley, “Person reidentification using spatiotemporal appearance,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 1528–1535.
- [14] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [15] A. Alahi, P. Vandergheynst, M. Bierlaire, and M. Kunt, “Cascade of descriptors to detect and track objects across any network of cameras,” *Computer Vision and Image Understanding*, to be published.
- [16] R. Fergus, P. Perona, and A. Zisserman, “Object class recognition by unsupervised scale-invariant learning,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2003, pp. 264–271.
- [17] Y. Yu, D. Harwood, K. Yoon, and L. Davis, “Human appearance modeling for matching across video sequences,” *Machine Vision Applications*, vol. 18, no. 3, pp. 139–149, 2007.
- [18] P. Fieguth and D. Terzopoulos, “Color-based tracking of heads and other mobile objects at video frame rates,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 21–27.
- [19] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–575, 2003.
- [20] J. Huang, K. Ravi, M. Mitra, W. Zhu, and R. Zabih, “Spatial color indexing and applications,” *International Journal on Computer Vision*, vol. 35, no. 3, pp. 245–268, 1999.
- [21] S. Birchfield and S. Rangarajan, “SpatioGrams versus histograms for region-based tracking,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, pp. 1158–1163.
- [22] O. Javed, K. Shafique, and M. Shah, “Appearance modeling for tracking in multiple non-overlapping cameras,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2005, pp. 26–33.

- [23] Hordley S. and Finlayson G. and Schaefer G. and Yun Tian G., “Illuminant and device invariant colour using histogram equalisation,” *Pattern Recognition*, vol. 38, no. 2, pp. 179–190, 2005.
- [24] G. Buchsbaum, “A spatial processor model for object colour perception,” *Journal of the Franklin Institute*, vol. 310, no. 1, pp. 1–26, 1980.
- [25] D. Truong Cong, C. Achard, L. Khoudour, and L. Douadi, “Video sequences association for people re-identification across multiple non-overlapping cameras,” in *International Conference on Image Analysis and Processing*, 2009, pp. 179–189.
- [26] D. Gray and H. Tao, “Evaluating appearance models for recognition, reacquisition, and tracking,” in *IEEE International Workshop on Performance Evaluation for Tracking and Surveillance*, 2007, pp. 41–47.
- [27] D. Gray and H. Tao, “Viewpoint invariant pedestrian recognition with an ensemble of localized features,” in *European Conference on Computer Vision*, 2008, pp. 262–275.
- [28] C. Stauffer and E. Grimson, “Similarity templates for detection and recognition,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, pp. 221–228.
- [29] Y. Rubner, C. Tomasi, and L. Guibas, “The earth mover’s distance as a metric for image retrieval,” *International Journal on Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.
- [30] J. Munkres, “Algorithms for the assignment and transportation problems,” *Journal of the Society of Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.
- [31] A. Andoni, P. Indyk, and R. Krauthgamer, “Earth mover distance over high-dimensional spaces,” in *ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2008, pp. 343–352.
- [32] K. Weinberger, J. Blitzer, and L. Saul, “Distance metric learning for large margin nearest neighbor classification,” in *Advances in Neural Information Processing Systems*, vol. 18. MIT Press, 2006, pp. 1473–1480.
- [33] WxWidgets, “Cross-platform GUI library ,” 2009. [Online]. Available: <http://www.wxwidgets.org>.
- [34] OpenCv, “Opencv — library of programming functions for real time computer vision,” 2009. [Online]. Available: <http://opencv.willowgarage.com/wiki/>.
- [35] Z. Zivkovic, “Improved adaptive gaussian mixture model for background subtraction,” in *International Conference on Pattern Recognition*, vol. 2, 2004, pp. 28–31.

- [36] Z. Zivkovic and F. van der Heijden, “Efficient adaptive density estimation per image pixel for the task of background subtraction,” *Pattern Recognition Letters*, vol. 27, no. 7, pp. 773–780, 2006.