Tracking and Activity Analysis in Retail Environments Technical Report 620

Alex Leykin
Indiana University, Bloomington, IN USA
oleykin@indiana.edu

Mihran Tuceryan
Indiana University Purdue University, Indianapolis, IN USA
tuceryan@iupui.edu

October 12, 2005

Abstract

In this paper we present a system that tracks customers in a store and performs a number of activity analysis tasks based on the output from the tracker. We obtain the trajectories by employing a human body tracking system designed as a Bayesian jump-diffusion filter. The customer travel trajectories on the floor map are extracted and post processed to remove noise. The shoppers that belong to the same group are identified by clustering their trajectories. The clustering is based on a distance metric that incorporates both time and location information. Our system also identifies "shopper groups" based on the proximity metric also presented in this paper. Further, store employees are detected as a separate group, based on a 2D color histogram analysis. Finally, "dwelling customers", i.e the customers stopping to browse for products are detected by analyzing the behavior of the recorded trajectories.

Keywords: Tracking and Motion, Crowded Environments, Human Activity Modeling, Background Subtraction, Camera Calibration

Contents

1 Introduction

As marketing researchers in academia and industry seek for tools to aid their decision making, the interest is more and more coming across computer vision and in particular human tracking systems. Unlike other types of sensors, vision presents an ability to observe customer experience without separating it from the environment. By tracking the path traveled by the customer along the store, important pieces of information, such as customer dwell time, isle penetration and product interaction statistics can be collected [1]. In our work we have concentrated on extracting from video, perhaps, one of the most important customer statistics: information about the *shopper groups*.

Our system segments foreground regions out of each frame by using a dynamically adapting background model presented here. Because each foreground region may contain multiple people, we further hypothesize about the number of human bodies within each such region by using the head-candidate selection algorithm. The head is chosen as the most distinguishable and pronounced part of the human body, especially when observing the store with a highly elevated monocular camera. As the next step, our system constructs a Bayesian inference model, based on the a priori knowledge of the human parameters and store layout and geometry. Observations of the body appearances at each frame are a second driving force in our probabilistic scheme. Tracked in this manner, the individual path of each customer superimposed on the floor plan of the store is recorded and can be further analyzed. To identify the customers which are shopping as a group we have designed a distance metric measured on the traveled trajectories. This metric, incorporating space and time deviations between two paths we then use in a clustering system, which labels shopper groups in the input video. We further perform histogram analysis to detect store employees and motion dynamics analysis to detect dwelling customers.

1.1 Related Work

In videos taken with a stationary camera, background subtraction is a primary technique used to segment out foreground pixels. Statistical background modeling based on color distortion has been presented in [2], but a single mean for each pixel is unlikely to account for the noisiness of the background in the changing environment of the store. We have also focused on the methods that use a mixture of Gaussians to model each pixel [3]. These methods are superior to the single-modality approaches, yet they operate on the fixed number of modalities which fails to comprehensively accommodate the noise and artifacts created by video compression algorithms, such as MPEG. We have developed an adaptive background model based on the dynamic codebook approach which compensates for these problems.

To create the initial estimates for any tracking algorithm, some form of head position estimation has been used in related studies. In [4, 5] the vertical projection histogram was computed to reliably establish the location of head-candidates. Although the aforementioned approach shows promising results

with the horizontally looking camera, in this paper we make an argument that it will be prone to significant distortion in the case of ceiling mounted camera if the camera extrinsic parameters are not accounted for. As a result we are using the projection histogram that accounts for the camera and 3D scene parameters.

Significant progress has been made in detection and tracking of people. The majority of the studies address tracking of isolated people in well controlled environment, however there is an increasing effort in tracking specifically in *crowded environments* [6, 7, 4, 8, 9, 10]. It is worth noting that many works assume the luxury of multiple well-positioned cameras or stereo vision, which are to a certain extent not present in most retail establishments and/or do not have the desired overlapping fields of view. In contrast, cheap low-resolution digital monocular color cameras are becoming more and more readily available as well as the hardware for capturing compressed real-time streams provided by these cameras.

The Bayesian modeling approach applied to human tracking has demonstrated potential in resolving ambiguities while dealing with the crowded environments [11, 5]. Working under the Bayesian framework it has been shown that particle filters can efficiently infer both the number of objects and their parameters. Another advantage is that in dealing with distributions of mostly unknown nature, particle filters do not make Gaussianity assumptions, unlike Kalman filters [12, 13].

Tracking followed by analysis of customer behavior in stores is becoming an increasingly active subject in computer vision publications [4, 8, 9, 5]. The novelty of this paper is that it brings the marketing applications perspective as well as implements one such application - detection and tracking of shopper groups based on the paths traveled by customers. Several approaches based on DFT and Dynamic Time Warping exist for comparing time series. Most recently a *longest common subsequence* based method for comparing trajectories has been implemented by Buzan et al. in [14]. The specificity of our task is that it requires a relative time component — as opposed to comparing just the shapes of the trajectories, yet is must not account for time warps — as it is done in speech recognition.

2 Method Overview

The methods presented in this paper are aimed at obtaining a tool for retailers to analyze patterns of behaviors by shoppers in stores and using the results of this analysis to make various marketing decisions. The system consists of layers ranging from low-level image processing operations, to tracking the positions of individuals in the store videos, to the higher level analysis of their movements and activities in the store. Figure 2 shows the various layers of the system and the individual modules that make up these layers. The first layer consists of four major steps (figure 1): (i) background modeling and subtraction, (ii) camera modeling, (iii) head candidates detection, and (iv) human height measurement. The output from the background subtraction is the binary foreground map as

well as an array of foreground blobs, each represented as a 2D contour. Camera calibration provides next stage of the system with the location of vanishing points $\mathbf{V}_X$, $\mathbf{V}_Y$ and $\mathbf{V}_Z$ as well as the scale factor.

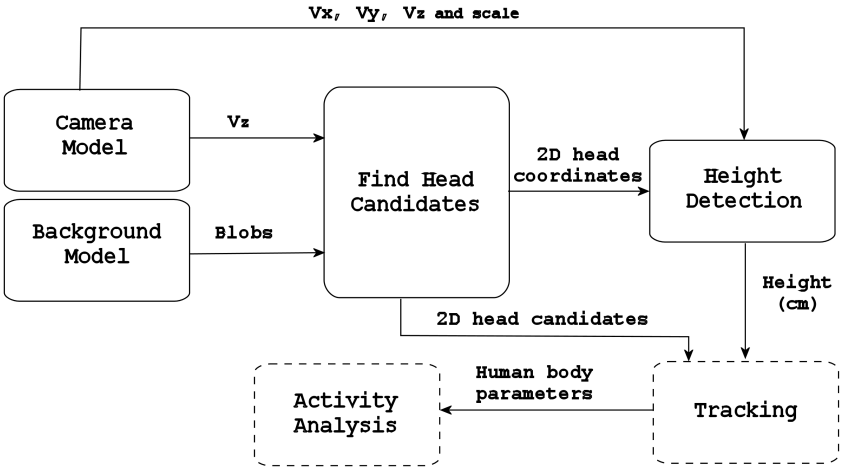

Figure 1: Low-level processing components of the system

This is accomplished by a background subtraction scheme which uses knowledge about the background model which is learned from the scene. In addition, knowledge about the camera model and the rough shape of shoppers is also used to map the position of the tracked person on a floor map of the store and to obtain rough estimates of the positions of their heads in order to resolve multiple shoppers possibly occluding each other. The details of this layer are described in Section 3. The next layer uses a Bayesian particle-filter model to track the segmented individuals and also keep track of their identities (e.g., when shoppers cross each other). It dynamically assigns identities when new shoppers enter the scene. The results of this analysis is also fed back to the background subtraction layer. The details of this layer are described in Section 4. Finally, the third layer uses the tracking results from the first two layers to analyze the behavior of shoppers. As part of this analysis, the paths of the shoppers on the store floor are extracted. This information is then used to determine if some of the shoppers belong to the same group based on their pattern of movements in the store. The details of this layer are described in Section 5.

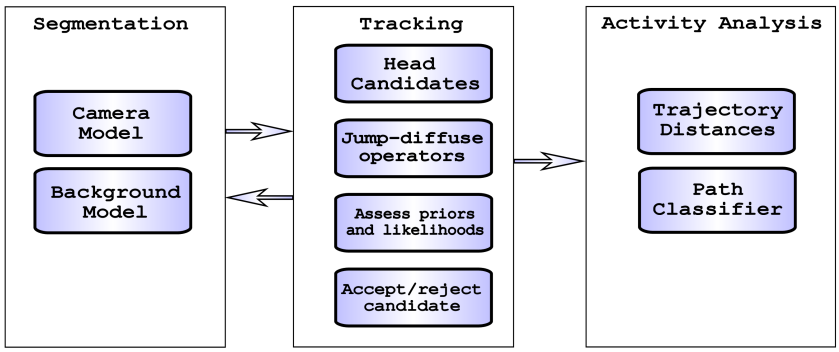

Figure 2: Major components of our system

3 Background and Camera Modeling

3.1 Camera Model

While building realistic human body models during the higher-level tracking stages of the system, it is important to work in 3D scene space. To accomplish this, intrinsic and extrinsic camera parameters must be obtained in order to to go from image space to scene space. Many man-made environments contain rectilinear structures in the scene. We have used algorithms that extract vanishing points from the images of parallel lines in such rectilinear scene structures.

All parallel lines in the image converge in the so-called vanishing point [15]. We are interested in finding the vertical vanishing point $\mathbf{V}_Z$ as the center of intersection of the lines which point in the vertical direction. Two lines are sufficient to find $\mathbf{V}_Z$, but in a noisy environment it is beneficial to consider more lines to achieve higher accuracy in the location of the vertical vanishing point $\mathbf{V}_Z$. This is computed as the centroid of the intersection points of the images of all the 3D vertical lines. In our application environment there is an abundance of man-made rectilinear structures with vertical lines that can be used for that purpose (isles, boxes, markings on the floor, doors and windows).

In the calibration phase, a number of lines, parallel in space are designated manually with a help of a simple point and click interface (figure 3). Each line is represented as two endpoints $\mathbf{e}_1 = [x_1, y_1]$ and $\mathbf{e}_2 = [x_2, y_2]$

Prior to computing the vanishing point all line endpoints are converted into the homogeneous coordinates with the origin in the center of the image $[\frac{w}{2}; \frac{h}{2}]$, where w and h are the width and height of the image in pixels, respectively. The scaling factor is set to the average of image half-width and half-height $(w+h)/4$ for better numerical conditioning.

$$\mathbf{e}'_1 \quad = \quad [x_1 \times \frac{w}{2}, y_1 \times \frac{w}{2}, (w+h)/2]$$

$$\mathbf{e}'_2 \quad = \quad [x_2 \times \frac{w}{2}, y_2 \times \frac{w}{2}, (w+h)/2]$$

Then in homogeneous coordinates each line can be computed as a cross-product of its endpoints $l = \mathbf{e}'_1 \times \mathbf{e}'_2$.

The 3×3 "second moment" matrix M is built from an array of lines $\mathbf{l}_i$ and $\mathbf{V}_Z$ is computed from the solution of M by singular value decomposition as the eigenvector that corresponds to the smallest eigenvalue [16].

Figure 3: VZ can be found by manually marking two or more vertical straight lines

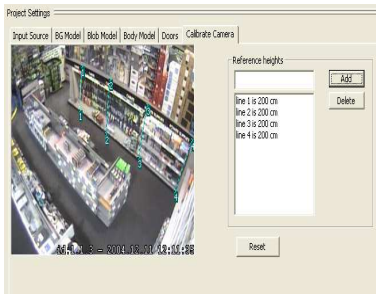

Figure 4: Marking the objects of known height to determine the scale

3.2 Background Modeling and Subtraction

Video sequences from the in-store surveillance cameras are frequently compressed with MPEG-like algorithms, which normally create a periodic noise on the level of a single pixel. We have incorporated a multi-modal statistical background model based on the codebook approach implemented in [17] with a number of improvements.

Each pixel in the image is modeled as a dynamically growing vector of codewords, a so-called codebook. A codeword is represented by: the average pixel RGB value and by the luminance range I_{low} and I_{hi} allowed for this particular codeword. If an incoming pixel is within the luminance range and within some proximity of RGB of the codeword it is considered to belong to the background. During the model acquisition stage the values are added to the background model at each new frame if there is no match found in the already existing vector. Otherwise the matching codeword is updated to account for the information from the new pixel. Empirically, we have established that there is seldom an overlap between the codewords. However if this is the case, i.e more than one match has been established for the new pixel, we merge the overlapping codewords. We assume that the background noise due to compression is of periodical nature. Therefore, at the end of training we clean up the values ("stale" codewords) that have not appeared for periods of time greater than some predefined percentage frames of in the learning stage as not belonging to

the background. For this as outlined in [17], we keep in each codeword a so-called "maximum negative run-length ($MNRL$)" which is the longest interval during the period that the codeword has not occurred. One additional benefit of this modeling approach is that, given a significant learning period, it is not essential that the frames be free of moving foreground object. The background model can be learned on the fly, which is important in the in-store setting.

As a further enhancement we eliminated the background learning stage as such to enable our system to operate dynamically. This was done by adding the age parameter to each codeword as the count of all the frames in which the codeword has appeared. Now, we can start background subtraction as soon as the majority of the codewords contain "old-enough" modalities. Typically, around 100 frames in our test sequences presented in section 6 were enough for reliable detection of the foreground objects. This improvement also allows us to perform the removal of "stale" codewords periodically and not as a one-time event. Now, to determine the "staleness" of a codeword we consider the ratio between its $MNRL$ and it overall age. We have found that when employing "stale" pixel cleanup for the heavily compressed sequences the length of the codebook required to encapsulate the background complexity within one pixel is usually under 20 codewords.

Additionally, we store the number of the last frame number f_{last} in which the codeword was activated (i.e. it matched a pixel). To make our model dynamic, we discard the codewords that have not appeared for long periods of time, which can be computed as the difference between the current frame and f_{last} for any given codeword. Such instances are indicating that the interior has change, due to possibly a stationary object placed or removed from the scene, thus causing our model to restructure dynamically.

The binary mask after background subtraction is filtered with morphological operators to remove standalone noise pixels and to bridge the small gaps that may exist in otherwise connected blobs. This results in an array of blobs created where each blob b is represented as an array of vertices b_i, $i = 1, \ldots, n$ in two-dimensional image space. The vertices describe the contour of b in which each adjacent pair of vertices b_j and b_i is connected by a straight line.

3.3 Finding Head Candidates

The surveillance cameras are typically mounted on the ceiling, more than ten feet above the ground. This can be advantageous in discriminating separate humans within a crowd. The head of a human will have the lowest chance to be occluded, therefore we pursued the goal of finding head candidates - points that represent the tops of the heads in the blob. In this section, we describe our approach in more detail.

To generate human hypotheses within a blob detected in the scene we have used a principle similar to that of the vertical projection histogram of the blob. Our novel method utilizes information about the vanishing point location we obtain from the camera during the calibration stage. The projection of the blob is done along rays going through the vanishing point instead of the parallel lines

projecting onto the horizontal axis of the image.

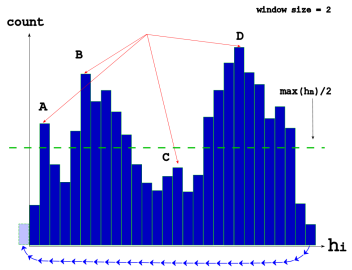

Figure 5: Vanishing point projection histogram

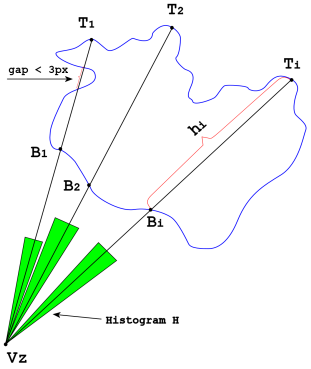

Figure 6: Vanishing point projection histogram

In our implementation each foreground blob is represented as an array of contour vertices $\mathbf{T}_i$ (see figure 6), converted to homogeneous coordinates as described in section 3.1. For each i our method starts at $\mathbf{T}_i$ and counts the number of pixels h_i along the line $r_i = \mathbf{T}_i \times \mathbf{V}_Z$ coming through the vanishing point, obtained earlier as part of camera calibration process.

Then r_i is rasterized by Bresenham's algorithm. Notice that $\mathbf{V}_Z$ is an ideal point which can sometimes fall out of the image boundary or even be situated at an infinity (in the case that the 3D parallel lines are also parallel to the image plane). Therefore we needed to modify the rasterization algorithm to stop as soon as it reaches the image boundary or $\mathbf{V}_Z$, whichever comes first. Note that there is no risk of the process spreading to adjacent blobs, because the foreground mask is rendered for each blob from its contour independently.

The process continues even after the end of the foreground region is reached, which can be defined as the first non-foreground pixel, to allow for important contour concavities, such as arms as well as gaps that are due to camera noise (e.g. see the line originating from $\mathbf{P}_1$ in 6). The last foreground pixel reached in such a manner is considered a bottom candidate $\mathbf{B}_i$ and the count of foreground pixels between $\mathbf{T}_i$ and $\mathbf{B}_i$ is recorded into the histogram bin i. The rays where $\mathbf{T}_i = \mathbf{B}_i$ are discarded as coming from the "underside" of the contour.

Resulting from this is our vanishing point projection histogram $H = [h_i]$. We attempt to isolate local maxima in the histogram in two steps. First, the value h_i is considered a local maximum within a window if it is greater or equal of M of its neighbors on either side (figure 5 shows as an example the window of size $M = 5$).

$$h_i \geq h_j, \forall j = i \pm \frac{M-1}{2}$$

Because this may result in a number of neighboring vertices of with equal values of h selected as local maxima, we merge all such peaks within their window

M and use their average as a candidate. Notice that to represent the cyclic nature of the contour for the leftmost and rightmost bins the neighbors are wrapped around from the end or the beginning of the histogram correspondingly. Typically the window size can be determined as the total number of bins in the histogram divided by the maximum amount of candidates allowed with one blob. This number is set normally from 3 to 10 depending on the average complexity or "crowdedness" of the scene. After this stage all the local peaks $h_i < \max_n(h_n)/2$ are further removed to ensure that we are only considering the vertices from that correspond to the upper parts of the body.

3.4 Human height detection

Utilizing the same interactive approach used to obtain $\mathbf{V}_Z$ (figure 6) we also have found $\mathbf{V}_X$ and $\mathbf{V}_Y$ (see section 3.1 for more details). Note that for a stationary camera this calibration procedure has to be performed only once for the entire video sequence, assuming the environment does not change. In the same manner (figure 4), the user can designate a number of vertical linear segments of known height (e.g. isles, shelves or boxes). Using the heights of the reference objects to compute the projection scale and knowing the positions in the image of head candidates with their corresponding floor locations we have employed the approach from [18, 19] to find human heights in centimeters.

4 Bayesian Tracking

As we outlined in section 2 after creating camera and background models we have sufficient information to build a tracking system. The components of our tracking system are described below.

4.1 Bayesian model: observations and states

We formulate the tracking problem as the maximization of posteriori probability of the Markov chain state. To implement Bayesian inference process efficiently we model our system as a Markov chain $M = \{x, z, x_0\}$ and employ a variant of Metropolis-Hastings particle filtering algorithm [20]. The state of the system at each frame is an aggregate of the state of each body $x_t = \{b_1, \ldots, b_n\}$. Each body, in order, is parametrically characterized as $b_i = \{x, y, h, w, c\}$, where x, y are coordinates of the body on the floor map, h, w its width and height measured in centimeters and c is a 2D color histogram, represented as 32 by 32 bins in hue-saturation space. The body is modeled by the ellipsoid with the axes h and w. An additional implicit variable of the model state is the number of tracked bodies n.

4.2 Computing Posterior Probability

The goal of our tracking system is to find the candidate state x' (a set of bodies along with their parameters) which, given the last known state x, will best fit

the current observation z. Therefore, at each frame we aim to maximize the posterior probability

$$P(z|x, x') = P(z|x) \cdot P(x|x') \tag{1}$$

The right hand side of equation (1) is comprised of the observation likelihood and the state prior probability. They are computed as joint likelihoods for all bodies present in the scene as described below.

4.2.1 Priors

The first type of priors imposes physical constraints on the body parameters. Namely, body width and height are weighted $N(h_\mu, h_\sigma^2)$ and $N(w_\mu, w_\sigma^2)$, with the corresponding means and variances reflecting the dimensions of a normal human body.

Body coordinates x, y are weighted uniformly within the rectangular region R of the floor map. Since we track bodies which are partially out of the image boundaries R slightly exceeds the size of what corresponds to the visible part of the image to account for such cases.

The second type of priors sets the dependency between the candidate state at time t and the accepted state at time $t-1$. Firstly, the difference between w_t, h_t and w_{t-1}, h_{t-1} lowers the prior probability. As another factor, we use the distance between proposed body position (x_t, y_t) and $(\hat{x}_{t-1}, \hat{y}_{t-1})$ — the prediction from the constant velocity Kalman filter. The state of Kalman filter consists of the location of the body on the floor and its velocity. Although tracking the head seems like a first reasonable solution, we have established empirically that the perceived human body height varies as a result of walking, thus the position of the feet on the floor was chosen as a more stable reference point.

When new body is created it does not have a correspondence, this is when we use a normally distributed prior $N(d_0, \sigma)$, where d_0 is the location of the closest door (designated on the floor plan) and σ is chosen empirically to account for image noise. The same process is taking place when one of the existing bodies is being deleted.

4.2.2 Likelihoods

The second component in forming proposal probability relates the observation to the model state. First, color histogram c is formed by the process of weighted accumulation, with more recent realizations of c given more weight. We then compare Bhattacharya distance between proposed c'_t and corresponding c_{t-1}

Two more components that we use in computing the likelihood are: the amount of blob pixels not matching any body pixels and the amount of body pixels not matching blob pixels. Note that we use a Z-buffer for these as well as for computing the color histogram of the current observation in order to detect occlusions. In this buffer all the body pixels are marked according to their distance from the camera, which we obtain during the calibration process (see

section 3.1). This way only the visible pixels are considered when computing the likelihood.

4.3 Jump-diffusion Dynamics

In essence, the approach of particle filtering is a non-deterministic multivariate optimization method. As such it inherits the problems to which other, classical optimization methods can be prone [20]. Here we present a way to overcome one such problem — traversing valleys in the optimization space by utilizing task specific information. On the other hand, particle filtering methods are robust because they do not require any assumptions about the probability distributions of the data.

4.3.1 Accepting or rejecting the state

Our joint distribution is not known explicitly, so we have chosen to use Metropolis-Hastings sampling algorithm.

$$\alpha(x, x') = min\left(1, \frac{P(x')}{P(x_t)} \cdot \frac{m_t(x|x')}{m_t(x'|x)}\right). \tag{2}$$

Where x' is the candidate state, $P(x)$ is the stationary distribution of our Markov chain, m_t is the proposal distribution. In equation (2), the first part is the likelihood ratio between the proposed sample x' and the previous sample x_t. The second part is the ratio of the proposal density in both directions (1 if the proposal density is symmetric).

This proposal density would generate samples centered around the current state. We draw a new proposal state x' with probability $m_t(x'|x)$ and then accept it with the probability $\alpha(x, x')$. Notice that the proposal distribution is a time function, that is at each frame it will be formed based on the rules outlined below.

To form the proposal distribution we have implemented a number of reversible operators. There are two types of jump transitions and five types of diffuse transitions implemented in our system:

Add body: Draws a random head candidate and adds a new body using its head and foot coordinates. At this point the actual height and floor coordinates of the body are estimated (see Section 3.1).

Delete body: Removes a random body. Body is excluded from further tracking, the path is terminated and saved.

Change height: Similar to height, changes the width of a random body. Body width is stored in the system as a percentage of the height and, thus, does not influence the actual location of the body.

Change width: Similar to height changes the width of a random body. Body width is stored in the system as the percentage of the height and this way does not influence the actual location of the body.

Mean shift: Move one of the existing bodies by applying the mean-shift operator with weighted anisotropic Gaussian kernel. The kernel is formed as a Gaussian, elliptically-shaped mask, where the weights increase with increased Mahalanobis distance. Additionally, if a pixel value of the foreground mask (corresponding to the background) is zero or the same pixel value from the Z-buffer is greater (i.e located further from the camera) than the current body, the weight in the kernel is effectively zeroed out. This, in essence, performs a standard color-based mean shift, but accounts only for the pixels belonging to the hypothesized body model.

Move: Second type of position shift is moving the body to a random "initial head candidate" drawn from a pool of head candidates contained in some proximity from the current body position. It allows for the head candidates, not initially revealed (possibly due to image noise), to be considered in the subsequent frames.

Switch IDs: Select two random neighboring bodies and exchange their IDs. The unique body ID stays the same throughout the visible life of the body, that is why switching IDs is in essence performing two "move" diffusions.

The Z-buffer is updated after each transition to reflect the new occlusion map. Notice that we use a set of controllable weight probabilities to add more emphasis to one or another transition type. In our application normally around 100 jump-diffuse iterations are required for each frame to reach convergence.

5 Activity Recognition

5.1 Obtaining Shopper Trajectories

As each accepted body candidate progresses along the floor map, we record at each step the x, y coordinates along with the unique body ID and append it to a separate data structure $A_i = \{x, y, ID\}$, where i is the path number. Even when the body exits the scene, A_i is not deleted. This way at each time moment the array A represents a complete trajectory history for all the bodies traveled in the scene since the start of a sequence. Equipped with this information we can proceed to shopper groups detection.

5.2 Determining Shopper Groups by Clustering Motion Trajectories

We assume that people who shop together can be identified by the following criteria: they enter the scene together, leave the scene together, have a small mean intra-group distance, have a small mean difference between paths.

When comparing two trajectories as signals, there are two important aspects: shift time between two signals and signal shape. To elegantly incorporate both of these considerations and to account for all empirically established criteria we have created the proximity metric based on Euclidean distance and the correlation of two signals.

$$f_{ij}(T) = \int \left[\big(x_i(t) - x_j(t+T)\big)^2 + \big(y_i(t) - y_j(t+T)\big)^2 \right] \mathrm{d}t \tag{3}$$

$$d_{ij} = \int f_{ij}(T) \times N(0, \sigma^2) \mathrm{d}T \tag{4}$$

If the time t is discrete, as it is in our case with each measurement corresponding to a single video frame, the equations above can be rewritten as:

$$d_{ij} = \sum_{T=-\Delta}^{\Delta} \left(\sum_{t=t_1+T}^{t=t_2-T} \left[(x_i(t) - x_j(t+T) \right]^2 + \left[y_i(t) - y_j(t+T) \right]^2 \right) \frac{N(0, \sigma^2)}{t_2 - t_1} \tag{5}$$

Thus the distance between two trajectories d_{ij} is the weighted sum of trajectory proximities at each time moment. The interval $[-\Delta : \Delta]$ is a time cutoff that can optimize the computation.

The standard deviation of normally distributed weights σ can be increased to account for higher time spread between people in the same group. We have chosen $\Delta = 3\sigma$.

We start the integration at time t_1 when is the first time that both objects are visible in the scene and end it at time t_2, when at least one object has left the scene. The interval $[t_1; t_2]$ is sometimes referred to as *longest common subsequence* in the literature. Note that since there must be no favoring shorter or longer paths, the distance measure is normalized by the length of the traveled segment $t_2 - t_1$.

Since people in a shopping group are not guaranteed to appear as well as leave the scene at the same time, we propose to compute the trajectory similarity measure on piecewise uninterrupted segments — i.e., the intervals of time where both bodies in question were present and successfully tracked in the scene.

Naturally, we do not compute the distance for the pairs with a very small longest common sub-trajectory, because these will not render a statistically significant metric. The cutoff for the common subsequence length was chosen empirically at 3 seconds, or 45 frames (at 15 fps).

Furthermore, for computational efficiency, and considering the fact that physical location of one person does not change significantly within a time interval of 10 frames (< 1 sec), we can sub-sample the trajectories and reduce the

computational load. Once the pairwise distances for all trajectories are known, as our next step we cluster the paths based on this measure.

We apply agglomerative hierarchical clustering, where each object is initially placed into its own cluster C. Therefore, if we have N objects to cluster, we start with N singleton groups.

Before we start the clustering, we need to decide on a threshold distance. Once this is done, the procedure is as follows:

1. Compare all pairs of groups and mark the pair that is closest.
2. The distance between this closest pair of groups is compared to the threshold value.
   (a) If the distance between this closest pair is less than the threshold distance, these groups become linked and are merged into a single group. Return to Step 1 to continue the clustering.
   (b) If the distance between the closest pair is greater than the threshold, the clustering stops.

If the threshold value is too small, there will still be many groups present at the end, and many of them will be singletons. Conversely, if the threshold is too large, objects that are not very similar may end up in the same cluster. We optimized the threshold value during a trial and error process on video sequences of varying complexity.

When merging two clusters, the center point of the new cluster at each frame C' is determined as a weighted average of two paths corresponding to the centers of the merged clusters

$$C'_t = C^1_t \cdot |C^1| + C^2_t \cdot |C^2| \tag{6}$$

where C_t is the location along the path of the group C at time t and $|C|$ is the number of the trajectories belonging to the cluster with center in C.

6 Results and Discussion

We have tested low-level parts of our system on a number video sequences from two different cameras (figure 8 (a)-(f)) mounted in a retail store chain and on the publicly available CAVIAR dataset [21] (figure 8 (g)-(l)). Some sample frames and results of the head candidates detection as well as height estimation from the test video sequences are presented in figure 8.

One of the most frequent cases of detecting false positives was occurring when there was not enough frames allotted for the background acquisition and therefore some people standing were interpreted as part of the background. When these people later moved, not only the moving person but the pixels where she used to stand are detected as a foreground objects. The background subtraction approach has given good results even under extreme lighting conditions (see (i) and (j) in figure 8).

Analyzing falsely detected head locations, we see that these are primarily due to the video compression artifacts influencing the background subtraction process. Nevertheless, the algorithm has shown robust performance with the significant levels of illumination noise, under the low-quality, real-life capturing conditions.

The false negative head candidates were primarily due to two reasons. First, parts of the foreground region become separated from the body or sometimes a part of the shadow is considered as a separate body, and this causes a false candidate to be detected (see (k) in figure 8). We believe that human shape modeling may help solve this problem. A second factor that badly influences the detection is when the heads are not pronounced enough to create a local maximum in the histogram (see (l) in figure 8). This problem can be attended in the future by color and texture analysis within the blob.

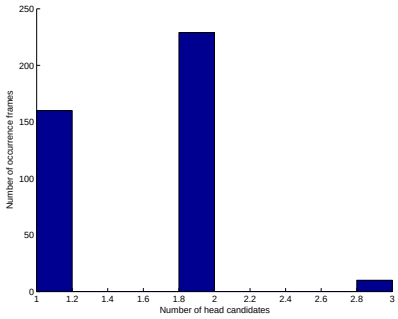

Figure 7: Algorithm performance evaluation. This graph shows the number of frames when 1, 2, or 3 heads were detected. The true number of heads is 2.

To partially evaluate the quality of the results we have analyzed a number of detected head candidates in the sequences with two people, that were detected as a single blob (Figure 7). The evaluation shows that the outputs from our methods can be used at the initialization stage of tracking algorithm. To further evaluate the quality of our method candidate hit/miss and average error analysis based on their coordinates may be required.

We performed preliminary evaluation of our tracking system for the presence of three major types of inconsistencies: misses, false hits and identity switches. A *miss* is when the body is not detected or detected but tracked for an insignificant portion of its path ($< 30\%$). A *false hit* is when a new body is created where there is now actual person present. Most of the false hits are a result of more than one body in the model being assigned to a single body in the scene. An *identity switch* is when two or more bodies exchange their IDs once within the close proximity from each other. By visually counting the number of

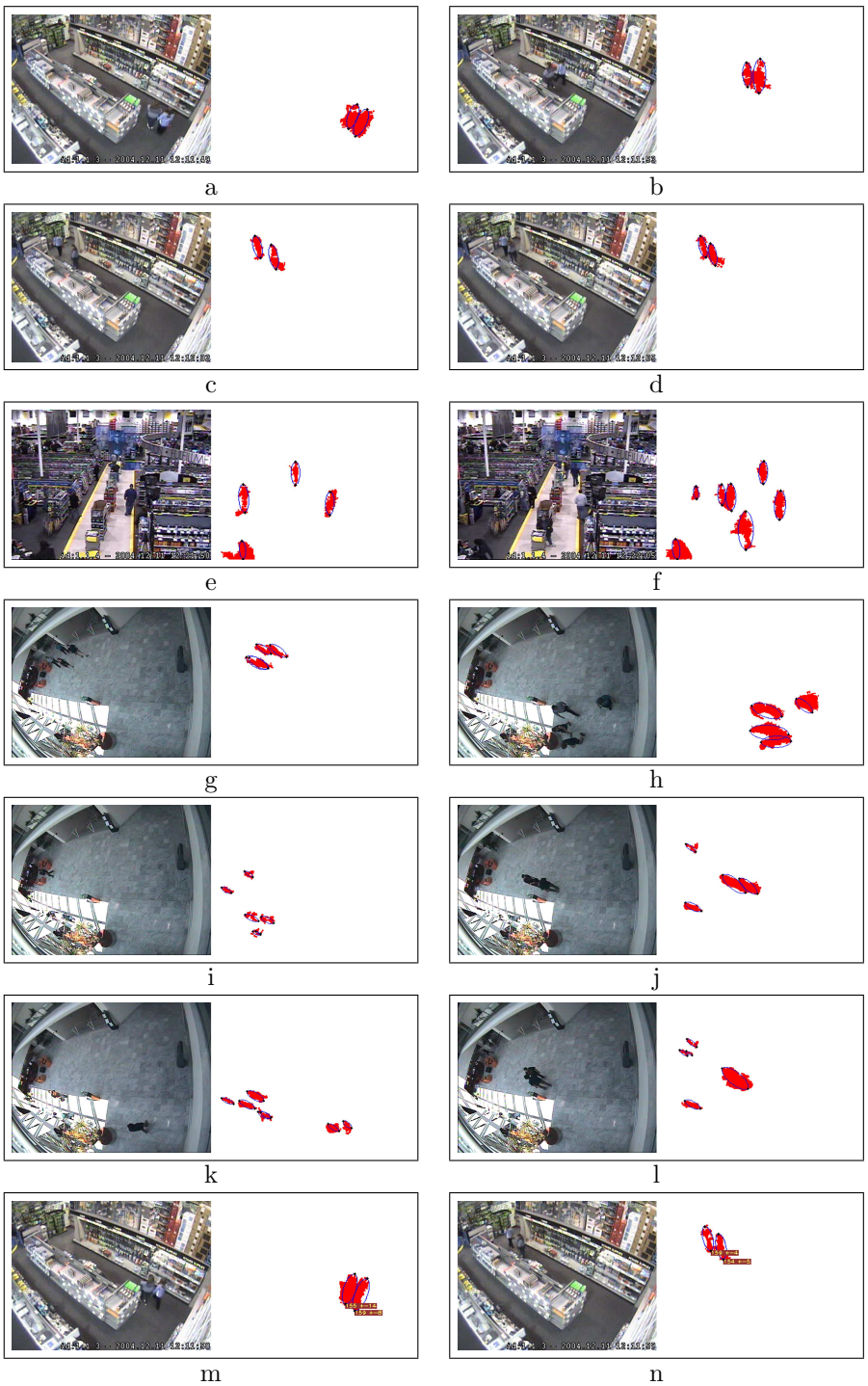

Figure 8: (a) - (l) Head candidates from test frames. Left image is the original frame. On the right image red represents foreground mask, small black dots indicate the locations of $\mathbf{T}_i$ and $\mathbf{B}_i$; blue ellipses are fitted with $\mathbf{T}_i\mathbf{B}_i$ as the major axis; (m) and (n) Height detection: brown plates contain height mean and variance for each ellipse

each of types of errors on a number of sequences of overall 6000 frames we have obtained results summarized in table 1.

Sequence	Total frames	Total people	Misses	False hits	Identity switches
1	3181	19	0	2	3
2	1287	18	0	8	4
3	1510	17	2	8	6
ALL	5978	54	2	18	13
%%	100	100	3.7	33.3	24.1

Table 1: Tracking results (based on the manually observed ground truth)

The most common mistakes made by the tracker, were false hits. We have observed that the majority of false hits (more than 50%) are short lived, i.e. typically last for only several frames. These cases can be further post-processed by temporal filtering to remove insignificatly short paths. Sometimes, however, false detections are accompanied by ID switches, when the body tracked for a long time is substituted for a false hit. This presents a more complicated case and deserves further study.

Overall performance of the tracker is promising, primarily because it produces satisfactory detection and prolonged tracking in the crowded scenes. The output from our tracking module serves as a reliable base for obtaining customer paths (Figure 9) and the detection of shopper groups (Figure 10).

Shopper groups determined by clustering motion trajectories coincide with the results of visual observation. The metric becomes more reliable as the length of the largest common subsequence increases for trajectories under comparison.

The video sequences that we used were real recordings from a large retail store, performed with 4 monocular digital cameras placed at different locations and angles of view. The processing was done on a dual core Pentium processor at around 7 fps, with the actual recording done at 15 fps. With a number of optimizations and slightly increased processing power we estimate the system to perform tracking and customer grouping at the real-time speeds.

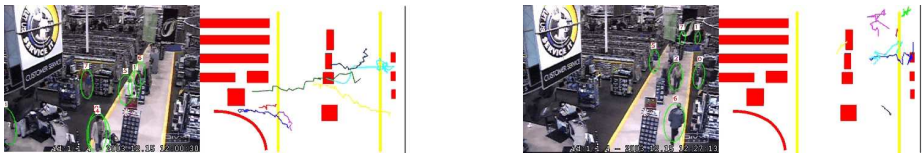

Figure 9: Select frames showing customer path marked on the floor map (for each frame the original sequences is on the *left* and the top-down view with the floor map in on the *right*

Figure 10: Select frames showing the detection of shopping groups (marked by white rectangles)

7 Future Work

The next steps in activity recognition we intend to take are: identifying queue lengths and queue wait times, building store traffic heat maps, aisle penetration maps and estimating customer conversion rate, i.e. the number of people making purchases in relation to the total number of customers in the store. Moreover, the color histogram information, recorded for each customer can potentially be used to determine person's age, gender or ethnicity.

We plan to extensively validate the accuracy of group detection algorithm using the manually marked dataset of more than 30000 frames provided by CAVIAR project [21]. Although the evaluation of the tracking subsystem shows promising results, the authors are aware that a more formal evaluation has to be performed for the each of the customer activity characteristics.

Another potential improvement is to enhance the quality of our depth maps using 3D CAD model of the store, which we expect to result in highly stable tracking. Such models, currently under development, will incorporate the layout of the store fixtures, product placement and camera location. We are currently also investigating the use of a single panoramic camera to cover large area in the store, which could provide continuous tracking for longer uninterrupted periods of time.

We are currently also investigating the use of a single panoramic camera to cover large area in the store, which could provide continuous tracking for longer uninterrupted periods of time. Combined with the customer counting camera installed at the entrance this method will allow to compute the percentage customer distribution in the different areas of the store as well as provide important clues into the "conversion rate" analysis (the ratio of the amount of purchases to the total number of customers). With the increased capturing quality we hope to get enough detail to perform an analysis of certain product interaction aspects: attention (i.e. turning the torso towards the product, or squatting/reaching for the product), browsing (when hands are performing "reaching out" gestures).

In future the position and orientation of body ellipsoid can be combined with multiple-view color representation for more reliable color tracking [22]. We believe that this kind of tracking will provide information for customer attention analysis, such as rough estimations of customer gaze center or interactions within

customer groups [8].

References

[1] Burke, R.: The third wave of marketing intelligence. In Krafft, M., Mantrala, M., eds.: Retailing in the 21st Century. Springer-Verlag (2005)

[2] Horprasert, T., Harwood, D., Davis, L.: A statistical approach for real-time robust background subtraction and shadow detection. In: In Proc. of International Conference on Computer Vision. (1999)

[3] Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: International Conference on Computer Vision and Pattern Recognition. (1999)

[4] Haritaoglu, I., Harwood, D., Davis, L.: W-4: Real-time surveillance of people and their activities. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 809–830

[5] Zhao, T., Nevatia, R.: Tracking multiple humans in crowded environment. In: International Conference on Computer Vision and Pattern Recognition. (2004)

[6] Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., Wixson, L.: A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Carnegie Mellon University, Pittsburgh, PA (2000)

[7] Mittal, A., Davis, L.: M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo. In: European Conference on Computer Vision. (2002)

[8] Haritaoglu, I., Flickner, M.: Detection and tracking of shopping groups in stores. In: International Conference on Computer Vision and Pattern Recognition. (2001)

[9] Havasi, L., Sziranyi, T.: Motion tracking through grouped transient feature points. In: Advanced Concepts for Intelligent Vision Systems. (2004)

[10] Elgammal, A., Davis, L.: Probabilistic framework for segmenting people under occlusion. In: International Conference on Computer Vision. (2001)

[11] Isard, M., MacCormick, J.: Bramble: A bayesian multiple-blob tracker. In: International Conference on Computer Vision. (2001)

[12] Kemp, C., Drummond, T.: Multi-modal tracking using texture changes. In: British Machine Vision Conference. (2004)

[13] Sminchisescu, C., Triggs, B.: Kinematic jump processes for monocular 3d human tracking. In: International Conference on Computer Vision and Pattern Recognition. (2003)

[14] Buzan, D., Sclaroff, S., Kollios, G.: Extraction and clustering of motion trajectories in video. In: International Conference on Pattern Recognition. (2004)

[15] Forsyth, D.A., Ponce, J.: Computer Vision: A Modern Approach. Prentice Hall Professional Technical Reference (2002)

[16] Collins, R., Weiss, R.: Vanishing point calculation as a statistical inference on the unit sphere. In: International Conference on Computer Vision. (1990) 400–403

[17] Kim, K., Chalidabhongse, T., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: International Conference on Image Processing. (2004)

[18] Criminisi, A., Zisserman, A., Van Gool, L., Bramble, S.: A new approach to obtain height measurements from video. In: SPIE. (1998)

[19] Criminisi, A., Reid, I., Zisserman, A.: Single view metrology. International Journal of Computer Vision **40** (1999) 123–148

[20] Doucet, A., de Freitas, N., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer-Verlag (2001)

[21] CAVIAR: Ist 37540. Found at http://homepages.inf.ed.ac.uk/rbf/CAVIAR/ (2001)

[22] Leykin, A., Cutzu, F., Tuceryan, M.: Using multiple views to resolve human body tracking ambiguities. In: British Machine Vision Conference. (2004)