Tracking Multiple Vehicles using Foreground, Background and Motion Models

Derek R. Magee

School of Computing, University of Leeds, UK

Abstract

In this paper a vehicle tracking algorithm is presented based on the combination of a novel per-pixel (Gaussian Mixture Based) background model and a set of foreground models of object size, position, velocity, and colour distribution. Each pixel in the scene is 'explained' as either background, belonging to a foreground object, or as noise. A projective ground-plane transform is used within the foreground model to strengthen object size and velocity consistency assumptions. A learned model of typical road travel direction and speed is used to provide a prior estimate of object velocity which is used to initialise the velocity model for each of the foreground objects. The system runs at near video frame rate (>20fps) on modest hardware and is robust assuming sufficient image resolution is available and vehicle sizes do not greatly exceed the priors on object size used in object initialisation.

Key words: Vehicle tracking, Background model, Gaussian Mixture Model
PACS:

Email address: `drm@comp.leeds.ac.uk` (Derek R. Magee).
URL: `http://www.comp.leeds.ac.uk/drm` (Derek R. Magee).

1 Introduction

Systems developed for object tracking (e.g. [1–8]) may be broadly divided into explicit model based methods (where a detailed, object specific, model of object characteristics is built e.g. [4]) or implicit model based methods (where more general object, or scene, characteristics are modelled, often dynamically e.g. [5]).

Our interest is in the analysis of traffic scenes with multiple (often large numbers) of vehicles interacting. A frame from a typical sequence is shown in figure 1.

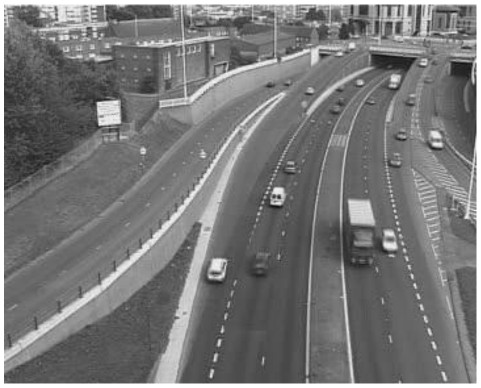


Fig. 1. Frame from a Typical Input Sequence

Several researchers have taken the approach of using detailed 3D models within a vehicle tracking scenario. Koller *et al.* [9] present a system that fits a parameterised 3D wireframe vehicle model to a scene by matching edge segments using the Levenberg-Marquart optimisation algorithm to minimise a distance measure. A uniform angular and translational velocity assumption is used for propagation of information from frame to frame and an illumination model is used to model potentially distracting shadow edges. Ferryman *et al.* [8] (based on earlier work by Sullivan [7]) take a similar approach using a set of fixed 3D models (car, van, lorry) in a vehicle mounted camera scenario. A fitness function based on image greylevel derivatives normal to estimated edges is optimised by local search. Limited results are presented for these methods and it is unclear how well such an approach would scale to large numbers of vehicles. The complexity of such methods appears proportional to the number of vehicles and no explicit multiple vehicle strategies are presented. Ferrier *et al.* [10] describe a 2D spline based model for vehicle tracking. *A priori* (2D) shape information is incorporated by the use of a template. An approximation to a Kalman filter is used to model system velocity and shape dynamics and propagate these from one frame to the next. Again the fitness measure is based on greylevel gradient/edge information and no explicit multiple object methods are suggested. Koller *et al.* [11] present a similar 2D spline based technique with occlusion reasoning for multiple objects incorporated in the fitness measure. All systems described so far use camera/groundplane calibration information to reduce the number of degrees of freedom in the model fit search and improve motion assumptions (which are generally more valid in the groundplane than the image plane).

An alternative strategy for tracking vehicles (and indeed a wide range of other objects) is to use more general models of object characteristics such as size, gross shape, colour, motion, texture etc. without a detailed *a priori* model of shape and/or appearance. Such models are often re-estimated adaptively during the tracking process. Beymer *et al.* [6] group corner features using coherent motion to track multiple vehicles. Their system has been demonstrated to be robust and reasonably accurate on a large body of evaluation data. Stauffer and Grimson [12] use a background/foreground segmentation scheme based on a mixture of Gaussian background model to track both vehicles and pedestrians. Connected component analysis is used to form foreground 'blobs' which are clustered on the basis of coherent motion using a set of Kalman filters.

We take a more general modelling strategy (a la Beymer *et al.* rather than Ferryman/Sullivan *et al.*), including both a background and foreground model. Our scheme extends the per-pixel (Gaussian mixture) background model of Stauffer and Grimson [12], improving sensitivity to lighting changes and computational efficiency. This is combined with a novel foreground model that borrows conceptually from this model. Our foreground model is based on dynamically modelling vehicle invariants size, colour and velocity (assumed locally invariant in time) for a particular vehicle. Foreground pixels (identified as outliers of the background model), rather than blobs (as used by Wren *et al.* [1]) or corner features / regions (as used by Beymer *et al.* [6]), are compared with current instances of our foreground model to determine to which object they belong (if any). These are then used to re-estimate model parameters. Kalman filters are used to propagate models over time (predictive tracking).

An alternative to the Kalman filter is the stochastic particle filter which uses samples ('particles') to represent a multi-modal distribution. Applications of this approach to true multiple object tracking (e.g. [13,14]) have highlighted that the computational cost of this approach does not scale linearly with the number of objects present. Although some methods are presented to overcome this 'curse of dimensionality' these methods are effectively limited to very small numbers of objects (typically no more than 2 or 3). Our application requires the online tracking of many objects (up to 30) and, as such, these methods are unsuitable.

A prior model of typical road velocity over the scene of interest is learned from an initial version of the tracker and used to initialise velocity and direction of travel estimates in the final implementation. This leads to 'lock' being achieved faster and fewer lost vehicles. This model uses a mixture of Gaussians to model the distribution of vehicle positions over the ground-plane. Each Gaussian in the mixture has an associated velocity. The model can thus provide a typical velocity estimate for any point on the ground-plane by forming a weighted sum of these velocities. This estimate is used when initialising new objects as an initial guess to vehicle velocity and (when normalised) direction of travel.

2 Modelling the Background Using a Multi-modal Statistical Model

The simplest background model is perhaps to take a single frame of a scene containing no objects of interest and subtract the greylevel/colour values of this image from a frame containing objects of interest. Any non zero pixel values are classified as foreground. Such a model is inadequate in most practical situations due to intensity noise, caused by imaging and lighting effects, and spatial noise, caused by camera jitter. Practical methods attempt to model this noise using statistical techniques.

Haritaoglu *et al.* [3] model background pixel intensities using minimum and maximum values and a maximum allowed difference between frames. If a pixel intensity falls outside these bounds it is classified as foreground. Ridder *et al.* [15] use a Kalman filter at each pixel to predict a single intensity value, however foreground detection is performed using a thresholded absolute difference. Wren *et al.* [1] model pixel colour as a full covariance Gaussian distribution in YUV colour-space. McKenna *et al.* [2] use a diagonal covariance Gaussian in RGB colour-space. Stauffer and Grimson [12] dynamically model the history of pixel colour values (in RGB space) at a pixel as a mixture of Gaussians[1]. Our method has its roots in the method of Stauffer and Grimson.

Mixtures of Gaussians allow the colour distribution of a given pixel to be multi-modal, which is essential if there is significant camera jitter as in our application. Figure 2 shows the distribution of pixel (RGB) colour over time at an edge pixel in the presence of camera jitter.

In Stauffer and Grimson's method [12] a fixed number of Gaussians (typically 3-5), with equal diagonal covariances, are used. The parameters of the mixture (weights ω, means μ and covariances σ^2) are updated dynamically over time using equations 1, 2 and 3.

$$\omega_{k,t} = (1-\alpha)\omega_{k,t-1} + \alpha(M_{k,t}) \tag{1}$$

Where α is the learning rate and $M_{k,t}$ is 1 if the current RGB input matches

[1] It is claimed this choice of colour-space is arbitrary and the method works well with any colourspace

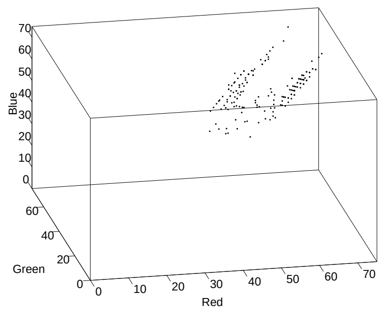


Fig. 2. Colour Values of an Edge Pixel over Time

Gaussian k (i.e. k is the closest Gaussian, in terms of Mahalanobis distance, and within n standard deviations of the mean [2]) and 0 otherwise. The weights are re-normalised after this update. If a Gaussian is matched, its mean (μ) and variance (σ) terms are updated with reference to the current RGB input (X). Parameters of unmatched Gaussians remain unchanged.

$$\mu_{k,t} = (1-\rho)\mu_{k,t-1} + \rho X_t \tag{2}$$

$$\sigma^2_{k,t} = (1-\rho)\sigma^2_{k,t-1} + \rho(X_t - \mu_{k,t})^T(X_t - \mu_{k,t}) \tag{3}$$

where:

$$\rho = \alpha\eta(X_t|\mu_{k,t}, \sigma_{k,t}) \tag{4}$$

$$\eta(X|\mu_{k,t}, \sigma_{k,t}) = e^{-\frac{1}{2}(X-\mu_{k,t})^T\sigma^{-1}_{k,t}(X-\mu_{k,t})} \tag{5}$$

If the current RGB input matches no Gaussian, the Gaussian with the lowest weight is replaced by one with its mean at the current input value and a large default variance. To remove transients (i.e. foreground pixels) from the distribution only a subset of the mixtures are used as the background model. The subset is chosen by ordering the Gaussians in descending weight order and selecting the first B distributions where:

$$B = argmin_b(\sum_{k=1}^{b} \omega_k > T) \tag{6}$$

Where T is “a measure of the minimum portion of the data that should be accounted for by background” [12].

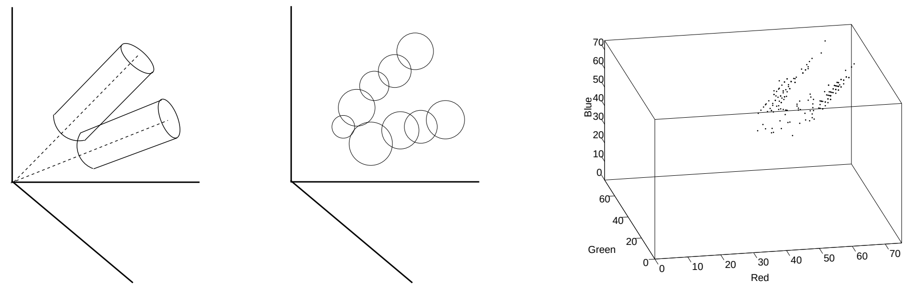
[2] n is typically 2.5

3 An Improved Background Model

Per-pixel background models such as the one presented by Stauffer and Grimson [12] are reasonably effective at classifying pixels as foreground or background but are still prone to a small proportion of false foreground and background classifications. They are also reasonably computationally expensive as they must be updated at each pixel for every frame. The false classification issue is generally a result of (i) severe lighting variations, (ii) imaging noise and (in the case of false background classifications) (iii) foreground objects being the same (or similar) colour to the background at a given pixel. Issue (i) may be (partially) resolved at the pixel level, however all three issues may be addressed using higher level (not per-pixel) processes both as post processing/analysis operations (see later sections) and feeding back into the per-pixel model. Various enhancements to such models are presented in this section.

3.1 Modelling Colour Variation along 'Pseudo-Principal Axes'

In this section we present an alternative to the mixture of Gaussians model for (RGB) pixel colour modelling, which uses a mixture of axes orientated cylindrical distributions. Figure 3 shows this model conceptually, side to side with real data from an edge pixel.



a) b) c)

Fig. 3. a) Model 'Membership' for Mixture of Cylinders Model, b) Model 'Membership' for Mixture of Gaussians, c) Multi-modal Pixel Data

In outdoor scenes cloud cover can lead to large intensity changes over short periods of time. Uni-modal background modelling schemes model this as as single elongated distribution in colourspace (sometimes reducing the specificity of the model) or use an intensity normalised colourspace (reducing the specificity of the representation and decreasing the signal to noise ratio for compressed streams). The Stauffer/Grimson method forms a Gaussian mixture model with equal diagonal covariance Gaussians, and puts a hard limit (typically 2.5 standard deviations) on mixture membership. This effectively defines the background colour distribution as lying within a set of spheres in colourspace. Figure 4 illustrates that (for a non-edge pixel) the RGB colour distribution is not modelled well by a sphere or set of spheres.

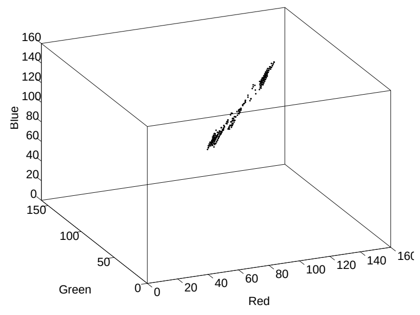


Fig. 4. Colour Values for a Single Non-Edge Pixel

It can be seen that (non edge pixel) colour values approximately lie along a straight line segment in colourspace that, if extended, approximately intersects the origin. The Stauffer/Grimson method models this uni-modal distribution as a multi-modal set of spheres. If an intensity normalised colourspace is used, any point along the complete line is equivalent, allowing a single sphere to be used, however the discriminative power of the space is reduced. In compressed streams the signal to noise ratio is also greatly reduced in such a colour-space as compression schemes compress colour more than intensity[3].

We propose an approach that uses a mixture of axis orientated cylindrical distributions to exploit our knowledge of the underlying data distribution. Our approach is to define a set of 'pseudo-principal axes' as lines between the origin and a mixture component means, represented by the vector $\bigtriangledown_n$ (calculated as $\bigtriangledown_n = \mu_n / |\mu_n|$), for each mixture component. Each mixture component consists of a 1D mean and variance along one of these axes which is estimated from colour components (X_{RGB}) projected as:

$$X_{Pr_n} = X_{RGB} \cdot \bigtriangledown_n \tag{7}$$

and by Pythagoras rule perpendicular deviation is:

$$X_{Per_n} = \sqrt{|X_{RGB}|^2 - {X_{Pr_n}}^2} \tag{8}$$

Mixture parameters are updated as in the Stauffer/Grimson method (equations 1 to 3) using X_{Pr_n} as X_t, however the data-mixture membership function ($M_{k,t}$) is adjusted to exclude membership if X_{Per_n} is above a noise threshold (thus defining a cylinder in RGB space). This method may be thought of as an approximation to a mixture of Eigenspaces [16,17], without the computational overhead of calculating Principal Components.

[3] Ad-hoc experiments have showed this is highly significant with many foreground artifacts appearing when using a normalised colourspace and compressed streams.

The Stauffer/Grimson method adapts over time to encode a finite history of events. A learning rate controls the scale of this history (a faster rate encodes a shorter history) with a trade off between being fast enough to adapt to changes and slow enough to store a useful temporal history. At fast adaptation rates the distribution quickly becomes dominated by a single Gaussian (thus uni-modal).

We use a variable framerate update for each pixel, which improves temporal history storage (for slow changing pixels) while running at reasonably high adaptation rates (adapting to a novel background within a second or two rather than 20+ seconds as in the Stauffer/Grimson setup) for less stable pixels. Stable pixels (i.e. pixels that fit the background distribution over a number of frames) are updated with a lower framerate than pixels that have been recently classified as foreground given by:

$$R_{x,y} = \begin{cases} R_{input}/N_{bgd_{x,y}}, & \text{for } N_{bgd_{x,y}} < N_{max} \\ R_{input}/N_{max}, & \text{otherwise} \end{cases} \tag{9}$$

Where:
$R_{x,y}$ = Update Framerate (frames/second) at pixel x,y
R_{input} = Input Framerate (frames/second)
$N_{bgd_{x,y}}$ = No. of consecutive frames classified as 'background' at pixel x,y
N_{max} = Maximum threshold on frame rate division (typically 25 frames or 1 second).

This addition also serves to reduce the computational expense of the method considerably. The maximum threshold on frame rate division (N_{max}) ensures the update framerate is not reduced to a level below which it fails to adapt to slow background colour or intensity changes.

Our scheme, as described so far, does not differentiate between changes in the background, and foreground models entering the scene. This results in fast adaptation at pixels containing a foreground object. As a result slow moving vehicles may become included in the background model (this is a problem of the original Stauffer/Grimson formulation also). A higher level (i.e. non per-pixel) process may be used to provide this object/noise classification which can be fed back into the per-pixel model. The foreground modelling/tracking method described in this paper (see sections 4 and 6) provides us with exactly this information. For pixels that are classified as foreground by the higher level process (the tracker) the update framerate (R) is reduced to a minimum level (typically R_{input}/N_{max}). In concurrent work with that presented here Harville

[18] suggests reducing this to zero (i.e. not adapting at all when a pixel is classified as foreground by the higher level process). This risks propagating any failures in the higher level system forward in time ad-infinitum. It is safer to set the update framerate very low, however the more detailed the higher level process is, the less likely this scenario is to occur. Harville also suggests a complex 'negative feedback' system to correct pixels classified as foreground by the background model but as background by the higher level process. This is based on having two mixture models per pixel. Negative feedback in not implemented in our system because such mis-classifications are dealt with (or ignored) by the higher level tracking process (see later), however it may be worth investigation for other applications.

3.3 Modelling Very High and Very Low Intensity Pixels Using Just Intensity

At low and high intensities the assumption that uni-modal data lies along a straight line breaks down due to reduction in signal to noise ratio (at low intensities) and colour saturation (at high intensities). This is why using intensity normalised colourspaces is not necessarily a good idea. If pixel intensity is above a maximum threshold[4], or below a minimum threshold the Gaussian mixture model is converted into a 1D (intensity) model by using the intensity of each Gaussian mean as the model. The foreground/background classification is based on the pixel intensity being above/below an absolute difference threshold from any of the Gaussian intensity means (i.e. if the difference is below the threshold for any Gaussian mean the pixel is classified as background in exactly the same way as for the colour based classification).

3.4 Other Modifications

To further prevent the distribution becoming effectively uni-modal when a pixel colour value is relatively constant, an upper limit is put on the weight of any one Gaussian component (typically 0.5). On update if a sample matches a Gaussian with a weight above this threshold the mixture weight is left unchanged. This preserves long time-period multi-modality.

[4] A better approach to detect saturation may be to put maximum thresholds on all three colour channels, adapting the method if any one channel is near saturation.

4 Modelling Foreground Objects

Stauffer and Grimson [12] associate 'blobs', extracted using connected components analysis, into objects using a Kalman filter. A similar approach is taken by Beymer *et al.* [6] who associate regions containing corner features. Such methods are essentially a simplification of the strategy employed by Wren *et al.* [1] in which prior spatial and temporal information is included in addition to motion assumptions. These methods work by classifying extracted features (corners, blobs / connected regions) as coming from one of a number of processes (vehicles, body parts).

In our scenario, features such as corners are unreliable due to the relative size of the objects of interest. Connected components analysis (based on foreground from a per-pixel background model) is also a poor tool as the similarity of objects to background in some cases can result in a highly fragmented foreground. This is illustrated in figure 5.

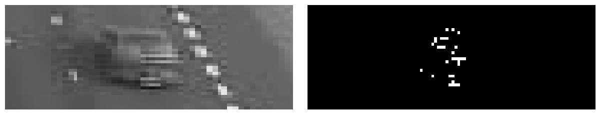


Fig. 5. Fragmented Foreground Produced Using the (original) Stauffer/Grimson Method

This shows that performing connected components analysis adds little information and, if small regions are discarded, may discard important information. We thus choose to associate foreground pixels[5], rather than blobs or regions, with object models. This has the added advantage of a computational saving. Our model consists of a velocity estimate (based on a Kalman filter) and a pixel-object membership function based on representing:

Position: A single 2D point is used to represent the centroid of an object on the ground plane.
Size: 1D Gaussians are used to represent the object size as the variation of contributing pixel locations about the object centroid (in ground plane coordinates) in, and perpendicular to, the direction of travel.
Colour Distribution: A Gaussian mixture is used to represent colour over the entire object in the same way as the background model.

[5] Foreground pixels are detected as outliers in the background model distribution, as described previously

5 Image-Groundplane Transformation

Foreshortening and perspective effects in the scene result in objects with constant size and velocity having non-constant size and velocity in the image plane. To alleviate these problems all pixels are back-projected onto a groundplane in which object size and velocity is a linear transform of actual object size and velocity. This is done by means of a projective transform. To perform this mapping we must consider the projective plane (projective 2-space $\mathbb{P}^2$) which is defined by the set of lines in $\mathbb{R}^3$ that pass through the origin. The equivalence between a euclidean 2-space $\mathbb{R}^2$ (e.g. the image plane or groundplane) and the projective 2-space $\mathbb{P}^2$ is given by considering the plane $z = 1$ in $\mathbb{R}^3$. Any point on this plane defines a unique point of $\mathbb{P}^2$:

$$(x, y)^T \in \mathbb{R}^2 \leftrightarrow [x, y, 1] \in \mathbb{P}^2 \tag{10}$$

A projective transform $\phi : \mathbb{P}^2 \rightarrow \mathbb{P}^2$ is defined by a non-singular 3×3 matrix $\mathcal{A}$ such that:

$$\phi[P] = [\mathcal{A}P] \quad \forall [P] \in \mathbb{P}^2 \tag{11}$$

Thus performing a projective transformation from one euclidean space to another (e.g. image-plane to ground-plane or vice versa) involves representing the point in $\mathbb{P}^2$ (equation 10), performing the transform and representing the result in euclidean space. Thus in projective space:

$$[x', y', z'] = \mathcal{A}[x, y, 1] \tag{12}$$

and in euclidean space:

$$\begin{aligned} (x_{proj}, y_{proj})^T \in \mathbb{R}^2 &\leftrightarrow [x', y', z'] \in \mathbb{P}^2 \\ &\leftrightarrow \left[\frac{x'}{z'}, \frac{y'}{z'}, 1\right] \in \mathbb{P}^2 \\ &\leftrightarrow \left(\frac{x'}{z'}, \frac{y'}{z'}\right)^T \in \mathbb{R}^2 \end{aligned} \tag{13}$$

Where (x_{proj}, y_{proj}) is the projective transformation of (x, y) from one euclidean plane to another.

The process of image to ground-plane calibration is simply a matter of estimating the 3×3 matrix $\mathcal{A}$. If a number of point correspondences are known between the ground-plane and image-plane $\mathcal{A}$ may be estimated by solving (in the least squares sense) for the set of simultaneous equations produced by plugging these points into equations 12 and 13.

If a set of corresponding points are not available (for example it is difficult to take measurements from the scene) other information may be used to estimate $\mathcal{A}$. In our scenario, we have a road in the image plane which we know is of constant width in the groundplane. A novel method for using this information to estimate $\mathcal{A}$ is presented in appendix A.

6 Tracking via Prediction and Model Re-estimation

Tracking is performed by predicting forward object positions from the previous frame into the current frame using a Kalman filter, and associating each foreground pixel with a single object model. This is done by defining a distance measure based on the Position and Size of each object. This is given as the maximum deviation (normalised by variances V_{in}, V_{per}) from predicted object mean (μ_{pred}) in, or perpendicular to, the direction of travel:

$$D^2 = \max\left(\frac{\Delta_{in}^2}{V_{in}}, \frac{\Delta_{per}^2}{V_{per}}\right) \tag{14}$$

Maximum deviation is used (as opposed to using Mahalanobis distance deviation), as the distribution of pixel locations in the two directions is not independent due to the rectangular nature of most vehicles when viewed from above. A pixel is associated with the model with the lowest distance (D) if this distance is below a specified threshold (typically 2.5). If this distance is above the threshold, but below a second threshold (typically 4 or 5) the colour value of the pixel is compared to the colour Gaussian mixture model by taking the minimum mahalanobis distance of the colour value from a mixture component mean. If this distance is less than a specified threshold (typically 2.5) the pixel is accepted as resulting from this model, otherwise it is rejected. This allows the object size hypotheses to be enlarged over time from the initial hypotheses if (and only if) the image colour information supports it (figure 6).

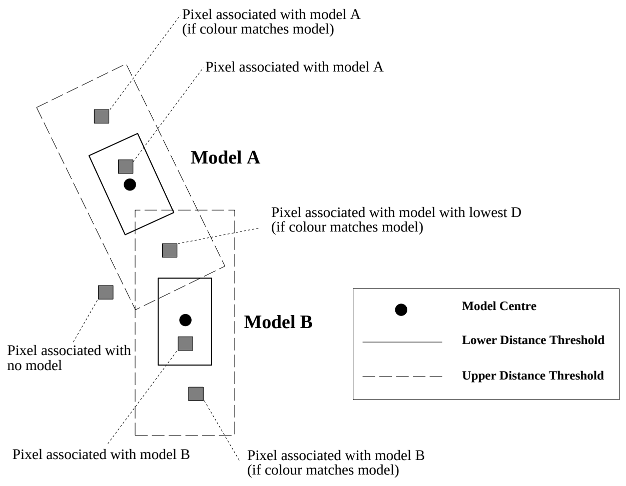

Pixel associated with model A (if colour matches model)
Pixel associated with model A
Model A
Pixel associated with model with lowest D (if colour matches model)
Model B
Pixel associated with no model
Pixel associated with model B
Pixel associated with model B (if colour matches model)
Model Centre
Lower Distance Threshold
Upper Distance Threshold

Fig. 6. Associating Foreground Pixels with Models

If a pixel is not classified as resulting from one of the current foreground objects, and it has neighboring foreground pixels[6], a new model is initialised centered at this pixel with pre-specified size parameters (based on the typical size of a car). Initially the colour distribution is uninitialised, and it is built up over the first few frames. Model parameters are updated at each frame from associated pixels as with the background model. Updated model positions are calculated as a weighted centroid of associated pixels:

$$\mu_{new} = \frac{\sum X_n \eta(\Delta_{in}|\mu_{pred}, V_{in})\eta(\Delta_{per}|\mu_{pred}, V_{per})}{\sum \eta(\Delta_{in}|\mu_{pred}, V_{in})\eta(\Delta_{per}|\mu_{pred}, V_{per})} \tag{15}$$

Where:
X_n = The x,y position of pixel n
$\eta()$ = Normalised Gaussian kernel function (see equation 5)

Size representations are calculated as the squared distances from associated pixel locations to predicted centroids in the direction of travel (V_i) and perpendicular to the direction of travel (V_p):

$$V_i = \frac{\sum((X_n - \mu_{pred}) \cdot \triangledown)^2}{N} \tag{16}$$

$$V_p = \frac{\sum((X_n - \mu_{pred}) \cdot \triangledown^{\perp r})^2}{N} \tag{17}$$

Where $\triangledown$ and $\triangledown^{\perp r}$ are unit vectors in, and perpendicular to, the direction of travel respectively. Colour distributions are updated in the same way as the background model (section 2), with pixels from different spatial, as well as

[6] This acts as a point noise rejection mechanism.

temporal, locations updating the model. As with the background model an update factor (α) is used for position/size models, updated as:

$$\mu_{n+1} = \alpha\mu_{new} + (1-\alpha)\mu_n \tag{18}$$

$$V_{i_{n+1}} = \alpha V_{i_{new}} + (1-\alpha)V_{i_n} \tag{19}$$

$$V_{p_{n+1}} = \alpha V_{p_{new}} + (1-\alpha)V_{p_n} \tag{20}$$

When an object is initialised it is unknown whether the object being tracked is a true object of interest or simply the result of non-point noise. This classification can only be performed with information from more than one frame on the basis that true objects of interest (vehicles) have different spatial, and more importantly temporal, characteristics to noise. An initialisation period (typically 10-15 frames) is defined. During this period the update factor (α) is set reasonably high to allow rapid adaptation to the new object. At the end of this period (assuming image support remains for this length of time) the object is classified as either a true object of interest or noise. This classification is based on (i) the number of pixels associated with the object model over this period and (ii) the distance traveled by the object. If both these variables are above a threshold the object is classified as a true object and α reduced to 'lock on' to the target, otherwise the object is discarded.

If at any point there is no image support for (i.e. no pixels associated with) a particular object model it is simply propagated forward in time using the Kalman filter velocity estimate. If this persists for a number of frames (typically 3-5) the object is removed.

7 Using A-priori Road Information

The initial implementation included no knowledge of road direction or typical speeds. This forced initial velocity estimates to zero and direction of travel estimates to arbitrary values. Velocity and direction of travel information are integral in the tracking process. Poor initial values for these can lead to failure of a model to 'lock on' to a vehicle (especially small objects further from the camera). Model means can lag behind the centroid of the moving vehicle until the velocity estimate becomes realistic. If image support is poor the lag can become large enough that track is lost. The direction of travel affects the perceived aspect ratio of the vehicle, this can also lead to poor tracking. The solution to this problem is to build a prior model of typical road travel direction and speed. As cars typically travel in the same direction, within a limited speed range, on the same stretch of road (due to traffic regulations) any prior based on previous observation should be a reasonable estimate of the actual direction and speed of a novel observation. Our model is trained offline on the output of the initial implementation of the tracker, on the basis that, in general, this gives accurate tracking results and false and missing tracks are rare enough to be treated as outliers.

The groundplane position output of the tracker is quantised using the vector quantiser proposed by Johnson and Hogg [19]. This is a Self Organising Map [20] like method that generates a fixed number of prototypes for a large data set. More prototypes are generated in areas of high data density and less in areas of low data density. Positional (ground plane) space is then represented as a set of 2D Gaussian mixtures centered on these prototypes using the adaptive Kernel method [21]. This method places Gaussian Kernels with higher variance in areas of lower prototype density. The adaptive kernel probability density function (PDF) is defined as:

$$P(x) = \frac{1}{N}\sum_{j=1}^{N}(h\lambda_j)^{-d}\mathbf{G}(\frac{x-x_j}{h\lambda_j}) = \sum_{j=1}^{N} w_j G(\mathbf{x}|\mu_j,\sigma_j) \tag{21}$$

Where:
N = No. of Data Points
$\mathbf{G}()$ = Gaussian Kernel Function (covariance equal to that of data points)
$G()$ = Alternative (probabilistic) representation of Gaussian Kernel Function
h = Window Width
λ_j = Local bandwidth factor for Gaussian j
d = Dimensionality of the data points

The local bandwidth factor (λ_j) for each Gaussian in the mixture is calculated as:

$$\lambda_j = (p'(x_j)/g)^{\frac{1}{2}} \tag{22}$$

Where:
g = The geometric mean of $p'(x_i)$

The optimal window width (h), in the sense of minimising mean integrated square error, may be determined using cross-validation. This is discussed by Silverman [21]. For a multivariate normal kernel (of which the multivariate Gaussian is the general form) the window width is:

$$h = \{4/(2d+1)\}^{1/(d+4)} \tag{23}$$

For this application a window width (h) approximately 10 times smaller is more appropriate as we are trying to form a good spatial representation of the space rather than an optimal PDF (using optimal h produces wide highly overlapping kernels).

In this way the 'contribution' of the i'th point in space to the j'th prototype may be calculated as:

$$p_{ij} = \frac{w_j G(\mathbf{x}_i|\mu_j, \sigma_j)}{\sum_{j=1}^{m} w_j G(\mathbf{x}_i|\mu_j, \sigma_j)} \tag{24}$$

Given the velocity output of the tracker, a direction vector ($\bigtriangledown_i$) and speed (S_i) can be calculated for each vehicle (i) at each timestep. We use the contribution of each of these to each prototype to calculate a weighted mean direction and speed as:

$$\bar{\bigtriangledown}_j = \frac{\sum_{i=1}^{N} p_{ij} \bigtriangledown_i}{\sum_{i=1}^{N} p_{ij}} \tag{25}$$

$$\bar{S}_j = \frac{\sum_{i=1}^{N} p_{ij} S_i}{\sum i = 1^N p_{ij}} \tag{26}$$

Figure 7 shows a typical ground plane velocity map.

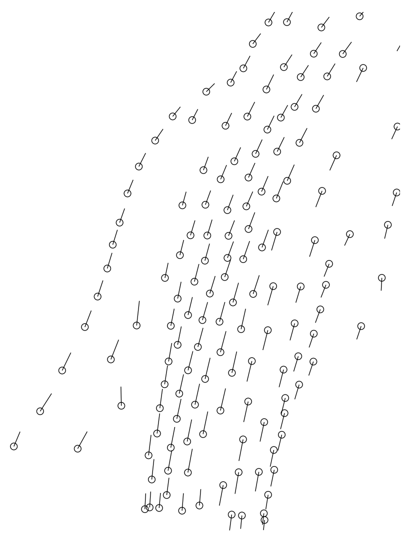


Fig. 7. Learned Ground Plane Velocity Map

The velocity map is used to calculate an initial estimate of velocity and direction (normalised velocity) when new object instances are initialised using:

$$V_{xy} = \frac{\sum_{j=1} M \bar{S}_j \bar{\bigtriangledown}_j G(x, y|\mu_j, \sigma_j)}{\sum_{j=1} M G(x, y|\mu_j, \sigma_j)} \tag{27}$$

8 Results

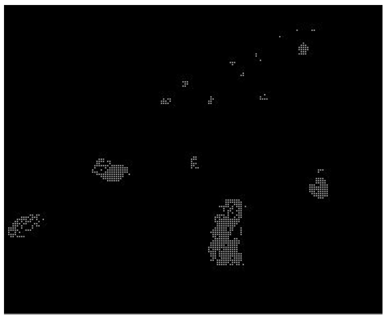
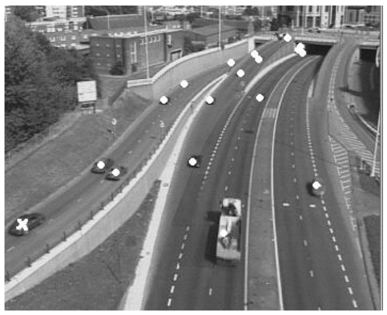
a)  b) 

Fig. 8. Sample results: a) Pixels classified as foreground (half image resolution), b) Tracked Centroids

Figure 8 shows sample tracker results projected into the image plane. The circles represent the centroids of objects definitely classified as vehicles and the cross represents an object newly detected that is yet to be classified. It is particularly interesting to note the tracker can differentiate between the two cars on the far left by their differing colour. Estimates for object size and velocity are available in groundplane co-ordinates. Figure 9 illustrates estimated velocity and object boundary box (based on 3 std. deviations from the centroid/mean).

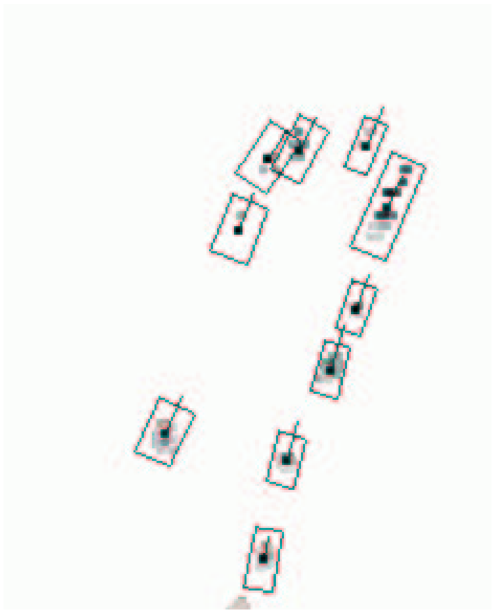


Fig. 9. Typical Ground Plane Centroid, Velocity and Bounding Box Estimates produced by the Tracker (and projected foreground pixels)

9 System Evaluation

The vehicle tracker was evaluated by drawing a box around all vehicles in a scene using an interactive tool. The results of this for a single frame is shown in figure 10.

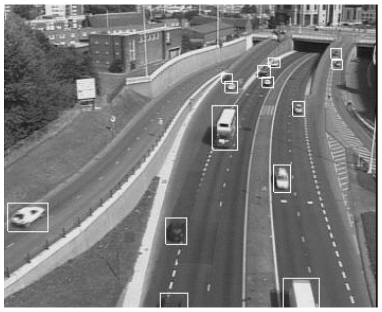


Fig. 10. Hand Fitted Boxes Used as 'Ground truth' for Evaluation

This fit was performed at ten frame intervals on a 1 minute (25fps) sequence. The sequences contains 102 separate objects with 12-28 being present in any one frame (2563 instances). The sequence was carefully chosen to contain several types of vehicles (cars, vans, lorries) and different sorts of flow (congested and relatively free flow). Background models were evaluated to determine what proportion of pixels outside these boxes were classified as foreground. For the original Stauffer/Grimson model 0.77% (s.d. 0.64) were wrongly classified as foreground, for the improved model this was reduced to 0.39% (s.d. 0.5) (resulting in the virtual elimination of false objects being tracked). Statistics for how well the foreground tracker matches the 'ground truth' were also gathered. Objects are significantly smaller in the background due to foreshortening, so the evaluation was performed with the scene divided into three equal regions:

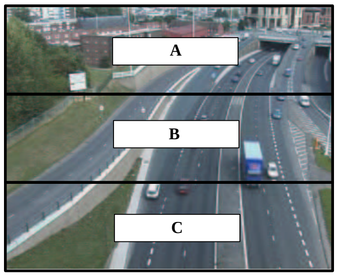


Fig. 11. Definition of the three evaluation areas

Typical sizes of objects in these regions are 4x3, 8x7 and 16x11 pixels respectively at a resolution of 180x144. This resolution was chosen as it is possible to run the tracker at near frame rate (20-23Hz) on a conventional PC (PIII 1GHz) at this resolution. Figure 12 shows the results for the full tracker presented divided by region.

These results are comparable with those presented by Beymer *et al.* [6] (al-

Section	Objects Tracked		Prop. Frames Tracked	
	Road Model	No Model	Road Model	No Model
C	100%	100%	95.4% [12.8]	92.0% [21.7]
B	100%	100%	97.1% [11.9]	96.4% [11.1]
A	97.9%	96.9%	71.7% [28.9]	71.2% [28.5]

Values in square brackets are std. deviations

Fig. 12. Tracking Results for different Regions

though evaluated on a different sequence). It can be seen that the tracker performs well in the regions C (near distance) and B (middle distance), but poorer in region A (far distance). The use of *a priori* road information gives most benefit in region C. This is due to the fact many objects enter the scene in region C (none enter in region B). An additional consideration when evaluating an object tracker is multiple object hypotheses being associated with different models. This is rarely a problem when tracking cars (typically less than 1% of cars are associated with multiple hypothesis), however for larger vehicles such as lorries this more frequent. These large vehicles represent 5-10% of the vehicles observed.

10 Discussion, Conclusions and Future Work

We have presented a vehicle tracking system that has potential for use in an online road monitoring scenario. The system is based on the combination of a mixtures of Gaussians colour background model and a set of foreground models each of which propagates a single estimate of an object position, velocity, size and colour distribution. An *a priori* model of typical road travel direction and speed and priors on object size are used in the initialisation phase. This system has been evaluated offline against a hand labeled sequence and found to be robust if a) There is sufficient resolution in the image for the background model to distinguish an object from sensor noise (typically the object must be at least 6x6 pixels in the image plane) and b) the prior on the size of the vehicle is accurate.

Issue a) is easily addressed by camera zooming (and multiple cameras if necessary). For issue b), the initialisation scheme described in this paper is computationally efficient but basic and not ideal in terms of initial position and size estimates. Pixels that don't match a current object hypothesis are used to initialise new object instances in raster order. This leads to new hypotheses located at the top left of a new object (not at the centroid. This is the subject of current research.

We have utilised the vehicle tracker implementation in a driving behaviour analysis system [22] in which interactions between different vehicles are analysed and non-typical events flagged. In future work we are hoping to install an online system at a road junction in York, England and interface the output with existing models of road traffic flow, with the eventual aim of automatic road traffic management.

In conclusion we have presented an vehicle/object tracking scheme based on generally true assumptions about scene and object characteristics (background colour consistency and local position, size, colour and velocity consistency for moving objects). We have identified previously where these assumptions break down. This is in contrast to more detailed models where assumptions (such as shape consistency over objects) are at best approximations to the truth. Priors on object characteristics (such as information on expected size and velocity) are easily incorporated into this scheme. The tracker could be extended to include shape priors (if appropriate) which may model objects better than the current Gaussian assumption. However, it is not clear that any single shape prior would be generally applicable due to the wide range of observed object shapes.

The combination of background and foreground models serves to completely 'explain' the observed scene and is an approach that is now feasible for online systems. There is much scope for future research into methods that fully model (explain) an observed scene as the result of a set of underlying processes.

11 Acknowledgments

This work was funded by the European Union under contract IST-2000-29375, the 'Cognitive Vision' project. I would like to thank my collaborators on this project David Hogg, Afrodite Galata and Tony Cohn for their support in carrying out this work and colleague Chris Needham for some valuable discussions.

References

[1] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: Real-time tracking of the human body, IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7) (1997) 780–785.

[2] S. McKenna, S. Jabri, Z. Duric, H. Wechsler, Tracking interacting people, in: Proc. International Conference on Automatic Face and Gesture Recognition, 2000, pp. 348–353.

[3] I. Haritaoglui, D. Harwood, L. Davis, W4: Who? when? where? a real time system for detecting and tracking people, in: Proc. IEEE Conference on Automatic Face and Gesture Recognition, 1998, pp. 222–227.

[4] T. Cootes, G. Edwards, C. Taylor, Active appearance models, in: Proc. Europen Conference on Computer Vision, Vol. 2, 1998, pp. 484–498.

[5] S. McKenna, Y. Raja, S. Gong, Tracking colour objects using adaptive mixture models, Image and Vision Computing 17(3-4) (1999) 225–231.

[6] D. Beymer, P. McLauchlan, B. Coifman, J. Malik, A real-time computer vision system for measuring traffic parameters, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 1997, pp. 495–501.

[7] G. Sullivan, Model-based vision for traffic scenes using the ground-plane constraint, Real-time Computer Vision D. Terzopoulos and C. Brown (Eds.). Cambridge University Press.

[8] J. Ferryman, S. Maybank, A. Worrall, Visual surveillance for moving vehicles, International Journal of Computer Vision 37(2) (2000) 187–197.

[9] D. Koller, K. Daniilidis, H.-H. Nagel, Model-based object tracking in monocular image sequences of road traffic scenes, International Journal of Computer Vision 10(3) (1993) 257–281.

[10] N. Ferrier, S. Rowe, A. Blake, Real-time traffic monitoring, in: Workshop on Applications of Computer Vision, 1994.

[11] D. Koller, J. Weber, J. Malik, Robust multiple car tracking with occlusion reasoning, in: Proc. European Conference on Computer Vision, 1994, pp. 186–196.

[12] C. Stauffer, W. Grimson, Adaptive background mixture models for real-time tracking, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, 1999, pp. 246–252.

[13] H. Tao, H. Sawhney, R. Kumar, A sampling algorithm for tracking multiple objects, in: Proc. Workshop on Vision Systems, 1999, pp. 53–68.

[14] M. Isard, J. MacCormick, BraMBLe: A bayesian multiple-blob tracker, in: Proc. IEEE International Conference on Computer Vision, 2001, pp. 34–41.

[15] C. Ridder, O. Munkelt, H. Kirchner, Adaptive background estimation and foreground detection using kalman-filtering, in: Proc. Recent Advances in Mechatronics, 1995, pp. 193–199.

[16] A. Heap, D. Hogg, Improving specificity in PDMs using a hierarchical approach, in: Proc. British Machine Vision Conference, Vol. 1, 1997, pp. 80–89.

[17] M. Tipping, C. Bishop, Probabilistic principal component analysis, Journal of the Royal Statistical Society B 12(3) (1999) 611–622.

[18] M. Harville, A framework for high-level feedback to adaptive, per-pixel, mixture-of-gaussian background models, in: Proc. European Conference on Computer Vision, 2002, pp. 543–560.

[19] N. Johnson, D. Hogg, Learning the distribution of object trajectories for event recognition, Image and Vision Computing 14 (1996) 609–615.

[20] T. Kohonen, The self organising map, Proceedings of the IEEE 78(9) (1990) 1464–1480.

[21] B. Silverman, Density Estimation for Statistics and Data Analysis, Chapman and Hall, 1986.

[22] A. Galata, A. Cohn, D. Magee, D. Hogg, Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models, in: Proc. European Conference on Artificial Intelligence, 2002.

Appendix A: Estimating the Image to Groundplane Transformation using a Constant Road-width Assumption

If we approximate the groundplane transformation ($\mathcal{A}$) by a rotation about the image X-axis the problem is reduced to one unknown (θ). This is a reasonable approximation for many scenes, including our own (see figure 1). Thus:

$$\mathcal{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & cos\theta \end{pmatrix} \tag{28}$$

We attempt to estimate θ for a particular scene by considering a road with constant width (K):

We consider the X component of the displacement between P$_l$ and P$_r$ (labelled **D** on figure 13). Thus:

$$K = \frac{D}{cos\Psi} \tag{29}$$

Consider the ideal transformation from the image plane that results in:

$$x_r^{''} - x_l^{''} = K\cos\Psi \tag{30}$$

Where $x_1^{''}$ and $x_r^{''}$ are the X components of P$_l$ and P$_r$ respectively, which are the projective transforms of the corresponding points on the image plane x_l, y_l

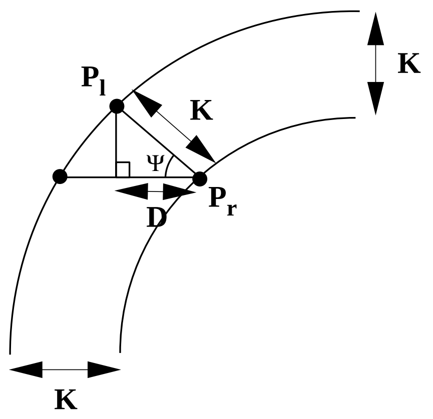


Fig. 13. Constant Width Road in the Groundplane

and x_r, y_r onto the ground plane. Substituting equations 12, 13 and 28 to into equation 30 we get:

$$\frac{x_r}{y_r \sin\theta + \cos\theta} + \frac{x_l}{y_l \sin\theta + \cos\theta} = K\cos\Psi \tag{31}$$

To solve for the unknown θ we need to make two approximations. Firstly we approximate y_l and y_r by $(y_l + y_r)/2$. This has little effect on equation 31 as long as Ψ is small or $|y_r - y_l|$ is small in comparison with y_l and y_r. In general one or other of these conditions is true for most road scenes of the sort shown in figure 1. This simplifies equation 31 to:

$$\frac{x_r - x_l}{0.5(y_l + y_r)\sin\theta + \cos\theta} = K\cos\Psi \tag{32}$$

The second approximation we make is for the road angle Ψ. Ideally the value of Ψ in the ground-plane should be used, however this is unknown. We approximate this by the equivalent angle in the image plane (ψ_{img}). Re-arranging equation 32 gives:

$$x_r - x_l = \cos\Psi_{img}(0.5(y_l + y_r)K\sin\theta + K\cos\theta) \tag{33}$$

Equation 33 has two unknowns ($K\sin\theta$ and $K\cos\theta$), and so at least 2 pairs of image plane points are required to solve for these. It should be noted that the corresponding groundplane point pairs are not required. In practice we solve for $K\sin\theta$ and $K\cos\theta$ using more than two point pairs and a least squares approach, solving the system:

$$\begin{pmatrix} 0.5(y_{l,1}+y_{r,1})\cos\Psi_{img_1} & \cos\Psi_{img_1} \\ 0.5(y_{l,2}+y_{r,2})\cos\Psi_{img_2} & \cos\Psi_{img_2} \\ . & . \\ . & . \\ 0.5(y_{l,n}+y_{r,n})\cos\Psi_{img_n} & \cos\Psi_{img_n} \end{pmatrix} \begin{pmatrix} K\sin\theta \\ K\cos\theta \end{pmatrix} = \begin{pmatrix} x_{r,1}-x_{l.1} \\ x_{r,2}-x_{l.2} \\ . \\ . \\ x_{r,n}-x_{l.n} \end{pmatrix} \tag{34}$$

We can then calculate θ (and as such the projective transformation from equation 28) as:

$$\theta = \tan^{-1}\left(\frac{K\sin\theta}{K\cos\theta}\right) \tag{35}$$

The value of θ calculated is only an approximation to the optimal value of θ due to the approximation to Ψ used. An improved estimate may be obtained by projecting the road from the image plane using this approximation and repeating the process on the projected road outline on the basis that this initial projection is closer to the actual groundplane outline than the image plane outline, and thus the values of Ψ used are a better approximation to the true Ψs. This process can be repeated iteratively to improve the estimate, with θ at each subsequent iteration estimated as:

$$\theta_{n+1} = \theta_n + \alpha\delta\theta_{n+1} \tag{36}$$

Where $\delta\theta_{n+1}$ is the estimated value of θ at iteration n+1 and α is a damping factor (between 0 and 1) to prevent overcorrection/oscillation in the final stages of the iteration. ($\theta_0 = 0$).

Road outlines and centrelines are approximated by 3 polynomials (typically 2nd or 3rd order). These are estimated using a least squares approximation technique from hand placed points along the road outline and centreline in the image plane. Point pairs (used in equation 34) are selected by selecting points uniformly along the centreline polynomial approximation. The intersections of lines normal to this line at these points with the polynomials representing the road edges gives a set of point pairs. The centreline and road edges are projected at each iteration by projecting these points from the image plane into the latest estimate of the groundplane (using θ_n) and re-calculating the polynomials. Figure 14 below shows the evolution of the three polynomials over a number of iterations with convergence to a reasonable estimate.

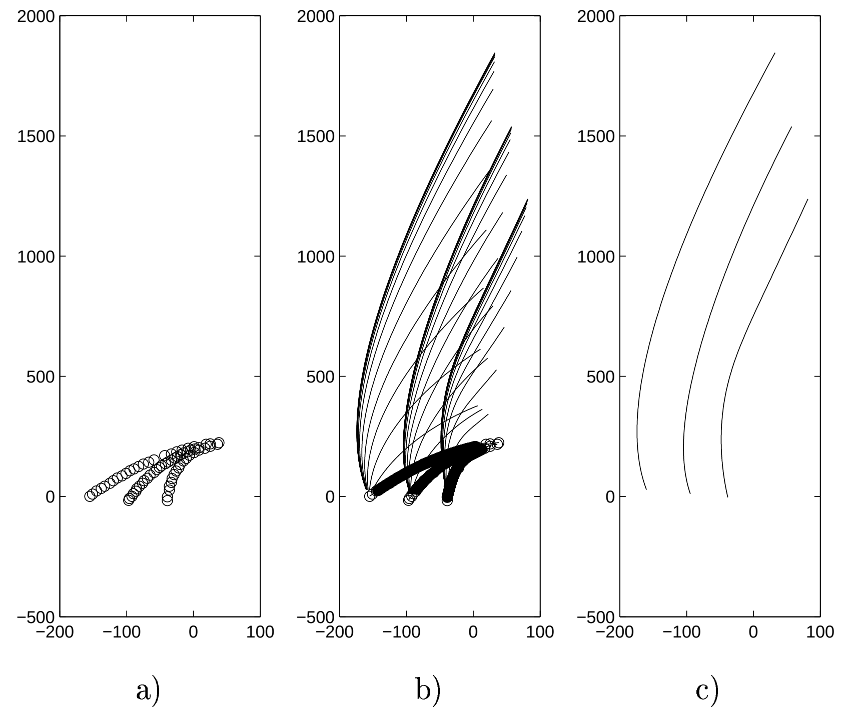


Fig. 14. Iteratively Estimating Projection Parameter using a Constant Road-width Assumption: a) Initial Image Points, b) Iteration, c) Result