2007

Parametric & non-parametric background subtraction model with object tracking for VENUS

Karthik Chandrasekaran

Parametric & Non-parametric background subtraction model with Object Tracking for VENUS

Masters Thesis

Karthik Chandrasekaran
Department of Computer Science
Rochester Institute of Technology
Rochester, New York
kxc7221@rit.edu

Dec 1, 2007

Thesis Committee:
Chairman: **Dr. Roger Gaborski**
Reader: **Dr. Rajendra Raj**
Observer: **Thomas J Borrelli**

Approved by:

Roger S. Gaborski 12/13/2007

Roger Gaborski
Professor, Department of Computer Science

R. K. Raj

Rajendra K. Raj
Professor, Department of Computer Science

Thomas J. Borrelli 12/18/2007

Thomas J Borrelli
Lecturer, Department of Computer Science

Thesis/Dissertation Author Permission Statement

Title of thesis or dissertation: PARAMETRIC & NON-PARAMETRIC BACKGROUND SUBTRACTION MODEL WITH OBJECT TRACKING FOR VENUS

Name of author: KARTHIK CHANDRASEKARAN
Degree: MASTERS
Program: COMPUTER SCIENCE
College: GCCIS

I understand that I must submit a print copy of my thesis or dissertation to the RIT Archives, per current RIT guidelines for the completion of my degree. I hereby grant to the Rochester Institute of Technology and its agents the non-exclusive license to archive and make accessible my thesis or dissertation in whole or in part in all forms of media in perpetuity. I retain all other ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Print Reproduction Permission Granted:

I, KARTHIK CHANDRASEKARAN, hereby **grant permission** to the Rochester Institute Technology to reproduce my print thesis or dissertation in whole or in part. Any reproduction will not be for commercial use or profit.

Signature of Author: Karthik Chandrasekaran Date: 05/05/08

Print Reproduction Permission Denied:

I, _____, hereby **deny permission** to the RIT Library of the Rochester Institute of Technology to reproduce my print thesis or dissertation in whole or in part.

Signature of Author: _____ Date: _____

Inclusion in the RIT Digital Media Library Electronic Thesis & Dissertation (ETD) Archive

I, KARTHIK CHANDRASEKARAN, additionally grant to the Rochester Institute of Technology Digital Media Library (RIT DML) the non-exclusive license to archive and provide electronic access to my thesis or dissertation in whole or in part in all forms of media in perpetuity.

I understand that my work, in addition to its bibliographic record and abstract, will be available to the world-wide community of scholars and researchers through the RIT DML. I retain all other ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation. I am aware that the Rochester Institute of Technology does not require registration of copyright for ETDs.

I hereby certify that, if appropriate, I have obtained and attached written permission statements from the owners of each third party copyrighted matter to be included in my thesis or dissertation. I certify that the version I submitted is the same as that approved by my committee.

Signature of Author: Karthik Chandrasekaran Date: 05/05/08

Contents

1. Problem Statement and Research Objective

The words 'Parametric & Non-parametric background subtraction model with Object Tracking for VENUS' are each explained separately below.

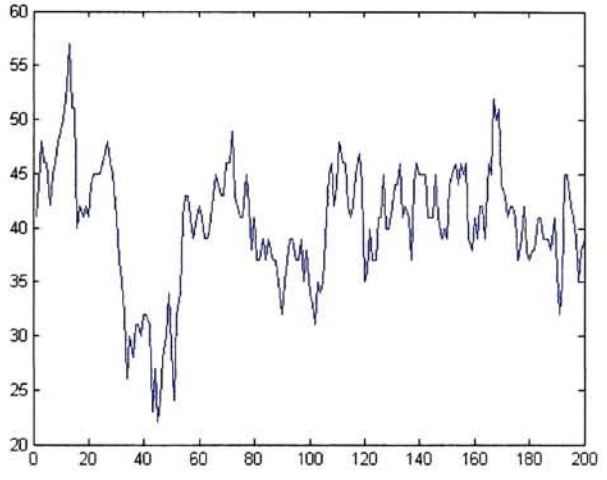
1.1 Background Subtraction

Background subtraction is the process by which we segment moving regions in image sequences. These moving regions refer only to movements in foreground objects. A background segmentation model is one that will not detect movements in the background or of objects that belong to the background. Hence the model needs to learn the static environment (such as in an indoor environment) and sometimes dynamic (outdoor) background environment in order to distinguish from movements in foreground objects. These image sequences are taken from a static camera by comparing each new frame to a model of the background scene. In the most basic example of background subtraction process, we take the mean of 'n' images which have no moving objects and then subtract from this mean image the current frame to obtain a set of values. A threshold is applied to these values to determine the background and foregrounds pixels. This is called the frame differencing technique explained in section 2.1

1.2 Parametric Background Subtraction

It is common to model the background using a Normal (Gaussian) distribution over its pixel's intensity values [1]. In a static scene, by monitoring the pixel's intensity value, we may model this value with a Normal distribution $N(\mu,\sigma^2)$. However, there is an inherent problem with this model: for non-static objects such as tree branches and leaves of plants, whose movement is dependent on the wind in the scene, the pixel intensities surrounding these objects tend to vary significantly in time. At one time a pixel may be a blue pixel corresponding to the sky region, at another, a leaf, and at another, a branch or part of a cloud and so on. Hence there are changes in the intensities of the pixel. The Normal distribution model fails to incorporate this multi-modal intensity distribution for a pixel as illustrated in the following image

Outdoor scene with a circle at the top showing the location of the sample pixel



Graph below shows the change in the intensity value of the circled pixel across 200 frames

A more generalized weighted Gaussian mixture distribution is used to model the pixel intensity [2]. For example, the pixel intensity can be modeled as the weighted mixture of three Normal distributions: sky, leaf and branch distribution. For K different modes (K is usually a small value between 3 and 5), the probability that a certain pixel has intensity x_t at time t is estimated as:

$$P_r(x_t) = \sum_{j=1}^{K} \frac{w_j}{(2\pi)^{d/2} |\sum_j|^{1/2}} e^{-\frac{1}{2}(x_t-\mu_j)^T \sum_j^{-1} (x_t-\mu_j)}$$

Where w_j is the weight, μ_j is the mean and $\Sigma_j = \sigma_j^2 I$ is the covariance for the j^{th} distribution. Initially the weights are set to be equal and the parameters of the distribution are updated recursively using a learning rate α, where $1/\alpha$ controls the speed at which the model adapts to change.

Thus the background subtraction in this method is performed with the help of setting parameters such as the learning rate detailed in section 3.1.

1.3 Non-parametric Background Subtraction

Previously, we discussed a background subtraction technique that is performed with the help of setting parameters such as the learning rate etc. In this section we will briefly discuss how we may use the Normal distribution model of pixel intensities without any parameters. The model estimates the pixel intensity probabilities independently for each frame. This model is not only faster to adapt to changes in the background process, but also detects targets with high sensitivity. This is based on the work described in [3]

The objective of this model is to accurately model the background process non-parametrically. This model should adapt faster to changes in the background process and be able to detect targets with higher sensitivity. This is done by capturing very recent information about the image sequence and continuously updating the information to capture fast changes in the scene background

As detailed in [3], we can use the Normal distribution model of pixel intensities without any parametric restrictions. The intensity of a pixel may vary significantly over time and hence we estimate the density function of this distribution at any moment of time given a recent history of pixel intensity values.

Given a sample history of pixel intensities $x_1, x_2, \ldots, x_N$, we can find the probability that the pixel value will have an intensity x_t at time t using the following equation:

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

Where K is a Normal Distribution, called the kernel estimator function and Σ is the kernel function bandwidth, which expands the above density equation to:

$$P_r(x_t) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{(2\pi)^{d/2} |\sum|^{1/2}} e^{-\frac{1}{2}(x_t - xi)^T \sum^{-1} (x_t - x_i)}$$

There are two types of variations that may occur in a pixel. On the one hand, there may be large variations due to different objects (such as leaves or branches) being projected on the same pixel at different times. On the other hand, there may be variations in intensity between the same object being projected at different times (for example, a brown branch may appear more blurred the next time it is projected onto the same pixel). The second type of variation is called the local variance which is different over different color

channels. Σ reflects this local variance in pixel intensity due to local variations from image blur and not the intensity jumps. For 'd' color channels, the above equation reduces to:

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

Using this probability estimate, a pixel is considered a targets pixel if $P_r < th$ where the threshold *th* is a global threshold that can be adjusted to achieve the desired percentage of false positives.

1.4 Object Tracking

Once the segmentation of foreground objects is complete, the next step involves the tracking of these objects from one frame to the other. While several techniques are used for tracking as explained in Section 3.3 of this document, the methodology used in this thesis work involves exploiting the functionality of Matlab's Matrix representation of each frame of the video and therefore using mathematical representations of segmented objects and their manipulation.

An object, in a video composed of a sequence of frames in temporal order, can be defined as the group of pixels that are *spatially* and *temporally* related between frames.

While working with frames of video, we do all processing at the most fundamental level of each frame, which are the pixel intensities itself. Hence at time t_1, in frame-1, it is possible to find spatially related pixels that belong to different objects, and one may then label these objects $obj_{1\text{-}1}$, $obj_{1\text{-}2}$, etc. This can be done with relatively simple processing as detailed in section 3.3 of this report. Similarly, at time t_2, in frame-2, one may find a set of new objects, and name them $obj_{2\text{-}1}$, $obj_{2\text{-}2}$, etc. However, at a higher level of processing, we know that some of the objects are the same between these two frames and have possibly only moved from their existing location between the frames. Thus it is not possible to label objects to be the same between frames by just using their location information. Other information needs to be used to *track* these objects from one frame to the other. One needs to also take into account that objects may appear and disappear between frames. This determining of *temporal* relationship between objects poses a set of challenges based on the type of information one may use regarding the *spatially* related pixels. By maintaining information about each object in a frame as a whole between frames and not just the pixel values in a single frame, one may accomplish a higher level of processing between temporally related frames. The details of this type of processing are elaborated in later sections of this report.

While various papers that define separate techniques exist for performing object tracking [6], the objective of this thesis was to use the functionality provided by Matlab along with the concepts used in background segmentation itself to help track objects.

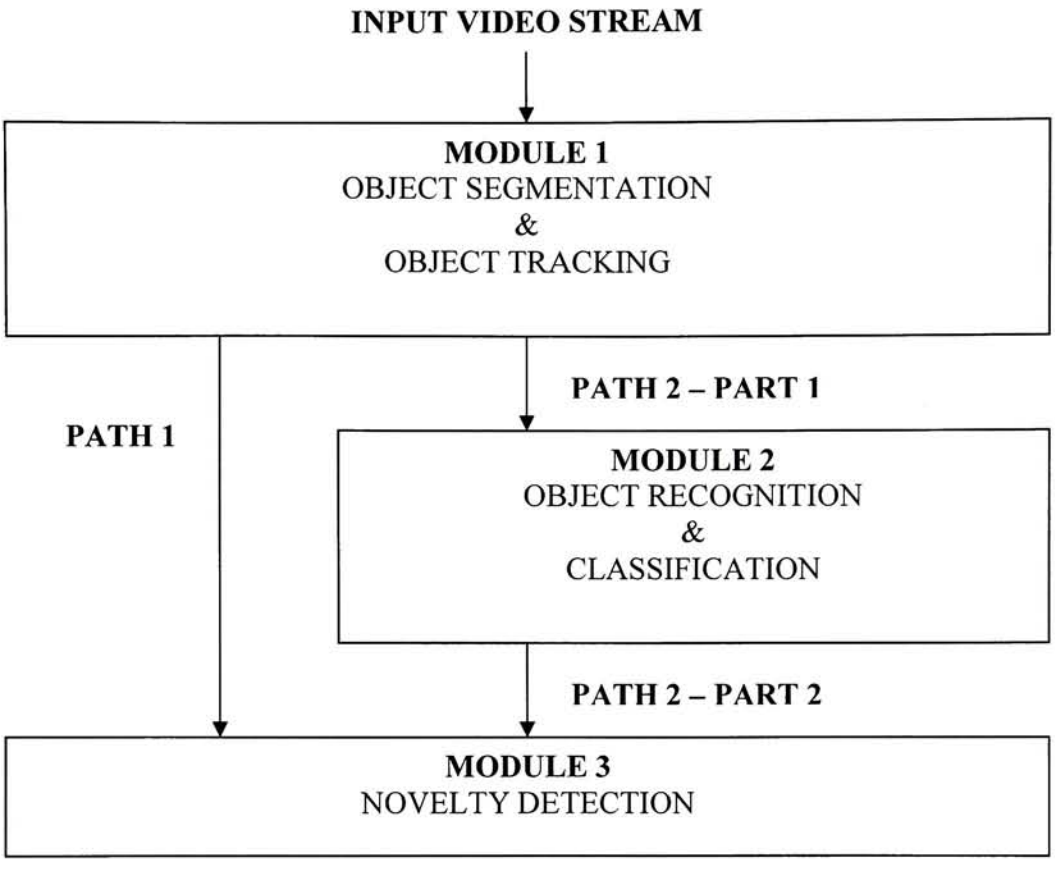
1.5 VENUS

Video Exploitation and Novelty Understanding System (VENUS) is a system for understanding novelty (or abnormal events) in streams of video [4]. The research in VENUS is currently spearheaded by Dr. Roger Gaborski. The intention of this thesis is to apply this thesis implementation code by integrating it into the existing VENUS system.

In systems that generate large streams of video date, such as in Astronomical observations, surveillance systems, ultrasound and cardiac medical imaging to name a few, it is often cumbersome and impractical for visual inspection of these streams of videos to flag abnormal of "novel" activity. An even that is not normal is classified as novel. One may define normal as an event that occurs frequently enough for it to be deemed as a consequence of habituation. VENUS exploits novelty in video with respect to the type of objects in the frame or with respect to the temporal behavior of objects. For example, if the VENUS system was to process data from a camera placed in the parking lot of a school building, it would over time, learn the type of objects such as cars, vans, trucks and bicycles that will park at this lot everyday. The system thus learns of the 'normal' objects expected in this parking lot. Now if a semi-trailer truck were to park in this lot on a certain day, the system would flag this new object as novel. This is detecting novelty with respect to objects. To explain novelty with respect to temporal behavior, consider that the system also learns that all vehicles in the parking lot tend to leave before midnight. Therefore a car or any other object that pulls into the parking lot after midnight on any day will be flagged as a novel activity.

In order to be able to process the novelty of objects at this higher level, VENUS needs to begin with processing the frames and first identifying the foreground objects in the frame along with tracking these objects between frames. This is where the research effort in this thesis and its results will be applied to the VENUS system. Further details on VENUS are presented in Section 6 of this report.

The modular architecture of the VENUS system may be explained using the following illustration:



As shown above, VENUS comprises of several modules and 2 alternate paths. In path 1, the results of object segmentation and tracking are applied directly to the novelty detection, without object recognition. This may be used for detecting novelty in temporal behavior without actually classifying the object causing the behavior. Thus a possible output of the system for a novel event may be "Object detected and tracked in parking lot at 3AM on Monday".

In the second path, we use the object recognition module to classify the objects. This may be used for a more detailed output such as "Red van detected and tracked in parking lot at 3AM on Monday". By passing output from one module as input to another module, one may replace any of the modules in VENUS with different and possibly more efficient algorithms.

1.6 Research Objectives

Based on the details provided in the previous sections, the research goals can now be specifically listed as follows:

1. Implement & evaluate background subtraction using the Parametric learning algorithm.

2. Implement & evaluate background subtraction using the Non-parametric learning algorithm.

3. Compare and select the better algorithm for object tracking

4. Implement object tracking on chosen segmentation results as input to the first phase of VENUS

2. Mathematical Background and Literature Review

2.1 Background Segmentation Techniques

As explained in earlier sections, background segmentation requires modeling a static or dynamic background accurately in order to be able to detect moving foreground objects. The simplest technique for detecting objects that have moved between frames of videos is to subtract the pixel intensity values between two frames. This difference will be zero for pixels that have not changed between frames and non-zero for any pixels that have changed between the two frames. The frames below illustrate this technique.

Frame 1 and 2 are the original consecutive frames taken from the video stream

Frame 3 shows the difference between these frames. All pixels that have not changed have a difference of zero and are therefore black in color. However, for any pixel that has changed, thereby accounting for objects that may have moved between the frames, the pixel values are non-zero and therefore not black.

The frame differencing technique detects all pixel intensities that have changed. Some of these may be due to moving foreground objects like the people walking in the above frames, and some of these could be due to objects moving in the background, such as clouds, leaves of trees, or from the varying intensity of sunlight on any background object. Such detections are undesirable since the objective of our system is to detect moving foreground objects. In order to not detect movements in the background, one has to model the background over time to be able to suppress movement in the background thereby distinguishing this movement from foreground object movement. Several techniques exist for modeling the background.

A common technique to model the background is to use a Normal (Gaussian) distribution over the pixel intensity values in the background environment [1]. In a static scene, by monitoring the pixel's intensity value, we may model this intensity value with a Normal distribution $N(\mu,\sigma^2)$. However, as described earlier this model fails to incorporate this multi-modal intensity distribution for a given pixel. A more generalized weighted mixture Gaussian distribution is used to model the pixel intensity [2]. For example, the pixel intensity can be modeled as the weighted mixture of three Normal distributions: road, car and shadow distribution for a pixel on the ground. For K different modes

(K is usually a small value between 3 and 5), the probability that a certain pixel has intensity x_t at time t is estimated as the weighted sum of K Gaussians. Initially the weights are set to be equal and the parameters of the distribution are updated recursively using a learning rate α, where $1/\alpha$ controls the speed at which the model adapts to change.

This *parametric learning model* constitutes the initial research of this thesis. The details of the implementation of this model are presented in section 3.1 of this report. As will be elaborated in later sections, the problem in the above approach is that the learning rate makes the model update slowly to changes leading to a 'ghosting effect'. This leads us to the second part of this thesis research which investigates a *non-parametric model* for segmentation. As will be elaborated in later sections, the non-parametric model is highly sensitive to object movements and does away with 'ghosting'. However, this high sensitivity poses its own set of related challenges.

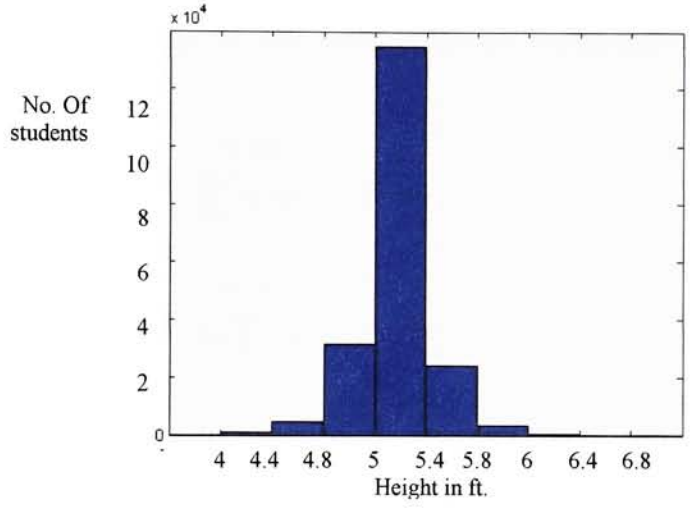
Before we delve into the details of the parametric approach, it is important to understand the universal applicability of the Normal (Gaussian) Distribution as the equation of choice for modeling background environments. The next section of this report attempts at introducing the reader to the concept of Gaussian.

2.2 The Gaussian Model

In earlier sections of this report, we mentioned that a 'common' model of choice for background environments is the Normal (Gaussian) Distribution. One may wonder what properties of this distribution make it the universal model of choice for our work. In order to understand this, the following section provides a statistical and biological background into the Normal Distribution.

Histograms

If you measure the heights of all the students in a classroom and count the number of students with each height, then the resultant set of counts can be used to construct a histogram as follows:



A histogram is thus, a graphical representation of such counts. As we want to count the number of students with different heights, we need to divide the range of measured heights into a number of intervals called bins. In the above histogram, the bins have been created with a width of 4 inches. For each bin, we count the number of measured heights that fall between the lower and upper bound of that bin. We can expect few measurements to fall in the first bin between the heights of 4 and 4.4 ft and the last bin between the heights of 6.4 and 6.8 ft., because both these values lie at the extreme ranges of human heights. Conversely, we can expect a large proportion of measurements to fall in the bins of 5 to 5.4 ft as this range corresponds to a common human height.

In a histogram, the *i* th bin with bounds $[x_i , x_i + \Delta x]$ provides an estimate of the number n_i of measured values that fall between x_i and $(x_i + \Delta x)$, for example between 4 and 4.4ft.

If the histogram is based on N measurements partitioned into M bins, then the estimated probability that x falls within the interval defined by the *i* th bin is equal

to the area occupied by the *i* th bin expressed as a proportion of the sum of areas occupied by all the M bins. The area of a bin is defined as the product of its height times its width, which in our example is n_i times Δx. Therefore the sum of the areas of M bins is therefore:

$$= \quad n_1 \Delta x + n_2 \Delta x + n_3 \Delta x +, \ldots, + n_M \Delta x$$

$$= \quad (n_1 + n_2 + n_3 +, \ldots, + n_M)\ \Delta x$$

$$= \quad (\sum_{j=1}^{M} n_j)\Delta x$$

Where $\sum$ represents the summation over M bins. Thus the probability that x falls in the *i* th bin. As mentioned earlier, is equal to the area occupied by the *i* th bin expressed as a proportion of the sum of areas occupied by all the M bins. Mathematically, this is:

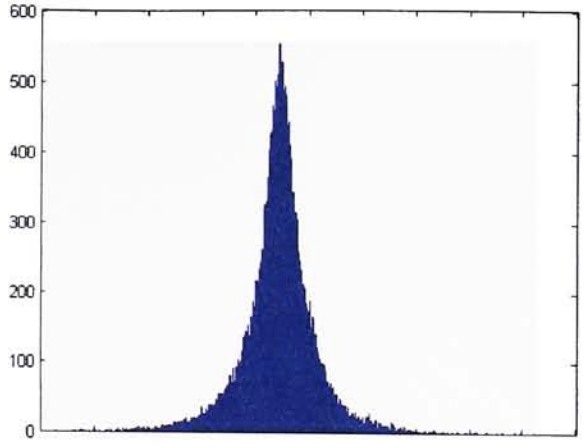
$$p(x) \quad = \quad \frac{n_j \Delta x}{\sum_j n_j \Delta x}$$

$$= \quad \frac{n_j \Delta x}{\Delta x \sum_j n_j}$$

$$= \quad \frac{n_j}{N}$$

Probability Density

If we were to make the sum total of all the bins to be equal to unity, the above equation will represent the probability density. The fact that the number of samples (students) is small makes the histogram appear lumpy. If we were to increase the number of students to a very large number we can decrease the width of the bins, the resultant histogram approaches a bell shape as illustrated below:



In the limiting case, as the bin width approaches zero, the height of each bin approaches the probability density and the shape of the histogram of x values approaches that of the probability density function (PDF) of the variable x, denoted by $p_x(x)$. This establishes the relationship between the PDF and the histogram. The histogram is a good approximation to the normal or a Gaussian PDF.

Gaussian Probability Density Function

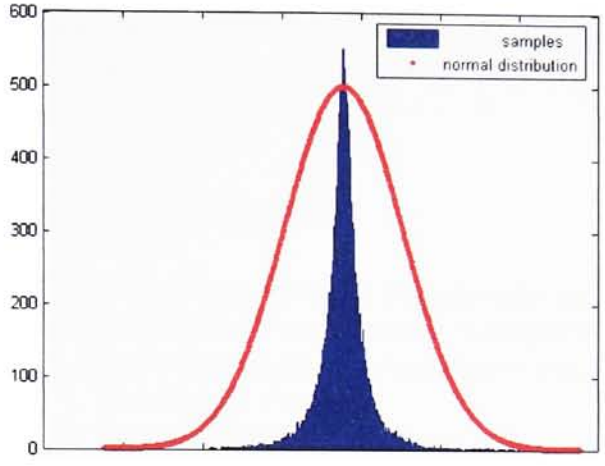
The reason for choosing a Gaussian PDF as the model of choice for our non-parametric background model is explained below.

A histogram in the limiting case is a good approximation of the Gaussian PDF. In our previous example of the histogram of measured heights of students, the measured quantity (heights), is determined by several variables such as genetic disposition, food habits, exercise, etc. This is not a co-incidence. Almost any measured quantity which depends on several underlying factors has a Gaussian PDF. This is the essence of the Central Limit Theorem (CLT)

Central Limit Theorem (CLT)

Human height depends on many underlying factors. CLT ensures that if we could make a histogram of the amount of each factor across all individuals in a population than the shape of the histogram for each factor would not significantly alter the Gaussian distribution of heights in the population. In other words, the Gaussian distribution is nature's default distribution if many factors contribute to a given physical attribute, be it height or weight.

The shape of the normal PDF is as shown below. The equation for the Gaussian PDF is



$$p_x(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e[\frac{-(x-\bar{x})}{2\sigma^2}]^2$$

Where σ is the standard deviation, which is a measure of the variability of $\overline{x}$, and x is the mean of x. The first term ($1/\sqrt{2\Pi\sigma^2}$) is a normalization constant which ensures that the sum of the areas under the normal PDF results in unity.

The Central limit Theorem is stated as follows:

If a set of quantities **x** = (x_1, x_2, ..., x_M) are independent with means (μ_1, μ_2, ..., μ_M) and variances (σ_1^2, σ_2^2, ..., σ_M^2) then, for a large number M of quantities **x,** the quantity

$$q = \sum_{j=1}^{M} s_j$$

has a PDF which is approximately a Gaussian with mean $\sum_j \mu_j$ and variance $\sum_j \sigma_j^2$

2.3 Density Estimation

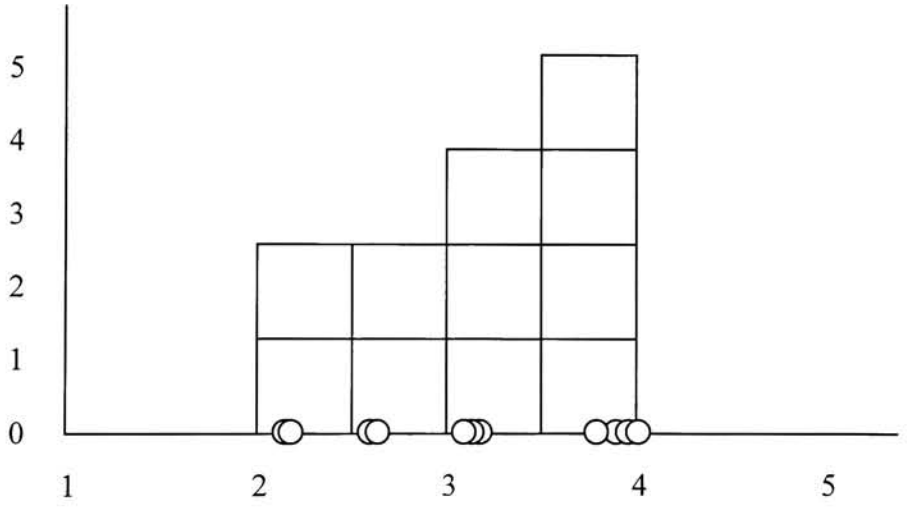
Density Estimators are a generalization and improvement over histograms. In order to understand why they are an improvement over histograms, one needs to first study the properties of the histogram in order to compare them to that of the kernel density estimator. Finally, we will study how to select an appropriate kernel to extract and reveal all the important features of the data.

Properties of Histograms

The simplest form of the density estimator is the ubiquitous histogram. The details of how to construct a histogram have been presented in the earlier section. When constructing a histogram, we need to consider two main points:

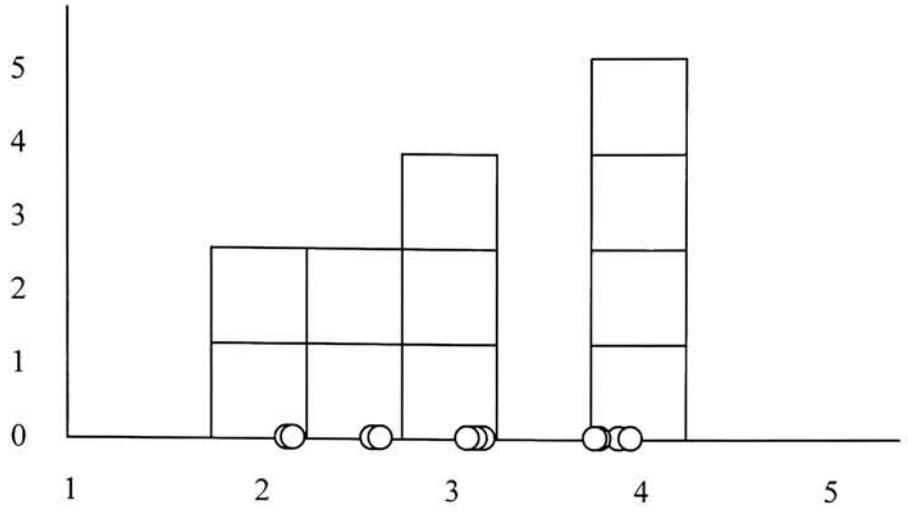
1. The size of the bins
2. The end points of the bins

To illustrate this example, consider that every time a data point falls within a bin, a block, of size equal 1 by the bin-width, is placed on top of it as shown below:



In the above histogram, we choose breaks at 0 and 0.5 and a bin-width of 0.5, and the data end points are represented by circles on the x-axis. The above histogram appears skewed to the right and is unimodal.

The choice of these end points is important. For example, if we use the same bin-width but with the end points shifted up to 0.25 and 0.75, then out histogram will change as follows



The estimate of density is now different. It appears bimodal. Therefore, it is clear to see that density estimation using histograms is limited by the following properties:

1. The graph is not a smooth one
2. Depends of the bin-width
3. Depends on the bin end points

We can reduce the dependency on the bin-width by changing the width to be very small. If we use a normal (Gaussian) kernel with bandwidth or standard deviation of 0.1 (which has area 1/11 under the each curve for each of the 11 data points) then the kernel density estimate will be a smooth curve. In order to choose an optimal bandwidth value, the following expression should be evaluated:

$$\text{optimal bandwidth} = \arg\min \text{AMISE}$$

i.e. optimal bandwidth is the argument that minimizes the Asymptotic Mean Integrated Squared Error

AMISE depends of the underlying density and since we don't have that density value as yet, we need to estimate the AMISE from our data. This means that the chosen bandwidth is an estimate of an asymptotic approximation.

Thus the properties of kernel density estimators as compared to histograms are as follows:

1. Can be smoothened
2. And thus have no end points
3. Depend on the bandwidth value

With respect to our research, kernel density estimators are used for estimating the probability density function of random variables. Statistically, when given information about a sample of a population, density estimators extrapolate information about the entire population.

As seen in section 3 of this report, kernel density approximation of the probability density function of a random variable is defined as:

$$f(x) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h} \right)$$

Where K is the kernel and h is the bandwidth.

For our experiments, we have chosen the K to be the Gaussian function given by the equation:

$$K(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{y}$$

$$y = -\frac{x^2}{2\sigma^2}$$

3. Implementation Details

3.1 Parametric Subtraction Model

Our goal is to segment non-background objects in video stream taken from a stationary camera. As seen in section 2.1 a number of systems perform segmentation of moving objects by image differencing with various enhancements [8] by thresholding the absolute difference between consecutive frames, a binary mask of moving pixels is obtained. The binary mask is then segmented using connected component labeling. As demonstrated in previous sections, this method is sensitive to noise in the background and changing light.

Parametric Subtraction model uses an adaptive, statistical background model to detect changes in the scene. Detection is done through dynamic learning of the background and segmentation of occluded regions. The background is modeled using a multi-dimensional Gaussian in HSV (Hue Saturation Value) color space. During the processing of each frame, the intensity value of a pixel in the frame is compared to the current distribution in order to either classify the pixel as background or an occluded element. A connected component is then formed from all the occluded pixels thereby forming a group of foreground objects in the scene. A group of pixels whose size is below a certain threshold is then discarded as a false positive. This background Gaussian distribution is then updated with the information from the current frame to account for any changes within the scene.

Hue Saturation Value (HSV) over Red Green Blue (RGB)

The background pixel is modeled using a Gaussian distribution. The red, green and blue (RGB) intensity values of a pixel can each be modeled separately with a Gaussian to improve the accuracy of segmentation [7], however working in RGB space does not allow modeling of camera-created artifacts that occur in high contrast areas, such as shadows and hue inconsistency.

The HSV color space allows for clear separation of the intensity (V) and chromatic information (HS) of pixels. A background pixel is thus a vector of 3 variables H(x,y), S(x,y) and V(x,y) each characterized by a mean μ and standard deviation σ. During initial processing, the mean value is initialized using the values from the first frame. Standard deviation is set to zero for the first frame. And the Gaussian distribution is formed with subsequent frame values.

Decision Logic

In order to decide if the current changes pixel is an observation of the background or a foreground object, one must use the intensity and chromaticity information as follows:

If the Hue, Saturation and Value fall within two standard deviations of the mean value of the component in the corresponding model distribution, the pixel is said to be and observation of the background

After the foreground pixels have been detected, the current distributions are updated to incorporate the latest information. If x denotes the current observation for a pixel, the background model is updated according to the following laws:

Update Mean

$$\mu = (1 - \alpha)\,\mu + \alpha\, x$$

Update Standard Deviation

$$\sigma^2 = \max\left(\sigma^2_{min},\ (1 - \alpha)\,\sigma^2 + \alpha\,(x - \mu^2)\right)$$

α is the learning rate

σ^2_{min} is a minimum standard deviation that acts as a noise threshold that prevents the standard deviation from decreasing below a minimum value if the background remains unchanged for a period of time.

Note on computation:

Matlab facilitates the easy conversion of a RGB frame to HSV color space and the subsequent processing of each of the H, S and V dimensions. This can be done with the following simple Matlab commands:

```
fhsv = rgb2hsv(frame);

H(:,:,a) = fhsv(:,:,1);
S(:,:,a) = fhsv(:,:,2);
V(:,:,a) = fhsv(:,:,3);
```

3.1.1 Problems with the Parametric Model:

The use of parametric learning poses various problems:

1. Because of the use of a learning parameter α, the learning process is slow leading to a ghosting effect as illustrated in the frames below.

2. In addition to being slow to update, the model is also insensitive to small changes leading, at times, to false classification of pixels into background and foreground objects

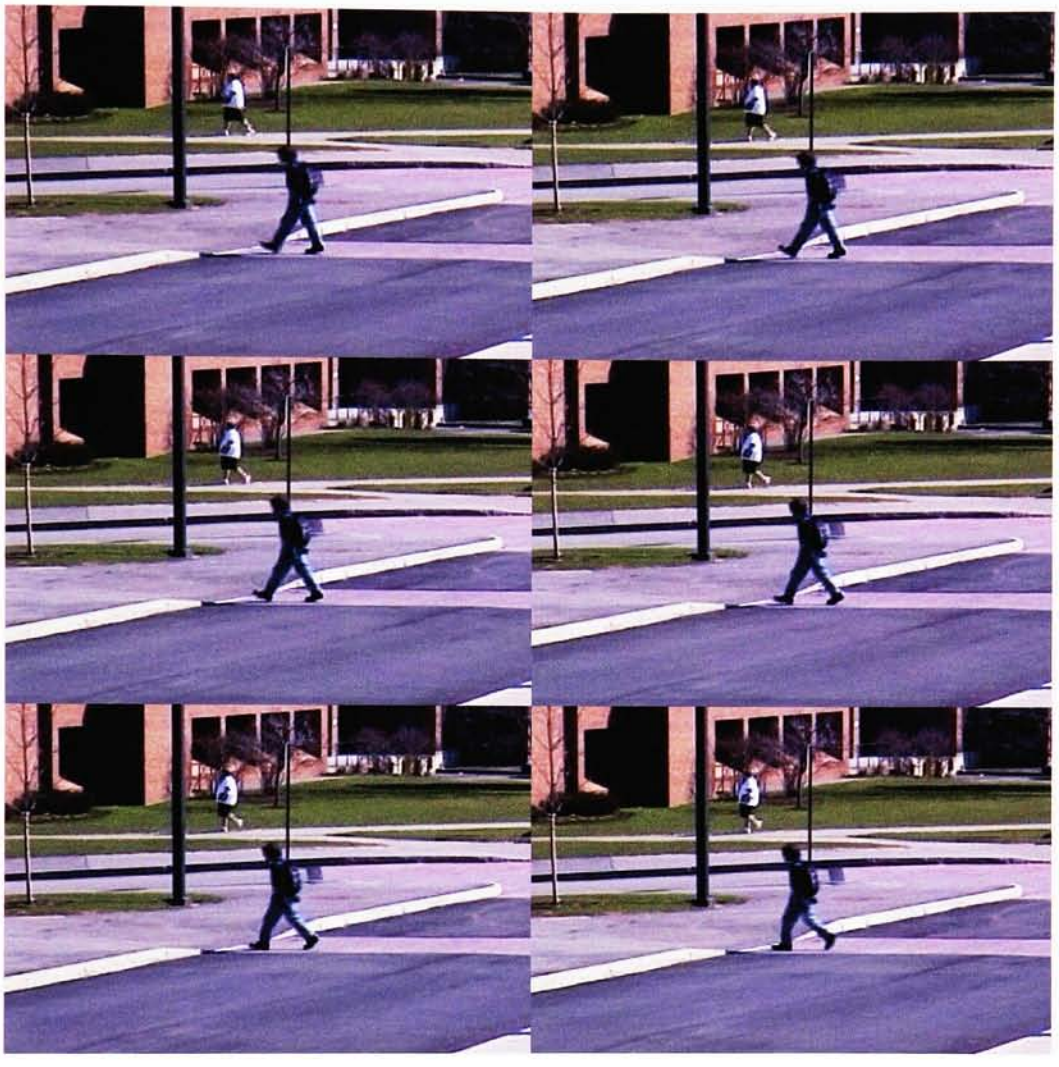
3. Since the background needs to be updated with every frame, processing is CPU intensive and slow.

Ghosting effect

In order to understand the ghosting effect, consider the following example. Let pixel *p* (x, y) be a part of the background model in frame number 20 of the input video. Now, if a foreground object were to occlude this pixel between frames 21 to 24, the background model will update to reflect that pixel *p* is now a foreground pixel between frames 21 to 24. At frame 25, when *p* is no longer occluded by a foreground object, the background model must learn that *p* is again part of the background. However this phase of learning that a foreground pixel is now a background pixel is much slower. As a result of this, the pixel remains marked as a foreground pixel even after frame 24, probably all the way up to frame 27. As a result of this, a region of the background trails every foreground object until these pixels are learnt to part of the background again. This is the 'ghosting' effect that leads to false detection of foreground objects in the video.

The frames in figure (a) show the original video frame sequence to show the exact region of movement of the foreground objects.

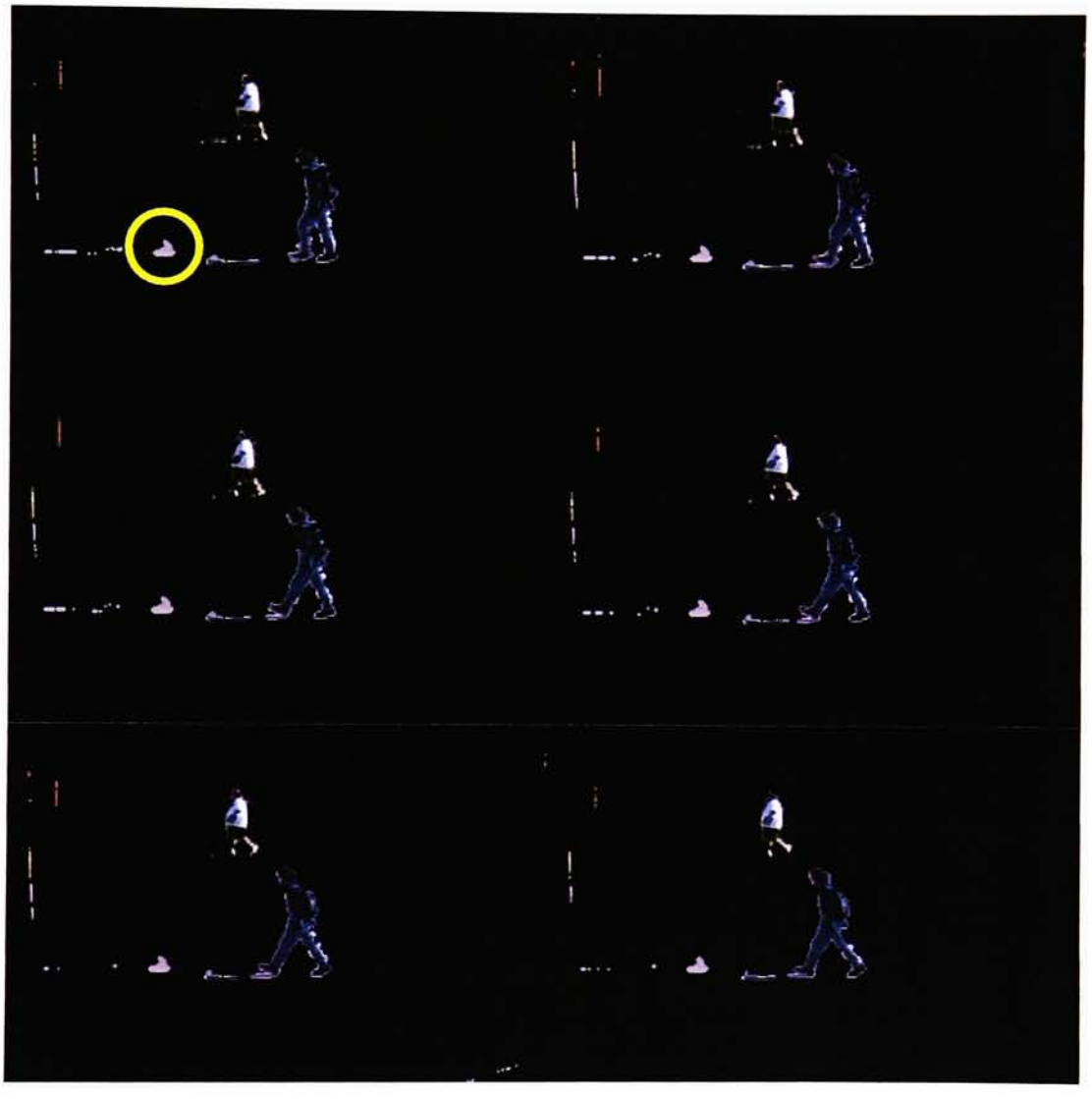
Frames 1 through 8

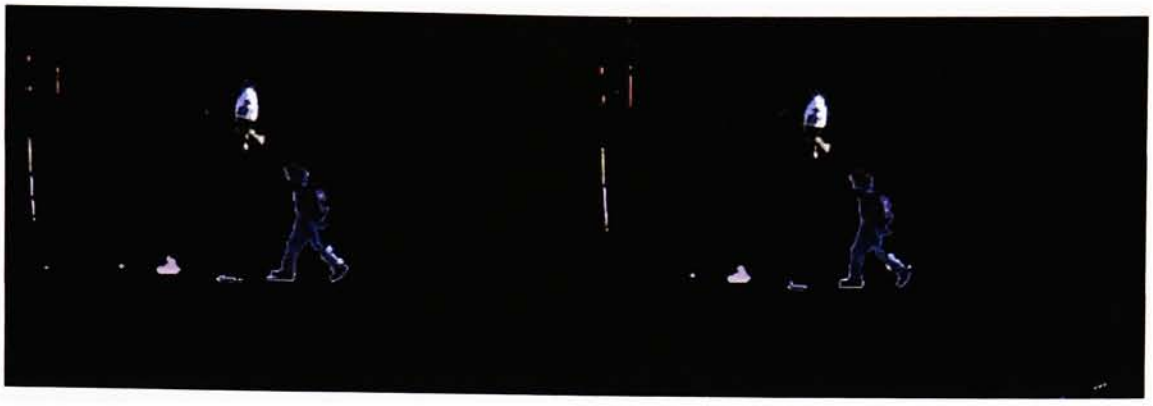


Frame 30…

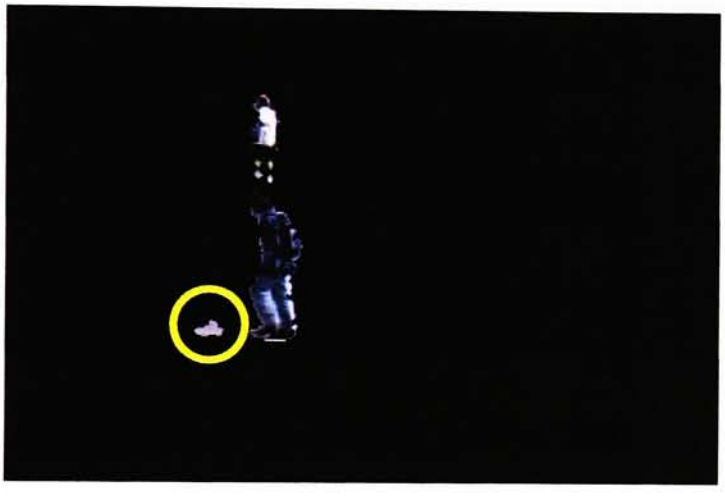
The following are the same frames after applying parametric learning for background segmentation. There are many falsely detected foreground objects during initialization of the distribution. While some are learnt and classified as background pixel in subsequent frames, one of the objects (circled in yellow only in frame1) remains to be falsely detected for the first 30 frames as shown below:

Frames 1 through 8



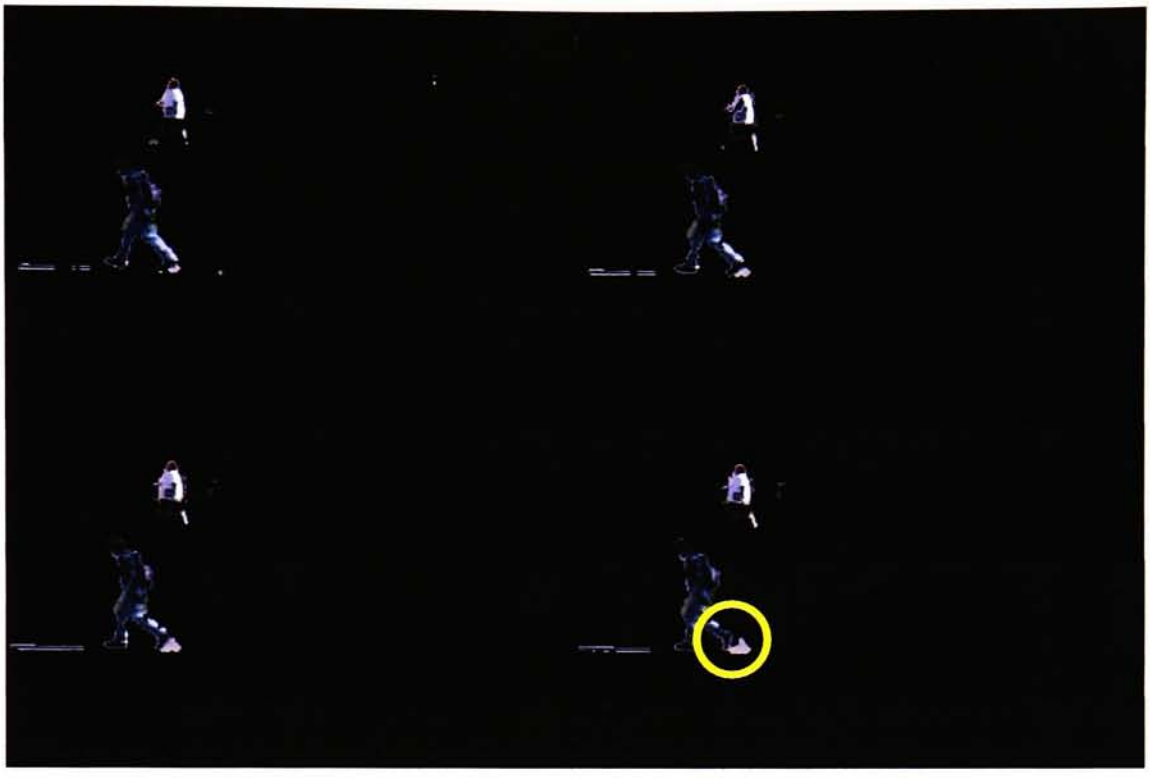


Frame 30...



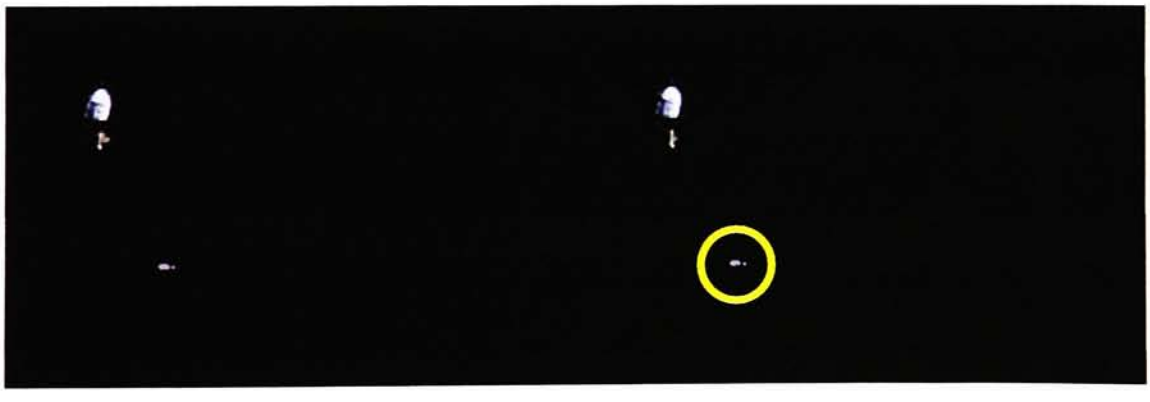
Note that the passing of a foreground object across this region falsely re-enforces its probability of being a group of foreground pixels and now the background model has to again learn that these pixels belong to the background.

Frames 50 through 55



This region is only learnt to be part of the background after frames 99 and 100. This retention of falsely classified foreground objects leads to a ghosting effect in the frames during segmentation:

Frames 99 and 100:



3.2 Non-Parametric Background Subtraction Model

The objective of this model is to accurately model the background process non-parametrically. This model should adapt faster to changes in the background process and be able to detect targets with high sensitivity. This is done by capturing very recent information about the image sequence and continuously updating the information to capture fast changes in the scene background.

We have already established the reason for choosing the Gaussian distribution to model the background. The first phase of this thesis uses a parametric Gaussian model as detailed in section 3.1. The shortcomings of this parametric model with its use of a learning parameter α led to the second phase of this thesis, namely modeling the background with a non-parametric Gaussian model. Just as in the parametric model, we work with a multi-dimensional Gaussian distribution for each of the color channels. Using 3 distributions to model a pixel's intensity value improves the overall sensitivity of the segmentation process. It is common to use a single Gaussian distribution [1]. This basic Normal distribution adapts to changes rather slowly and is easily prone to false detections due to illumination changes and camera artifacts etc. The basic Normal model can be update using a simple adaptive filter such as a Kalman filter [2]. Since pixel values can be multi-modal (as described in section 1.2), the next type of common model to be used involved using multiple Gaussian distributions, one for each mode. Thus to model a background pixel that changes from being having the color intensity of a blue sky to a brown branch of a tree or a green leaf on a branch, one would require 3 Gaussian distributions to model the pixel's intensity values. Depending on how much each of these modes contribute the pixels intensity value (for example, if the pixel tends to be a sky pixel for most of the video and a branch pixel in only some of the frames) the individual Gaussian distributions representing each mode is weighted by a factor and the sum of these weighted Gaussians is used to determine if the current value is a foreground or background pixel. The equation for this combined distribution is presented in section 1.2 of this report. In the case where the background has very high variations in pixel intensities, this model fails to achieve sensitive detection. Also these models are slow to adapt to changes in the background which leads to the construction of a very wide and inaccurate model that will have low detection sensitivity.

In our model, in order to increase the sensitivity, we must estimate the density function of our Gaussian distribution (section 2.3) at any moment of time using only the most recent history information. Given a sample history of pixel intensities $x_1, x_2, \ldots, x_N$, we can find the probability that the pixel value will have an intensity x_t at time t using the following equation:

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

Where K represents the kernel density estimator namely, the Gaussian Distribution (or Normal function),
Σ is the kernel function bandwidth

The details of kernel density estimators and the importance of the approximation and choice of the kernel bandwidth are detailed in section 2.3.

Substituting for the Gaussian distribution to represent the kernel density estimator K in the above equation, we get the following expanded Multi-dimensional Gaussian Distribution to model the background:

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

Where x_t represents the pixel x at time t
P_r represents the Probability value of pixel x
N is the size of the recent history (the number of frames in history)
d represents the number of color channels (RGB or HSV)
σ is the standard deviation of the pixel intensity values

Π in the above equation represents the product of different distributions across d color channels, more specifically in the case of RGB images, the product of 3 Gaussian Distributions.

Σ in the above equation represents the sum of this resultant product across N frames of the history

Using the above probability estimate, a pixel is considered a targets pixel if:

$P_r < th$ where the threshold *th* is a global threshold that can be adjusted to achieve the desired percentage of false positives.

Kernel Width Estimation for the Non-parametric model

The non parametric model accounts for two types of variations that may occur in a pixel intensity value. On the one hand, there may be large variations due to different objects (such as leaves or branches) being projected on the same pixel at different times. On the other hand, there may be variations in intensity between the same object being projected at different times (for example, a brown branch may appear more blurred the next time it is projected onto the same pixel). The second type of variation is called the local variance which is different over different color channels. Σ reflects this local variance in pixel intensity due to local variations from image blur and not the intensity jumps.

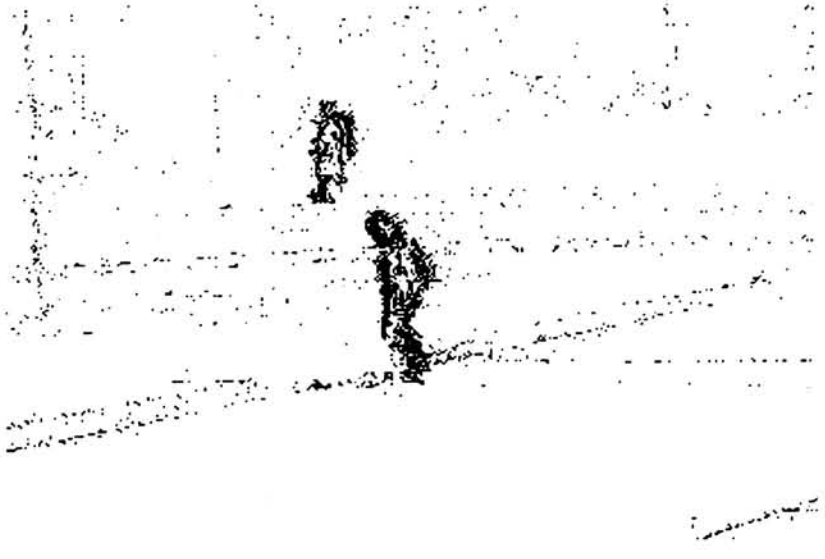
In paper [3] to estimate the kernel band width σ_i^2 for the *i*-th color channel for a given pixel we compute the median absolute deviation over the sample for consecutive intensity values of the pixel. In other words, the median, *m*, of the $|x_i - x_{i+1}|$ for each consecutive intensity pair (x_i, x_{i+1}) in the sample, is calculated independently for each color channel. The standard deviation for the first distribution is estimated as

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

3.2.1 Suppressing False Detection

After the background has been modeled using the above density estimation process, small movements in the scene background may cause false detection. For example, if a tree branch moves farther than it did during model generation; the algorithm will incorrectly detect it as a target pixel. It can also be caused due to small camera displacements. If some part of the background moves to occupy a new pixel, but was not part of the model for that pixel, then it will be detected as a target object.

Original Frame (above) and Frame after segmentation with false detections (below)



Original Frame (above) and Frame after segmentation with false detections (below)

3.2.2 Component and Displacement Probabilities

If some part of the background moves to occupy a new pixel, but was not part of the model for that pixel, then it will be detected as a target object. This falsely detected object however, will have a high probability to be part of the background distribution at its original pixel. If we consider a small neighborhood of the pixels that belong to this background object, then we can determine for certain if the change in pixel intensity has been caused by a background object. Hence the maximum probability that the observed value x_t, belongs to the background distribution of some point in the neighborhood N(x) of x is given by the pixel displacement probability $P_N(x_t)$:

$$P_N(x_t) = \max_{y \in \mathrm{N}(x)} P_r(x_t \mid B_y)$$

Where B_y is the background sample for pixel y. This does eliminate false detections however it also introduces false negatives (eliminates true detections) since some target pixels may be similar to the background of some nearby pixel. In order to avoid this, we make sure that we know if the entire target object has moved from its nearby location and not just some of its pixels. For this we use the component displacement probability P_C:

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

Hence a pixel will be considered a background pixel only if

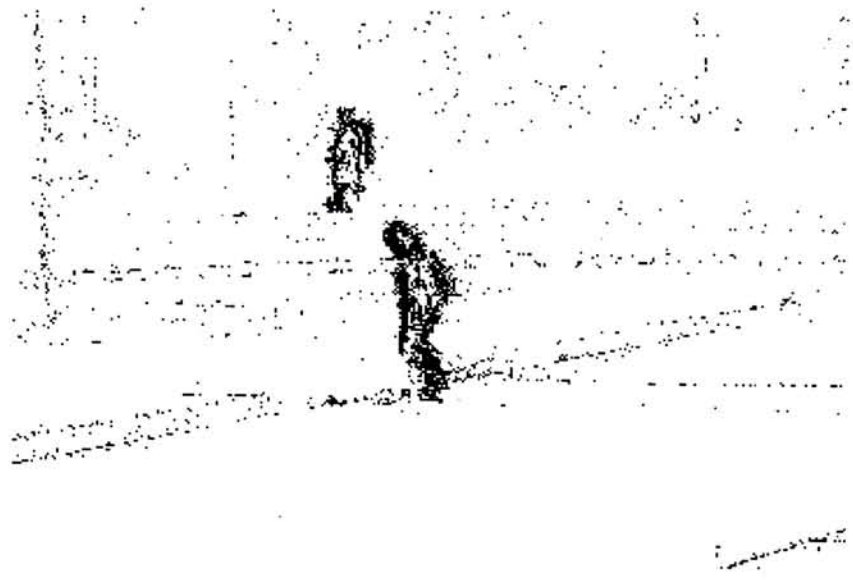
$$(P_N(x) > th_1) \wedge (P_C(x) > th_2)$$

Where th_1 and th_2 are threshold values that determine the percentage of false detection.

> *Thus if the displacement probability $P_N(x_t)$ of pixel x is below the threshold th_1 and the component displacement probability P_C is also below a second threshold value th_2, then the pixel x appears to be a background pixel*

The following figure clearly show the effects of b) density estimation P_r, c) pixel displacement probability P_N and d) component displacement probability P_C

Frame after segmentation with false detections (above) and after thresholding the component and displacement probabilities (below)



Frame after segmentation with false detections (above) and after thresholding the component and displacement probabilities (below)

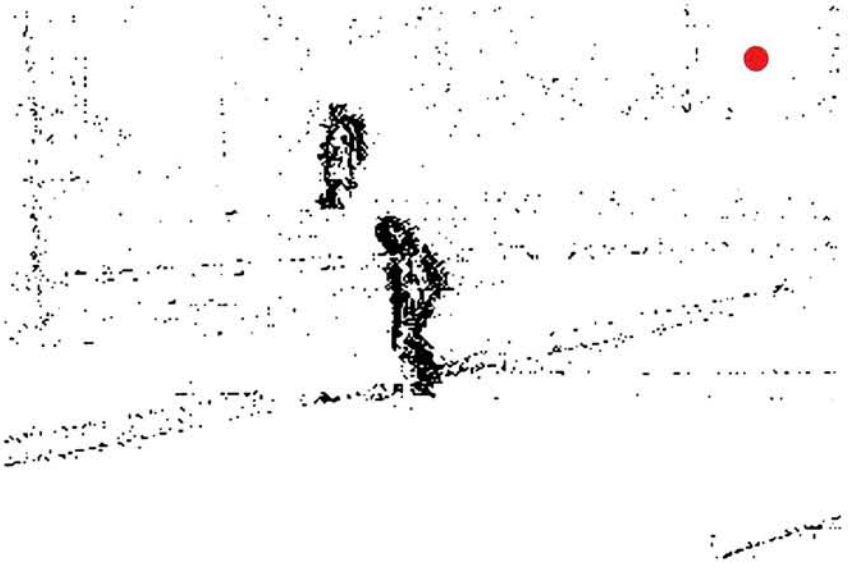
Morphological Processing

At this stage of results discussion it is important to differentiate between using Matlab's image processing toolbox to suppress false detections versus using the component and displacement probabilities as explained in the previous section

The operations of dilation and erosion are fundamental to morphological image processing. Dilation is the operation of growing or thickening objects in a binary image. The extent to which objects can be thickened can be controlled by using a shape called the structuring element. Erosion, on the other hand, shrinks or thins objects in a binary image. Again we can control the extent of shrinking using a structuring element.

By combining the operations of Dilation and Erosion, we can 'open' or 'close' objects in an image. The frame below shows how a structuring element, the size of the red circle in the image, may be used to remove false detections to achieve the desirable level of segmentation in each frame of the video.

However, the dangers of using such morphological processing lie in the fact that desirable true foreground objects whose size is smaller than that of the structuring element may also be removed during the processing. Thus using the combined probability values (as detailed in the previous section) is a better approach at suppressing false detections.



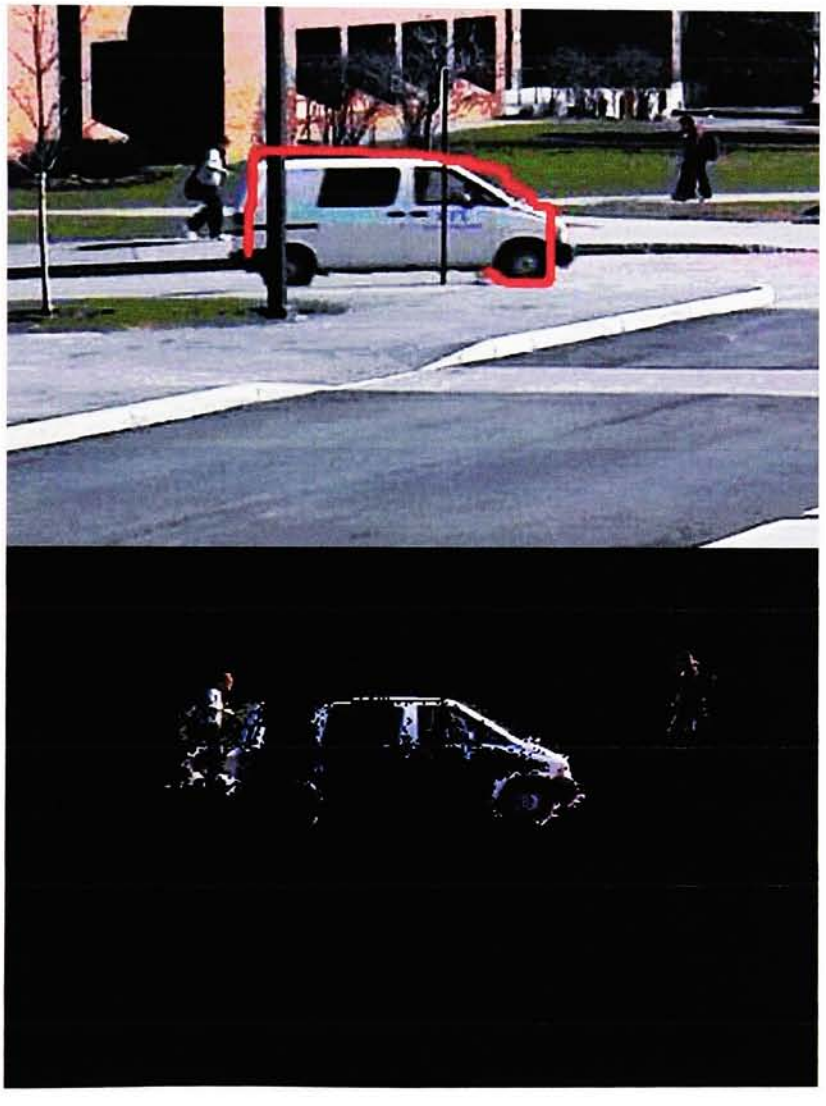
Updating the Background

The sample needs to be updated continuously to adapt to changes in the scene. The update can be performed in two ways 1) Selective Update 2) Blind Update. In selective update, the sample is added to model only if it has been classified as a background sample. Obviously, this adapts to changes more quickly and is also more sensitive since target pixels are never added to the model. However, if there is an incorrect detection decision made, the incorrect result will persist throughout the model. The blind update, we simply add the sample to the model. However, since intensity values that do not belong to the background model are added, this may lead to false negatives or bad detection of the targets as they erroneously become part of the background model. Also this model is slower to adapt to changes in the background. The hybrid approach involves using the intersection of the results of the two background update models.

3.2.2 Problems with Non-parametric learning

Problem 1: Sensitivity

As seen in the previous section component & displacement probabilities reduce the number of false detections in each output frame leaving behind the segmented object. In the case of larger objects such as the blue van in the sequence of frames below the high sensitivity of the algorithm to movements within an object leads to poor detection of these large objects due to large regions of pixels within the object that are not moving.



In the above frame, when the regions of the van marked in red are detected correctly as dynamic foreground pixels. However due to the nature of the sensitivity of the algorithm the regions beyond this red area within the van are falsely detected as background regions within the foreground object. In other words the algorithm adapts to the foreground region rapidly enough to being

classifying regions that are not dynamic any longer (for a sequence of frames) within this foreground object as background pixels.

One may increase the number of frames to be used within the window used as the reference history for classification by the algorithm to reduce the sensitivity of detection to overcome this problem. The objective is to use an optimal number of frames that will provide the best detection results without making the mathematical computations infeasible.

In this thesis, the number of frames within the window of reference was set to 100.

Problem 2: Artifacts

As seen in the frame below, there are a number of artifacts that are introduced during the recording of the input video due to tiny movements in the stationary camera that are falsely detected during background segmentation.

These undesirable moving regions can be removed from the output using object tracking knowledge as explained in the following section.

3.4. Object Tracking

For object tracking in a video composed of a sequence of frames in temporal order, we need to keep track of groups of pixels that are both *spatially* and *temporally* related between frames.

In the 'Moving Object Tracking in Video' paper [12], objects are tracked with a rule-based algorithm using the information of the object trajectories, sizes, grayscale distribution and textures. The variables for each of these properties are first calculated and the objects are tracked based on these variable values. The variables for object trajectories are the object position co-ordinates. Object position is determined using the centroid of the object. Each object is then tracked as a single point representing the object. Object trajectories are assumed to be close to straight lines and the object acceleration rate is also assumed to be a constant between frames. Herein lays the problem with applying this technique to our segmentation results.

While the non-parametric background model is more accurate than the parametric model in segmenting objects, this model is over-sensitive to changes in the frame. In order to explain this, consider the following segmentation results from the non-parametric model:

Original Frame

Frame after non-parametric segmentation

In the frame above, the van is fairly large in size. Due to the acute sensitivity of the algorithm the following problem occurs: when the van moves over a background pixel, the algorithm correctly detects the pixel to now be part of the foreground. However, even after a few frames, since the van is a long object, the pixel remains to be part of the foreground van object. The algorithm now falsely detects the pixel to be part of the background as there has been no change in its intensity value for a few frames now as the van is still moving over the region.

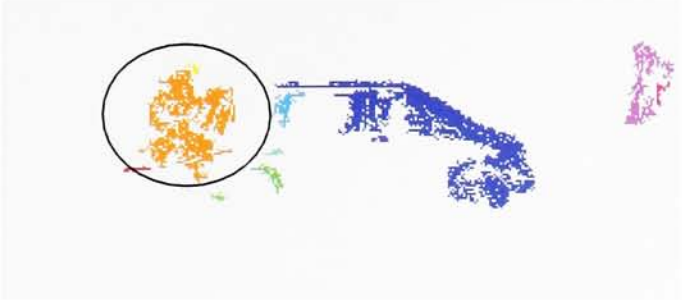
Thus the segmentation process incorrectly detects background within the foreground objects if the foreground object is fairly large in size.

bwlabel() function:

This Matlab function is very useful in grouping pixels that are spatially connected.

- L = BWLABEL(BW,N) returns a matrix L containing labels for the connected components in BW.
- N can have a value of either 4 or 8, where 4 specifies 4-connected objects and 8 specifies 8-connected objects
- The elements of L are integer values greater than or equal to 0.
- The pixels labeled 0 are the background. The pixels labeled 1 make up one object, the pixels labeled 2 make up a second object, and so on.
- [L,NUM] = BWLABEL(BW,N) returns in NUM the number of connected objects found in BW.

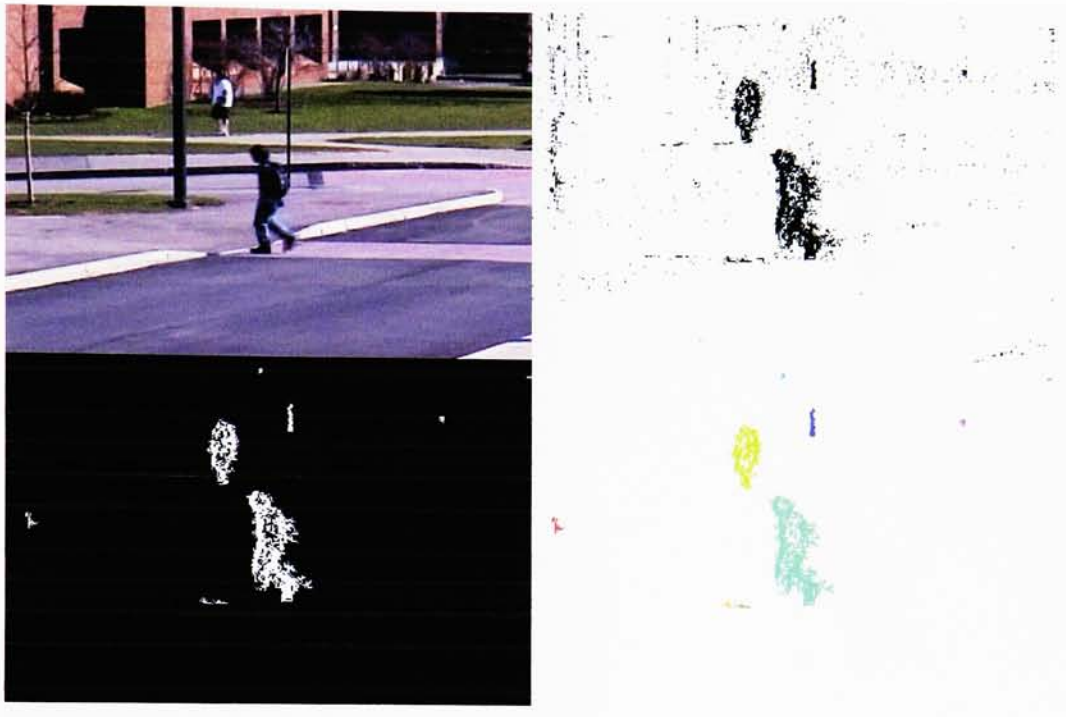
If we labeled the objects in this frame by grouping spatially connected pixels, the van itself will be represented as various separate objects. This is illustrated in the frame below which shows different spatially related objects labeled by color.



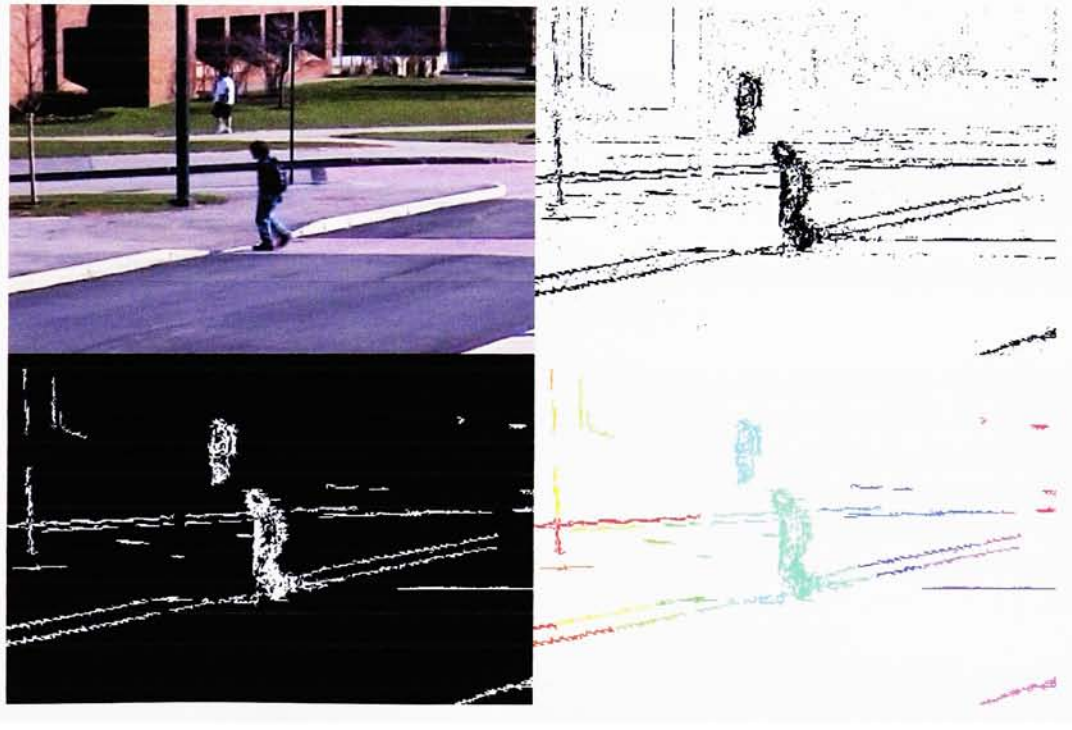
Overcoming the difficulties of segmenting and tracking correctly the object in the above frame, is a vital part of the object recognition algorithm used in this thesis research. In the above frame not only is the van segmented as various unrelated objects, but also once of these objects that belong to the van is incorrectly labeled together with the moving person as one object.

Clearly just the property of spatial connectedness is insufficient to group pixels into objects. As we see later, we use other properties of an object to help correct these errors.

In addition to the above problem due to over-sensitivity of the non-parametric algorithm, the segmentation results are sometimes incorrect due to artifacts in the frames introduced by slight movement of the camera capturing the video. This is illustrated in the frames below:



Frame 11



Frame 12

Frame 13

The above frames are each a combination of 4 frames:

Frame at the top-left is the original input frame
Frame at the top right in the results from segmentation before suppression of false detections (section 3.2.1)
Frame at the bottom left is the results after suppression of false detection (section 3.2.2)
Frame at the bottom right is the results of grouping pixels based only on their spatial orientation

Between frames 11 and 13, frame 12 introduces a large number of artifacts due to a slight movement in the stationary camera capturing the video. Such artifacts have to be removed from the segmentation results in order to correctly identify only the true foreground objects. The simple fact that these artifacts do not exist in either the previous or the next frame can be used to remove them from the segmentation process. Thus using this temporal relationship between the frames, it is possible to rid of camera induces artifacts.

The above problems are uniquely tied to the segmentation results. And thus instead of applying a readily available external object tracking algorithm to our segmentation results, this thesis tries to develop a object tracking algorithm that not only proved tracking but can also provide information for correcting segmentation errors.

3.4.1 Connected Components, Velocity & Direction

In the previous section we can clearly see that using just the property of spatial connectedness is insufficient to group pixels into objects. In this section we use other properties of an object to help correct these errors.

While bwlabel() provides spatial connectedness, it is by itself insufficient to track objects just using this property as between frames the objects need to be temporally matched as well.

FRAME 5 **FRAME 6**

As seen in the above frame sequence objects in frame 5 (each object is represented by a unique color) are not necessarily classified as the same objects in frame 6. To introduce this temporal information, this thesis uses Matlab's bwselect() function along with velocity and direction information of the objects.

bwselect() function:

Matlab's bwselect function is useful for maintaining the temporal information of objects. Formally the method does the following:

- BW2 = BWSELECT(BW1,C,R,N) returns a binary image containing the objects that overlap the pixel (R,C).
- BW2 contains the set of objects overlapping with any of the pixels (R(k),C(k)).
- N can have a value of either 4 or 8 (the default), where 4 specifies 4-connected objects and 8 specifies 8-connected objects.

Thus by comparing the pixels that overlap between frames this method classifies the object containing these pixels to be either be the same or different objects between these frames.

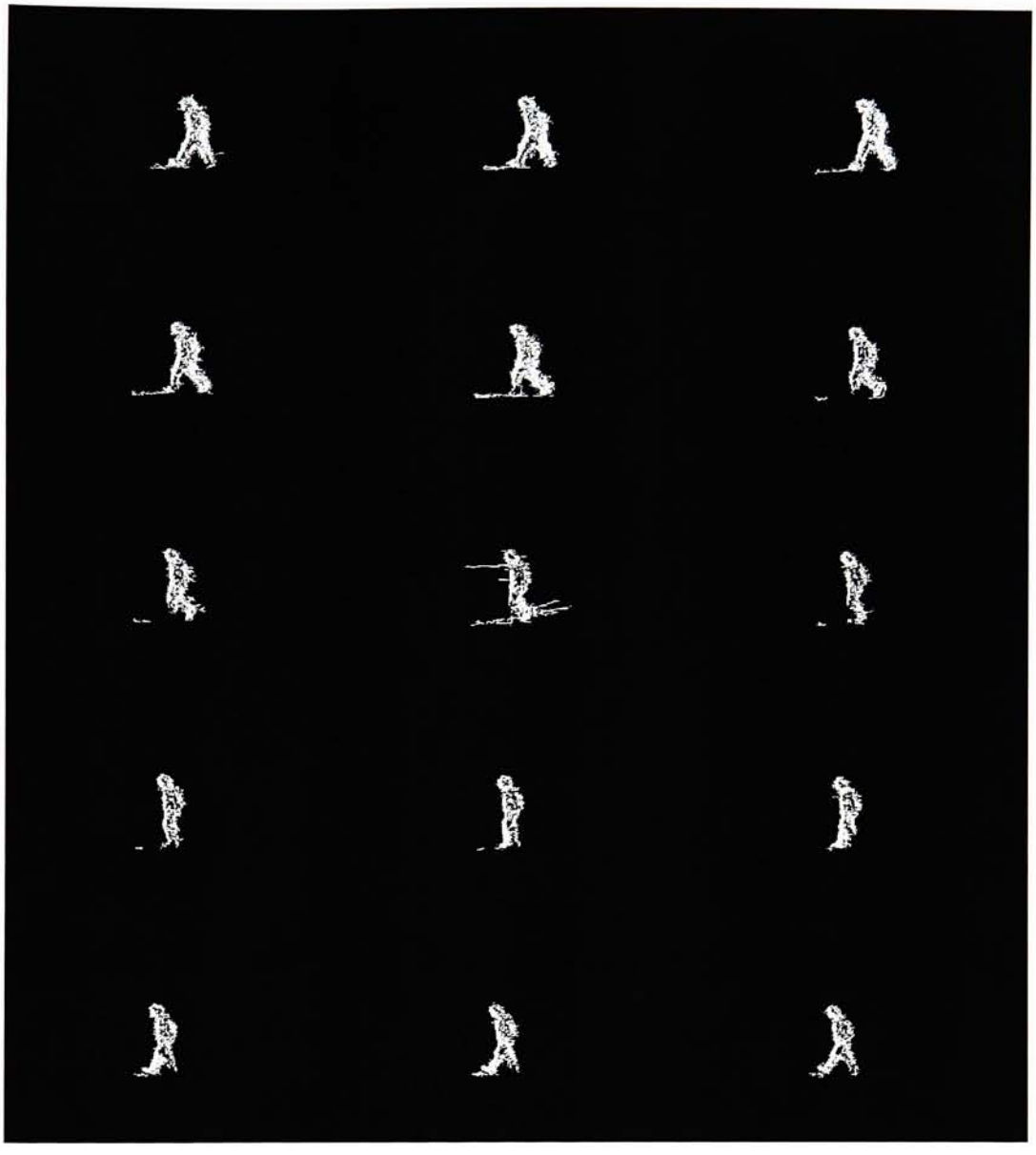
Once the groups of pixels have been classified as the same object between multiple frames, it is necessary to maintain the direction and velocity information of these objects for the following reasons:

1. When two different objects move towards each other, intersect and then move away continuing to move in their original directions respectively, the algorithm may falsely interchange the objects identification during the intersection. This can be overcome by maintaining the direction in which each object was moving. Thus an object labeled 'A' that was moving from left to right will continue to be labeled 'A' after intersecting with another object moving in another direction.
2. When two objects moving in the same direction yet with different velocities intersect the object tracking algorithm may falsely interchange the identification of these objects. This can be overcome by maintaining the velocities of the individual objects.

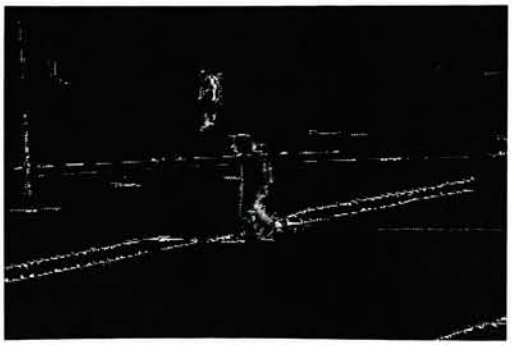
The object direction reference was maintained by using Matlab's row and column numbers for each frame. Thus an object that has incremental column numbers between frames is classified as moving right. If the row numbers also increase then the object is moving left and upwards.

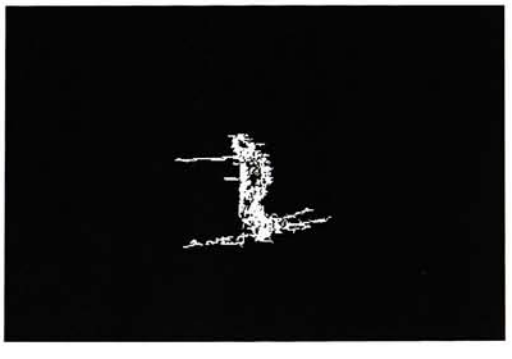
The velocity information is maintained by calculating the movement of the object in terms of number of pixels per frame. As we have seen in earlier sections the sensitivity of detection affects the pixels classified as the foreground pixels as opposed to background pixels. For larger objects some of the pixels within the large object not situated on the periphery of the object may be falsely classified as background pixels. Hence using a pixel on the periphery of the object as the reference pixel with which to measure the velocity of the object serves as a better choice than a pixel located deeper within the object. Counting the number columns or rows this pixel moves between frames gives us the velocity of the object in pixels/frame.

Using all of the above information, it is possible to achieve spatial and temporal object tracking as illustrated in the sequence of frames below



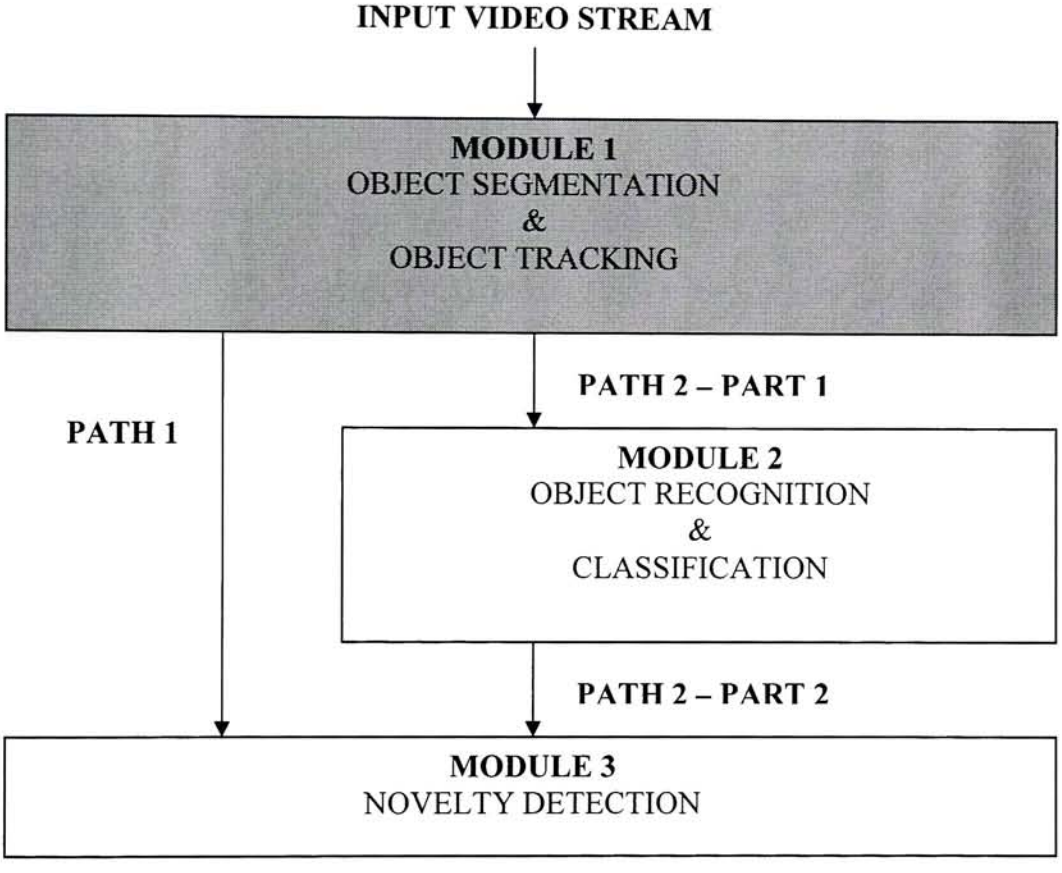
It is important to note that this object tracking algorithm removes any artifacts in the input video sequence.





3.4 Incorporation of Results into VENUS

As explained in section 1.5, Video Exploitation and Novelty Understanding System (VENUS) is a system for understanding novelty (or abnormal events) in streams of video [4]. The research in VENUS is currently spearheaded by Dr. Roger Gaborski (www.cs.rit.edu/~rsg). The intention of this thesis is to apply the results of the implementation to the existing VENUS system as detailed in the following illustration:



In the illustration, the box marked in gray represents the results from this thesis implementation. We have, thus far, studied the segmentation of foreground objects. This only forms part of the first module. The second part of this model requires the tracking of these foreground objects in the input video. Hence, in addition to segmentation, we need to keep track of the segmented objects temporally.

In order to understand the importance of object tracking as an input to either module 2 or module 3 in the VENUS system, the following section briefly describes the Novelty detection module of VENUS.

4. Conclusion:

Comparison of Parametric and Non-Parametric Gaussian Models

This thesis presents the implementation techniques and results of both parametric and non-parametric background subtraction techniques.

In parametric background subtraction methods, the algorithm updates the Gaussian model of the pixel probability distribution by changing the mean and standard deviation using a learning parameter alpha. Consequently the algorithm yields an output that is strongly reliant on the rate of update of the model. This leads to the undesirable ghosting effect that causes pixels once classified as foreground to very gradually be classified again into background once the foreground object has moved away (section 3.1.1).

Conversely in the non-parametric background subtraction model, the learning parameter is not used and the Gaussian is instead updated based on a recent history of frames. This leads to highly sensitive segmentation of foreground pixels and consequently falsely classifying parts of larger foreground objects as background pixels (section 3.2.2).

Between the two algorithms, the non-parametric approach presents more accurate segmentation. The problem with not being able to segment the entire object if the object is large can be possibly overcome by using the information provided by the object tracking algorithm. The object tracking algorithm developed in this thesis addresses part of the problem by removing the artifacts in the input video.

In conclusion the choice of algorithm for object tracking is the Non-parametric background subtraction technique. These segmented results along with the object tracking information are presented as input to the novelty detection algorithm of the VENUS system (section 3.4)

5. Research Dependencies

The existing resources of the Computer Vision Lab (70-3400) proved sufficient for all the research work in this thesis. Input data for the research will be available from the database of videos collected and available at this lab.

6. Future Work

Several important areas of this thesis present immediate opportunities for future work.

The non-parametric techniques provide accurate segmentation of smaller more dynamic objects with moving parts such as human beings in motion with their arm and leg movements. However this higher sensitivity presents a problem with larger objects when trying to segment the entire object accurately. Future course of research to overcome this problem may include investigation of color histograms to accurately represent a color model of segmented objects. Thus in addition to maintaining the direction and velocity information for object tracking, including the color information may help to completely segment such larger objects.

The second opportunity for future research would be to improve the computational time of the algorithm for segmentation as well as object tracking. The most computationally feasible step in this algorithm is during the suppression of false detections using the component and displacement probabilities (section 3.2.2). it is certainly more elegant to use values of variables from previous steps in future computations as this reduces recalculation of the older variable values, however there is a restriction on the amount of usable memory to store such values. This restriction is a problem presented when working with Matlab. Future work may investigate the feasibility of porting or developing these algorithms in Open C's image processing libraries.

Finally, on a more macroscopic level of the problem statement, this thesis only strives to solve a part of the problem of object detection and classification of the entire system VENUS. VENUS itself is modular in design to allow for re-work and re-implementation of individual segments incorporating more recent researched solutions with possibly higher efficiency. Future work may focus on possibly non-Gaussian algorithms using Hidden Markov Models or neural networks for background segmentation.

7. References

[1] C. R. Wern, A. Azarbayejani, T. Darrell, and A. P. Pentland, ``Pfinder: Real-time tracking of human body," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 1997

[2] N. Friedman and S. Russell, ``Image segmentation in video sequences: A probabilistic approach," in *Uncertainty in Artificial Intelligence*, 1997.

[3] Ahmed Elgammal; David Harwood; Larry Davis, "Non-parametric Model for Background Subtraction", ECCV 2000

[4] Video Exploitation and Novelty Understanding System (VENUS): http://www.cs.rit.edu/~rsg/Research.html

[5] C J M Tornieri, F Bremond and M Thonnat, "Updating of the reference image for Visual Surveillance Systems" IDSS 2003

[6] GeÂ rard Medioni, Senior Member, IEEE, Isaac Cohen, Member, IEEE, FrancËois BreÂmond, Somboon Hongeng, Student Member, IEEE, and Ramakant Nevatia, Fellow, IEEE: "Event Detection and Analysis from Video Streams"

[7] Alexandre R.J Francois and Gerard G. Medioni, "Adaptive Color Background Modeling for Real-Time Segmentation of Video Streams.

[8] Jain R.C., "Segmentation of Frame Sequences Obtained by a Moving Observer".

[9] Independent Component Analysis: A Tutorial
Aapo Hyvärinen and Erkki Oja, Helsinki University of Technology, Laboratory of Computer and Information Science

[10] Kernel Density Estimators:
http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0405/MISHRA/kde.html

[11] Digital Image Processing (2nd Edition) by Rafael C. Gonzalez (Author), Richard E. Woods (Author)

[12] Y. Wang, R.E. Van Dyck, J. F. Doherty, Tracking Moving Objects in Video Sequences, Proc. Conference on Information Sciences and Systems, Princeton, NJ, March 2000.