# Multiperson Tracking Using Kalman Filter

Salil P. Banerjee, Kris. Pallipuram

December 10, 2008

## Abstract

This paper addresses the problem of implementation of the Kalman filter to track multiple persons in a room. First, an occupancy map of the room has been created using the six cameras which are distributed across the room. Then the Kalman filter has been implemented to track the centroids of the persons detected in the room. The Kalman filter tracks persons even when their blobs merge, providing increased efficieny in tracking multiple persons in the room.

## Introduction

Tracking is the process of locating moving objects in time using a camera. An algorithm analyzes each frame and outputs the location of moving targets within the video frame[1]. This algorithm has two basic steps - detection of moving objects (here people) in each frame and filtering/tracking them in each consecutive frame. Detection of objects is a bottom up approach and have low computational complexity. There are various algorithms such as the blob tracking, kernel-based tracking, contour tracking and visual feature matching, which can be used for the detection algorithm. Filtering/tracking is basically a top down process and have higher computation complexity. Kalman filter and particle filter are two popular algorithms used for filtering/tracking moving objects.

In 1960, R.E. Kalman published his paper describing a recursive solution to a discrete data linear filtering data. Since that time, extensive research and development has taken place on the Kalman filter due to the advances in digital computing[2].

The Kalman filter is an algorithm which smooths the measurements by weighting them against the predicted values by their variances. In other words, the Kalman filter tries to find a balance between predicted values and noisy measurements. The values of the weights are determined by modelling the state equations. The purpose of the Kalman filter is to track the system being measured at discrete intervals of time.

For a completely known system the noise can be reduced by the repeated application of window filters such as the mean filter,the median filter, the averaging filter and the gaussian filter. However these filters can only be applied to a completely known system. The Kalman filter differs from these window filters in respect that it predicts the next state based on the previous states and needs no information of the future states. Its main purpose is to predict and smooth the predicted state.

Our approach has been to implement the Kalman filter on the centroids of the people detected in the occupancy map. The occupancy map and the centroids of the people have already been computed for the Frogger[3] game. In this problem, the state variables are the x and y coordinates of the centroid of each tracked person.

The Kalman filter has been implemented using the software Visual C++ version 6.0 running on a ACPI Multiprocessor PC having a 3.2 GHz Intel Xeo Processor with 1 GB of RAM and Windows 2000 Professional service pack 4 as the operating system.

# Methods

## System

The system is located in the sensor network laboratory in the basement of Riggs Hall. The floor of the sensor network laboratory is used as trackable space. Six static cameras have been fixed across the room, four at the corners and two at the center of the room length. The layout of the room can be shown in Figure 1. The program for starting the system, camera calibration, acquiring live images from the camera, background subtraction, centroid calculation and person identification decision making is credited to Dr. Adam Hoover and Bent Olsen[3].

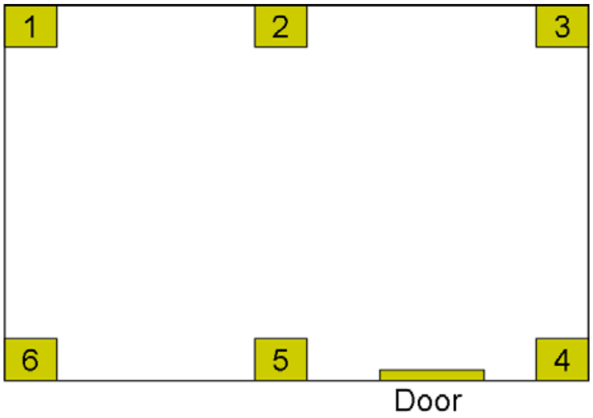


Figure 1: Basic block diagram of the layout of the room and the cameras.

In order to detect the people from the live images that have been captured, segmentation is required. In our project segmentation has been achieved by the creation of the occupancy map from the six static cameras. An occupancy map is a two dimensional raster image, uniformly distributed in the floor plane. Each map pixel contains a binary value, signifying whether the designated floorspace is occupied or not. A spatial frame of the occupancy map is computed from a set of intensity images, one per camera, captured simultaneously using synchronous signal. The system provides a 480×640 resolution occupancy map. The approach that has been used to create the occupancy map is the image-freespace perceptual paradigm. The model starts with an assumption that every space in the image is filled, that is, has a value of 1 in case of a binary image. It then processes this image to clear out the space. This effectively ignores the observed object pixels. Hence a cell of the model is transformed from being occupied to empty. Thus if a camera cannot see an object then it marks the corresponding pixel in its image as having a zero value, thereby creating a freespace. The union of the freespaces from multiple cameras creates a reasonable picture of occupied space.

Creation of the occupancy map involves three basic steps - calibration, acquisition of the background image and then finally determining the occupancy map for each frame. The cameras are uncalibrated and need to be calibrated so as to represent the two dimensional image plane into points in the three dimensional world points. The transform $T_n$ for each camera's image space $I[n,c,r]$ to the $(x,y,z)$ world space is given by

$$T_n : [n,c,r] \rightarrow (x,y,z) + (i,j,k)d \quad d > 0 \qquad (1)$$

where d is the distance from the camera to the ground, n is the camera number, c and r are the columns and rows of the image formed in the camera plane respectively.

Next step is to acquire background image B[n,c,r] for each camera while the floorspace to be monitored is empty. A binary mask M[n,c,r] is created for each background image. If the pixel in the mask is 0, it denotes empty floorspace. The program uses polygon drawing tool to create the mask space. The floorspace used by our experimentation is cleared of chairs, tables and person, while keeping heavy drawers near the walls of the room.

The third and the final step is determine the occupancy map cell O[x,y] that each pixel I[n,c,r] views. The basic algorithm is to detect the difference between the background image $B[n,c,r]$ and the live image $I[n,c,r]$ for each camera. The difference image is computed as

$$D[n,c,r] = \begin{cases} 1 & if|I[n,c,r] - B[n,c,r]| > T \\ 0 & if|I[n,c,r] - B[n,c,r]| \leq T \end{cases} \qquad (2)$$

where the threshold T controls the sensitivity of the algorithm.

Using the above equation, an occupancy map cell is designated empty floorspace if atleast one image pixel that views it sees no difference. Hence a cell is designated occupied if every image pixel sees a difference. The occupancy map can be shown in the Figure 2

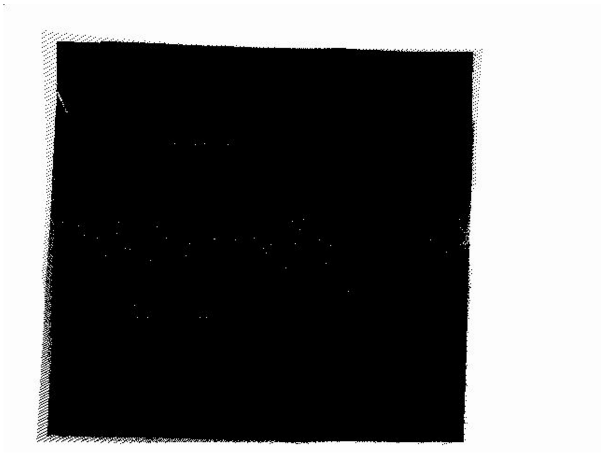


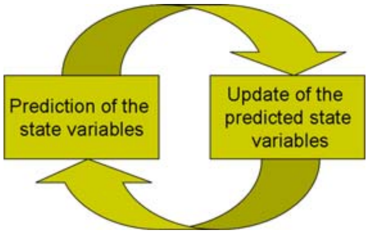
Figure 2: A snapshot of the occupancy map of the room with no objects.

Some of the advantages of this method are that segmentation is avoided, data fusion is performed at the pixel level and triangulations performed for centroid computation can be done offline.

## Tracking

### Kalman filter

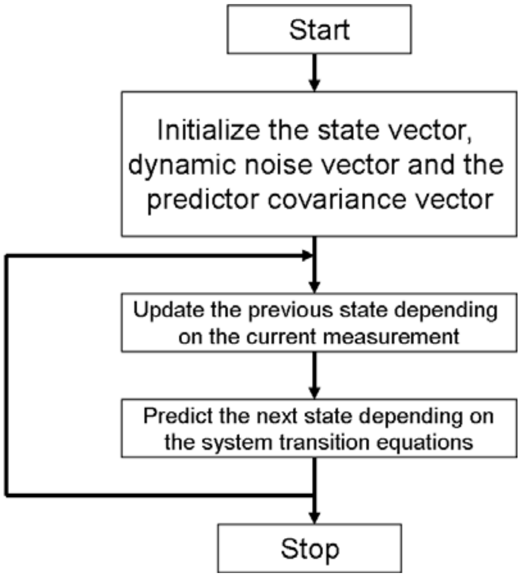
A recursive minimum mean-square estimator such as a Kalman filter consists of two basic steps. They are the prediction of the next state using the current set of observations and the use of innovation to update the current set of predicted measurements. The second step is basically smooths the predicted values and gives a much better estimate of the next state. This can be shown in the following Figure 3.



Prediction of the state variables
Update of the predicted state variables

Figure 3: Basic step of the Kalman Filter.

The Kalman filter tries to find a balance between predicted values and noisy measurements. The values of the weights are determined by modelling the state equations. The purpose of the Kalman filter is to track the system being measured at discrete intervals of time. The algorithm can be shown in Figure 4.



Start
Initialize the state vector, dynamic noise vector and the predictor covariance vector
Update the previous state depending on the current measurement
Predict the next state depending on the system transition equations
Stop

Figure 4: Kalman Filter Algorithm.

### Implementation

After the person has been extracted from the background, the next step is to track his/her motion. In order to do this, we consider the x and y coordinates of the centroid of the blob defining the person. In this project our state variables are the x and y coordinates of the centroid of the blob. We assume that the person is moving with a constant velocity. The

live images are captured at a frame rate of 30 fps. Hence the time interval between each frame will be 1/30 seconds.
We can now define the state equations as

$$x_t = x_{t-1} + T\dot{x}_{t-1} \tag{3}$$

$$\dot{x}_t = \dot{x}_{t-1} + u_t \tag{4}$$

$$y_t = y_{t-1} + T\dot{y}_{t-1} \tag{5}$$

$$\dot{y}_t = \dot{y}_{t-1} + v_t \tag{6}$$

Therefore the state transition matrix can be defined as

$$\phi = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{7}$$

The covariance matrix of the dynamic noise is

$$\mathbf{U_t} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \sigma_u^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_v^2 & 0 \end{bmatrix} \tag{8}$$

while the measurement noise covariance matrix is given by

$$\mathbf{R} = \begin{bmatrix} \sigma_m^2 & 0 \\ 0 & \sigma_n^2 \end{bmatrix} \tag{9}$$

The observation matrix M is given by

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{10}$$

The Kalman gain matrix $K_t$ is a matrix with dimensions $4 \times 2$ whereas M has dimensions of $2 \times 4$. We subsitute these values in the Kalman filter equations and obtain the predicted and smoothed states. We continue to do this for every detected person in the occupancy map.

# Experimental Results

Figure 5 shows the tracking results in some frames. The video of multi-person tracking can be found at [http://www.mediafire.com/file/ymwmngmdyjy/tracking.avi].

The plot of the measured and estimated values vs. time for various values of dynamic noise are shown in figures 6 to 8.

We observe from the plots, that for lower values of dynamic noise the estimated plot tracks the predicted values more closely, while for higher values of dnoise, the estimated plot tracks the measured vlaues more closely. Also the outlier noise which affects the measurements due to the imperfectness of the measuring system, is smoothed to a great extent by the Kalman filter. This is because of the fact that the filter predicts the next estimate to be along a linear path and does not expect such a sudden increase or decrease in the values of the next measurement.

# Conclusion

We have implemented the Kalman filter for tracking multiple persons, where the two-dimensional data of the x and y positions of the centroid is tracked. We also observe that the Kalman filter not only predicts the next state of the variable, but also attempts to smooth the curve of the predicted values of the variable. Also the persons are tracked effectively, even when they come close to each other and their blobs merge, which is due to the fact that their next positions depend on the value of the predicted values and not the measurements.

The dynamic noise, which is caused by system equations that do not represent an accurate model of the behavior of the system (i.e., truncation of higher order variables such as acceleration, jerk etc.), is smoothed to a great extent by the filter as it is unable to predict the sudden change in the higher order parameters. We also observe that the convergence of the Kalman filter is slower for higher variances of the measurement and dynamic noise and faster for lower variances. The values of the variances of the noises also effect the probability of the tracking being closer to the estimated values or the measurements.

In future, more variables such as the height and width of the bounding box or the values of the corner pixels of the bounding box and also the area of

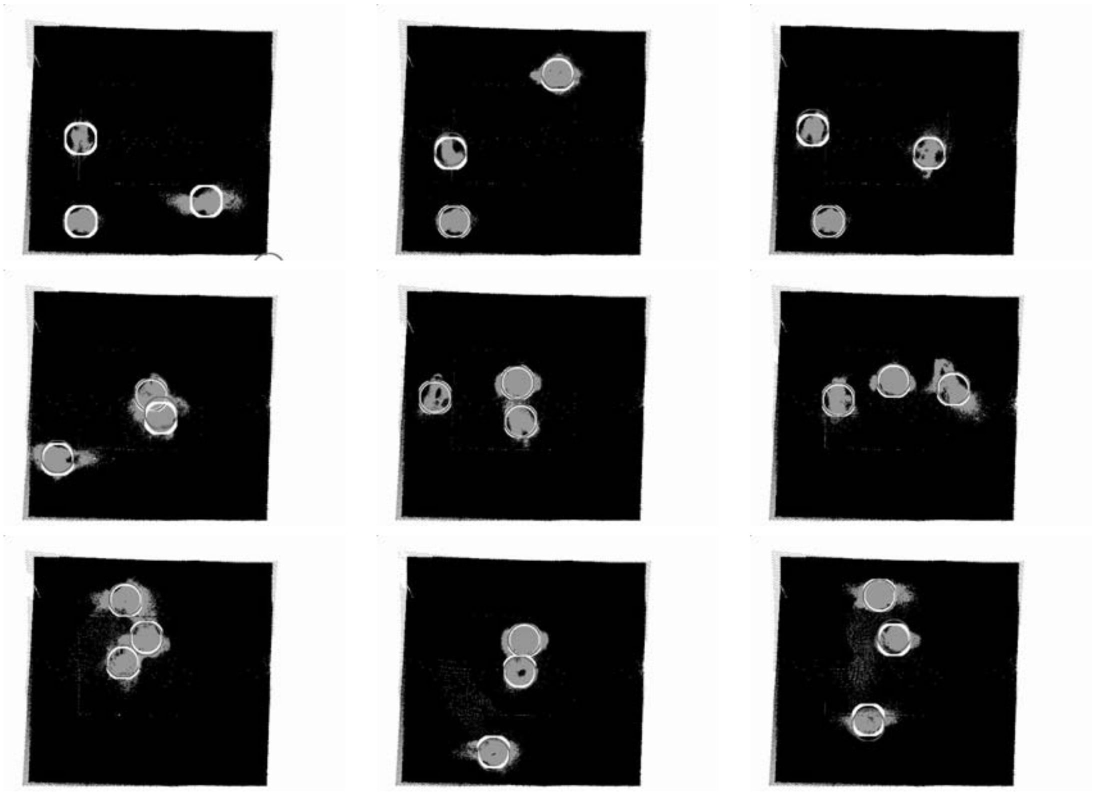


Figure 5: Tracking frame numbers (L-R):(a) Frame number 1, (b) Frame number 92 (c) Frame number 164 (d) Frame number 275 (e) Frame number 651 (f) Frame number 739 (g) Frame number 843 (h) Frame number 931 (i) Frame number 1053.

the blob could be incorporated in order to reduce the dynamic noise and track the person more effectively. Other filters such as the extended Kalman filter, unscented Kalman filter or particle filter could be implemented in order to take care of the non-linear system equations and other non-Gaussian noises.

## References

1. http://en.wikipedia.org/wiki/Video_tracking.

2. Greg Welch, Gary Bishop, An Introduction to the Kalman filter, SIGGRAPH 2001.

3. Adam Hoover, Bent David Olsen: A Real-Time Occupancy Map from Multiple Video Streams. ICRA 1999: 2261-2266.

4. http://en.wikipedia.org/wiki/Kalman_filter

5. Adam Hoover, Kalman Filter(class notes), 2008.

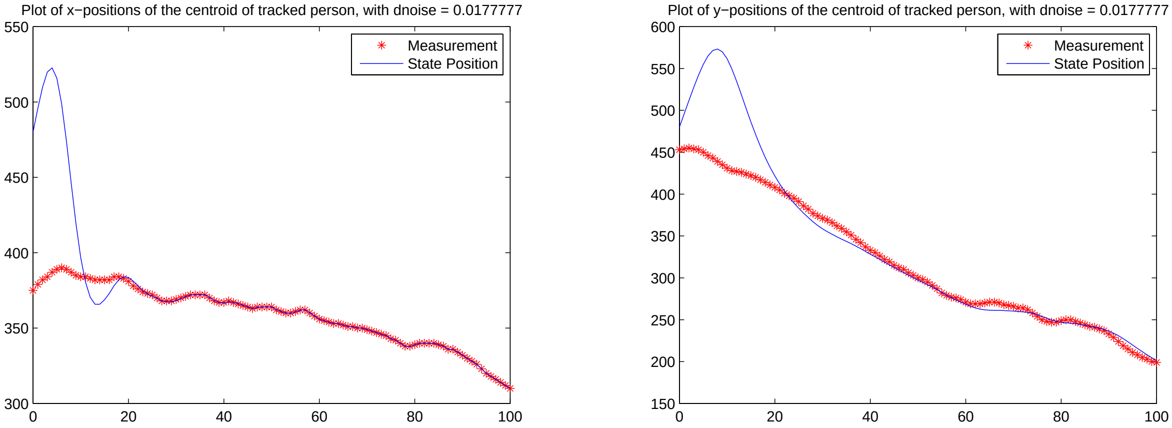


Figure 6: Plot of measured and estimated values vs. time of the centroid for dynamic noise = 0.0177777(a) x-position (b) y-position.

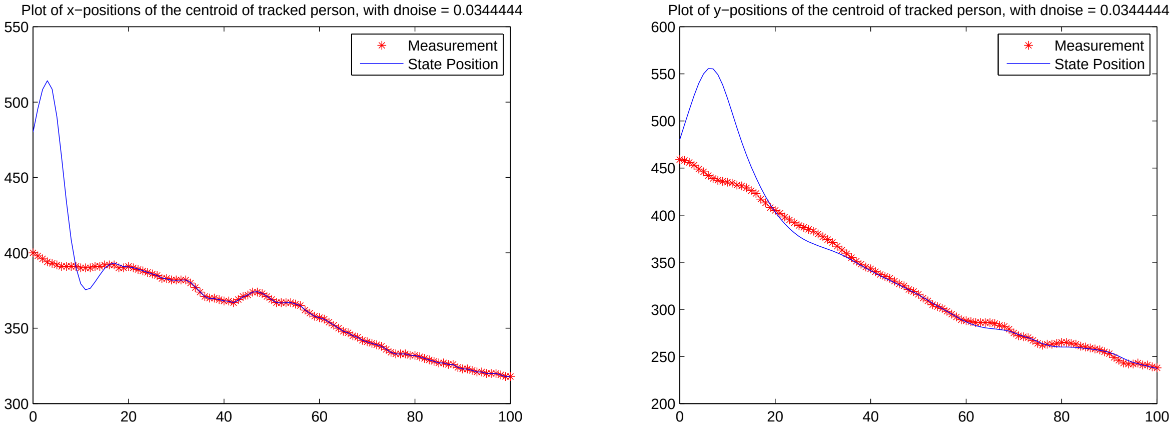


Figure 7: Plot of measured and estimated values vs. time of the centroid for dynamic noise = 0.0344444(a) x-position (b) y-position.

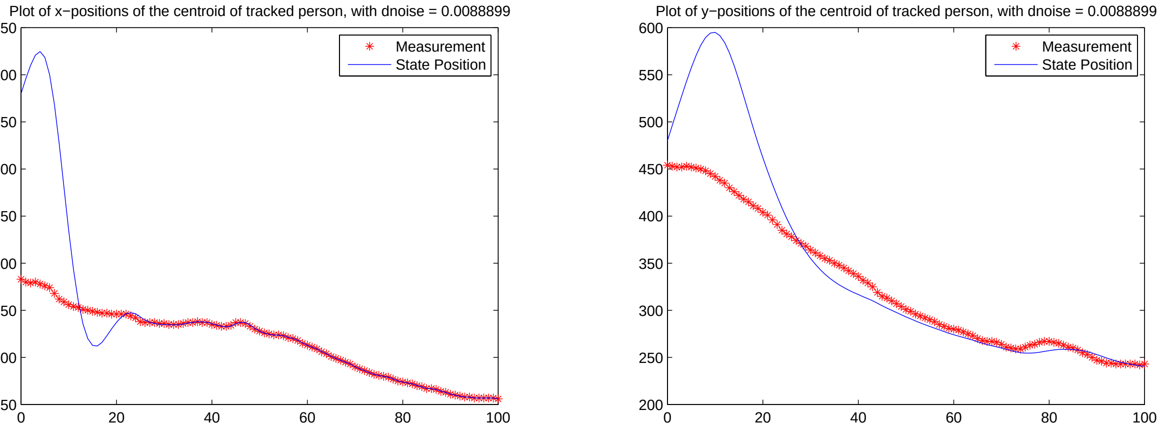


Figure 8: Plot of measured and estimated values vs. time of the centroid for dynamic noise = 0.0088999(a) x-position (b) y-position.