# Modelling Pedestrian Trajectory Patterns with Gaussian Processes

David Ellis, Eric Sommerlade* and Ian Reid
Department of Engineering Science, University of Oxford
OX1 3PJ, Oxford, UK
{dre,eric,ian}@robots.ox.ac.uk

## Abstract

*We propose a non-parametric model for pedestrian motion based on Gaussian Process regression, in which trajectory data are modelled by regressing relative motion against current position. We show how the underlying model can be learned in an unsupervised fashion, demonstrating this on two databases collected from static surveillance cameras. We furthermore exemplify the use of model for prediction, comparing the recently proposed GP-Bayesfilters with a Monte Carlo method. We illustrate the benefit of this approach for long term motion prediction where parametric models such as Kalman Filters would perform poorly.*

## 1. Introduction

Visual surveillance systems often observe scenes through which pedestrians follow common motion patterns. In this paper we propose a new scene representation allowing a generative model of motion given a set of observed trajectories. Previous approaches have used parametric methods such as spline fitting to model common pedestrian paths. However, such models are often too restrictive since the trajectories of actors are inherently stochastic, with varying degrees of uncertainty depending on factors such as physical scene structure, the presence of other people and the time of day. The use of Gaussian Processes (GPs) allows us to be explicit about such uncertainties and to adapt to the various complexities of different scenes and situations online.

Figure 1 shows a typical scene of interest[1]. This scene has a number of entrance and exit points and various routes between them are commonly used, with some points being linked by more than one possible route. The aim is to describe the typical motion patterns using these already observed trajectories with a probabilistic model, which can be used for a variety of purposes such as object tracking, semantic labelling of motions and anomaly detection. In this paper we focus on long term motion prediction as a primary example application of the model.

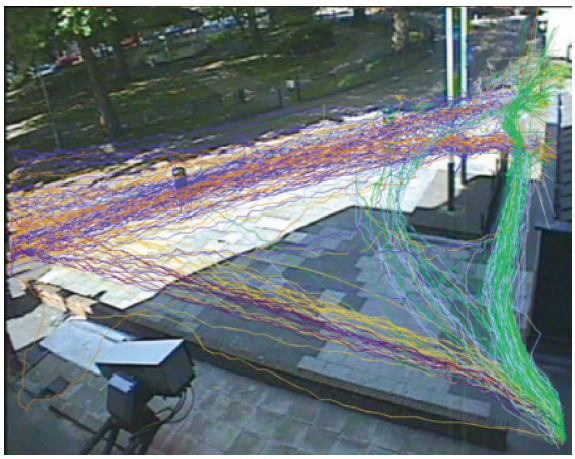


Figure 1. Observed trajectories from a static surveillance camera courtesy of [8]. Colours indicate cluster membership where clustering has been performed based on matching start and end points.

Various approaches have been made to model pedestrian motion patterns from surveillance video. Makris and Ellis[7] define routes using linear splines between entry and exit points in the scene, along with a variance normal to the spline along each segment to represent the path width. This computationally efficient model allows the learning of routes online; however, routes can not handle multiple paths between points and so increasingly complex scenes must be broken down into a large number of smaller paths. Also the one dimensional variance fails to model uncertainty in target speed along the path. Baiget *et al*. [2] also model paths between entry and exit points using slightly more flexible and compact B-splines, but without any notion of uncertainty. They also outline a method for clustering different trajectories based on matching similar start and end points. Johnson and Hogg [5] do not cluster the trajectories initially, but rather compute a point density estimate of the joint probability over object positions and instantaneous ve-

*Parts of this work were supported by an EPSRC studentship and EC grant IST-027110 for the HERMES project in the EU sixth framework programme.

[1]Thanks go to Makris *et al*. for allowing use of the trajectory database which appeared in [8]



2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops
978-1-4244-4441-0/09/$25.00 ©2009 IEEE

locities using a neural network. Hu *et al*. [4] parameterise trajectories at discrete time steps with the distribution over velocities represented as a chain of Gaussians. After clustering using fuzzy K-means, they demonstrate the ability to detect abnormal behaviour and perform long term prediction for road traffic. Both applications require that the object is tracked from the start of the training data trajectories. Ali and Shah [1] have used natural crowd flow as a prior for tracking. They use optical flow (amongst other cues) to build a 'flow field' which describes the general crowd motion at each point in the image plane. This method is focused on analysis of only one dominant motion pattern in the image and is less suitable for scenes with fewer people where actors are relatively free to move where they wish.

We present a probabilistic approach that bridges the gap between the full joint pdf over trajectories estimated in Johnson and Hogg's method [5] with the efficiency of the spline based models [2, 7]. At the core of the model is the use of Gaussian processes to estimate instantaneous velocity of an actor given its current position. We cluster the training trajectories based on the associated entry point into the scene and then build a separate model for each cluster. Therefore instead of estimating the joint probability distribution over object positions and instantaneous velocities as in [5], we estimate the conditional distribution over instantaneous velocities given the current position and cluster membership.

## 2. Gaussian Process Regression

Here we briefly introduce GP regression [10]. A Gaussian Process is a collection of random variables with any subset of them having a joint Gaussian distribution. A GP is fully defined by its mean $m(x)$ and covariance $k(x, x')$ functions. A function $f$ being distributed as such is denoted

$$f \sim \mathcal{GP}(m(x), k(x, x')). \tag{1}$$

This distribution over functions will be used as a prior for inference. It specifies properties of possible functions through the mean and covariance but often it is assumed that $m(x) = 0$ over the whole input space, and the form of possible functions is determined through the choice of $k(x, x')$ and so called hyper-parameters within this covariance function.

If the vector $\mathbf{f}$ contains some observed values of the function $f$ at inputs $X$ and we wish to predict the value of the function $f_*$ at a particular input $X_*$, then we have by the definition of a GP above

$$\begin{bmatrix} \mathbf{f} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K & K_*^T \\ K_* & k_{**} \end{bmatrix}\right), \tag{2}$$

where the following shorthand for the covariance submatrices has been used

$$K = K(X, X) \quad K_* = K(X, X_*) \quad k_{**} = k(X_*, X_*).$$

The posterior distribution over the predicted value is obtained by conditioning on the observed data

$$f_* | \mathbf{f} \sim \mathcal{N}(K_* K^{-1} \mathbf{f}, k_{**} - K_* K^{-1} K_*^T). \tag{3}$$

This gives the predicted mean value $\bar{f}(X_*)$ along with a variance $\mathbb{V}[f_*]$ describing the uncertainty in this prediction. Generalising, we adopt the notation that for a dataset $D = \{X, \mathbf{f}\}$ where the inputs $X = [\mathbf{x}_1, ..., \mathbf{x}_n]$ have scalar targets $\mathbf{f} = [f_1, ..., f_n]$, then the posterior process mean and variance at a test input $\mathbf{x}_*$ are denoted by $\bar{f}(\mathbf{x}_*) = \mathrm{GP}_\mu(\mathbf{x}_*, D)$ and $\mathbb{V}[f_*] = \mathrm{GP}_\Sigma(\mathbf{x}_*, D)$.

## 3. Modelling Clusters of Trajectories

We now focus on the problem of predicting changes in target position based on its current position and that it is following a previously identified cluster of similar trajectories (a path) through the scene. Trajectories are represented by a series of 2D-points $\mathbf{x}_t = [x_t, y_t]^T$ corresponding to measurements taken at discrete time steps $t$. We wish to learn the prediction model $f$ such that

$$\mathbf{x}_t = \mathbf{x}_{t-1} + f(\mathbf{x}_{t-1}) + \mathbf{n}_{x,t-1}, \tag{4}$$

where $\mathbf{n}$ is zero-mean white Gaussian noise. By treating the change in state $\Delta\mathbf{x}_t = \mathbf{x}_t - \mathbf{x}_{t-1}$ as the target variable and considering changes in $x$ and $y$ to be independent, we place a Gaussian process prior over $f \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$ leading to the approximate prediction model

$$\begin{aligned} p(\Delta x_t | \mathbf{x}_{t-1}, D_x) \approx \\ \mathcal{N}(\mathrm{GP}_\mu(\mathbf{x}_{t-1}, D_x), \mathrm{GP}_\Sigma(\mathbf{x}_{t-1}, D_x)). \end{aligned} \tag{5}$$

Here the dataset $D_x = \{(x_i, y_i), \Delta x_{i+1}\}_{i=1...n}$ represents a set of trajectories containing $n$ position and instantaneous velocity observations in total, and a similar but independent model is used to predict $\Delta y$ using $D_y$. The mean of the prior process is taken to be zero which is a reasonable assumption if predictions are only required at test points close to some input values in the training set. For the covariance function we use the squared exponential (SE) with an added bias term given by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}||\mathbf{x}_i - \mathbf{x}_j||_\Lambda^2\right) + \delta_{ij}\sigma_n^2 + b. \tag{6}$$

The SE covariance function enforces smoothness since points that are close together in the input space have highly correlated outputs. This is intuitive for the current application where it would be expected that neighbouring points along a path tend to lead to movements in similar directions. $\Lambda$ is a diagonal matrix containing the length scale hyper-parameters $l_x, l_y$. These dictate the rate at which the correlation decays as a function of distance between the inputs, and so quantify how close two points must be to have

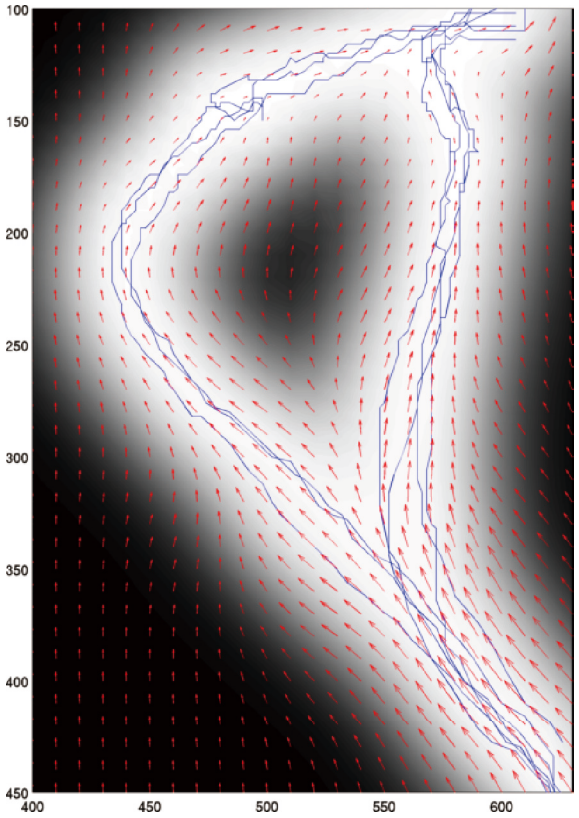



Figure 2. A Gaussian process based motion model. The model is trained using trajectory data shown by solid lines. Arrows correspond to the mean predicted instantaneous velocity at the position of their tails. The background shading indicates the variance of the prediction from each point with darker shading indicating higher uncertainty. Axis labels are in pixels.

correlated responses. $\sigma_f$ determines the expected range of the output (velocities). The bias term $b$ has the effect of correlating all values to some degree, meaning that predictions well away from the data will tend to towards the average velocity rather than the value of $m(\mathbf{x})$ (zero in this case). Finally the noise term in equation 4 is incorporated into the model by the addition of $\sigma_n^2$ along the diagonal of $K$. This term allows for the variation between observed velocities for points that are close in the input space.

As an example, we consider a subset of the trajectories on the right hand side of Figure 1, where people enter the scene from the bottom right, move upwards, deviate either side of the lampposts and then continue upwards to the exit point. The actual data used to form $D_x$ and $D_y$ for the GP model is taken from the trajectories shown in Figure 2. The mean predictions for instantaneous velocity in Figure 2 follow the direction of the path even in the presence of noisy trajectories. More importantly, the uncertainty in the predicted velocity increases further away from the observed data. Notice also in Figure 2 how the mean predictions are in a slightly upwards direction in these areas of high uncertainty. This is due to the bias parameter $b$, the value of which has been inferred, as with the rest of the hyper-parameters, from the data.

### 3.1. Learning the Hyper-Parameters

Setting the values of the hyper-parameters in the covariance function (equation 6) is a way of expressing some prior information about the expected form of the motion model. However, it would be inconvenient to have to tune these parameters by hand for each scene being modelled. Instead, inferences about the hyper-parameters can be made from the observed data itself. By our GP assumption, for observations $\mathbf{f}$ at inputs $\mathbf{x}$ we can express the log likelihood of the data given the hyper-parameters as

$$L = \log p(\mathbf{f}|\mathbf{x}, \theta) = -\frac{1}{2}\log|K| - \frac{1}{2}\mathbf{f}^T K^{-1}\mathbf{f} - \log 2\pi \quad (7)$$

where the hyper-parameters have been collected into $\theta = \{\sigma_f, \Lambda, \sigma_n, b\}$. The maximum likelihood parameters $\hat{\theta}_{ml}$ can be found by optimising equation 7 with respect to $\theta$. This is easily performed using standard optimisation routines (we use iterative conjugate gradients) since the first derivative of the log likelihood w.r.t. $\theta$ is easily evaluated, see [10] for examples.

Unfortunately, $L$ is non-convex and so there is the possibility of reaching local minima corresponding to alternative explanations of the data. For example, a rapidly varying signal generated by a process with a small lengthscale $\Lambda$ can also be interpreted as arising from a very noisy process with high $\sigma_n$. For this particular application we have observed that choosing a starting point for $\theta$, specifically $\sigma_n$ or $\Lambda$, close to a reasonable value is sufficient. For example, starting with a length scale of one metre when the measurement space is on the ground plane produced good estimates for $\hat{\theta}_{ml}$. This step is inevitable since we can not learn the GP model without expressing some prior beliefs.

## 4. Long Term Prediction

Given a current estimate of the pdf of the target position $p(\mathbf{x}_t)$, an estimate for the target position at the next time step $\mathbf{x}_{t+1}$ is

$$p(\mathbf{x}_{t+1}) = \int p(\mathbf{x}_{t+1}|\mathbf{x}_t)p(\mathbf{x}_t)\, d\mathbf{x}_t. \quad (8)$$

The prediction model used is calculated by GP regression as described in the previous section

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_t + \mathrm{GP}_\mu, \mathrm{GP}_\Sigma). \quad (9)$$

To obtain a long term prediction of the position, the above integral can be evaluated recursively. However, this integral has no analytic solution unless $\mathbf{x}_t$ is known exactly. This is due to the dependence of $\mathrm{GP}_\mu$ and $\mathrm{GP}_\Sigma$ on $\mathbf{x}_t$ through

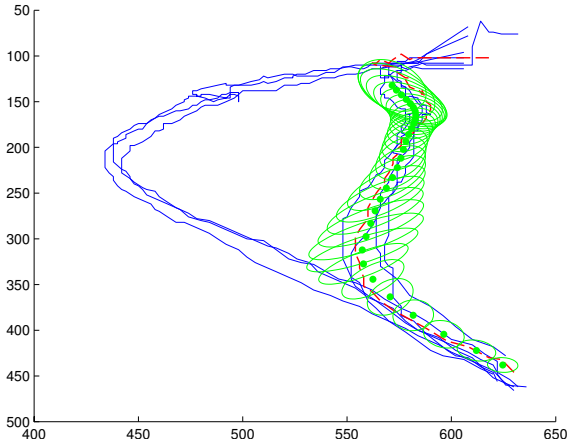



Figure 3. Prediction using a GP-EKF for a target moving from bottom right upwards. The ellipses show the $2\sigma$ bounds on the predicted postion at successive time steps. The solid lines are a sample from the dataset used to train the scene model and are intended to show the possible paths to be taken. The dashed is a trajectory from a test set, the starting point of which was used as the initialisation for the prediction.

the covariance function (see equation 3). In this section, two approximations are discussed which allow long term prediction.

### 4.1. Using GP-Bayesfilters approximations

If it is assumed that $p(\mathbf{x})$ is Gaussian at all time steps, then prediction can be performed using a Kalman filter. Ko and Fox [6] introduced the GP-EKF, an extended Kalman filter which uses Gaussian Process based prediction and measurement models. In the prediction step, the predicted mean $\hat{\mu}_{t+1}$ is given directly by the GP mean function

$$\hat{\mu}_{t+1} = \mathrm{GP}_{\mu}(\hat{\mu}_t, D). \tag{10}$$

The additive process noise is given by the variance of the GP prediction

$$Q_{t+1} = \mathrm{GP}_{\Sigma}(\hat{\mu}_t, D). \tag{11}$$

To estimate the predicted state covariance, the Jacobian of the GP mean prediction with respect to the state is required

$$G_{t+1} = \frac{\partial \mathrm{GP}_{\mu}(\hat{\mu}_t, D)}{\partial \mathbf{x}_t}, \tag{12}$$

which can then be used to propagate the state covariance through the plant model

$$\hat{\Sigma}_{t+1} = G_{t+1}\hat{\Sigma}_t G_{t+1}^T + Q_{t+1} \tag{13}$$

Repeated application of equations 10 and 13 allows an estimate of target postion to be calculated many time steps into the future. Figure 3 shows the propagation of the state

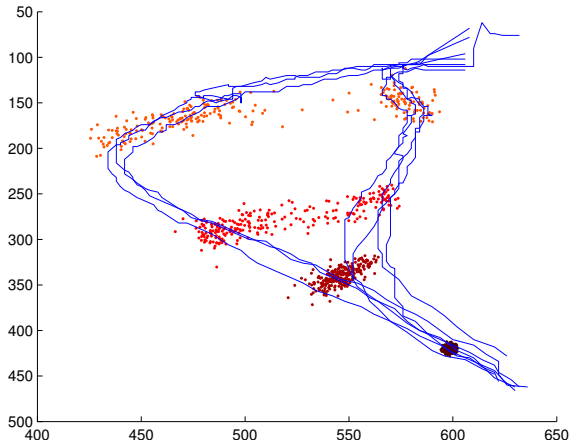



Figure 4. Distribution of particles at four snapshots in time when starting from the estimate shown in the bottom right.

through the motion model depicted in Figure 2 with a starting point in the bottom right corner. The covariance ellipses correctly distort to express the uncertainty in target postion caused by changes in direction of the path. However, the use of a Gaussian state estimate is clearly too restrictive when paths diverge and take different routes, since the unimodal estimate can only predict one of the possible tracks after a junction.

### 4.2. Sequential Monte-Carlo Prediction

The GP scene model is capable of describing complex paths which can diverge and even reconverge, unlike other methods which separate diverging paths into smaller elements. To use the model effectively for prediction, we must allow a multi-modal estimate of the position to be maintained. We use a sampling approach where the distribution $p(\mathbf{x}_t)$ is represented by a set of particles. At each time step, for each particle, the distribution is updated by sampling $p(\Delta\mathbf{x}_{t+1}|\mathbf{x}_t)$, provided by the GP scene model. This is effectively the prediction step of a particle filter, with an $\mathcal{O}(n^2)$ operation per particle due to calculation of the variance in equation 5.

Figure 4 shows how a distribution of 250 particles evolve over a number of time steps, starting with a Gaussian estimate shown in the bottom right corner. The next distribution shows $p(\mathbf{x})$ as the target reaches the junction. In the next step a bimodal distribution begins to form as there are two possible hypotheses as to which route could be taken. Particles in the region between the two tracks disperse quickly because of the high predictive uncertainty in this area where no observations have been made. The last snapshot clearly shows two modes to the distribution with a low probability of the target lying between the two tracks. Notice how there is a fairly high variance in the position of particles on the left hand track, which is due to the variability in the time taken by pedestrians traversing the corner. This effect is

seen again in the analysis of a more complex dataset in the next section.

## 5. Implementation

The previous section used the toy example of bifurcating trajectories extracted from a larger dataset to illustrate the potential power of the GP motion model. We now attempt to model a whole dataset extracted by an overhead camera covering a scene with many entrances and exits in the atrium of a building. Figure 5 shows the dataset under consideration. Since many of the paths cross in the central area, conventional tracking methods often lose targets when many actors are present. Long term prediction can therefore be used to aid in target reacquisition.

Firstly, the trajectories are grouped by start point using a simple mean shift clustering method[3], leading to the cluster assignment shown in Figure 5. The reason for clustering by start point is that the current GP model can only give a unimodal estimate for $p(\mathbf{x}_{t+1}|\mathbf{x}_t)$ (see equation 3), and so can not model crossing trajectories. Clustering by start point exploits the fact that people tend not to walk back on themselves to provide a fairly general model, without introducing the problem of multi-modalities in predictions from a single point.

The trajectories are then subsampled such that position measurements at around 4Hz are used for the GP training data. This is to limit the number of data points used for prediction to around 1000, otherwise calculation of the predictive variance, which has complexity $\mathcal{O}(n^2)$ in the number of data points, becomes computationally too expensive. For long term prediction this is not an issue since much of the discarded data provides little extra useful information. It is much more useful to have a larger sample of trajectories, rather than more points per trajectory, to cover the data space as well as possible with a representative set of common paths.

The hyper-parameters for each cluster are then learned from the corresponding data sets. Following this, the model is ready to be used for prediction. As an example, consider a target which has been observed entering the scene on the left hand side of figure 5. The GP motion model for this cluster of trajectories is shown in Figure 6(a), and long term prediction with 250 particles is shown in figure 6(b). Notice how the distribution spreads to cover all three possibilities. The sharp left corner has the effect of spreading the estimate along the direction of the path due to differences in the time taken for targets to traverse the corner in the observed data. Although there are some particles in the area to the right of the corner, these are actually fairly sparsely distributed compared to the areas around the modes.

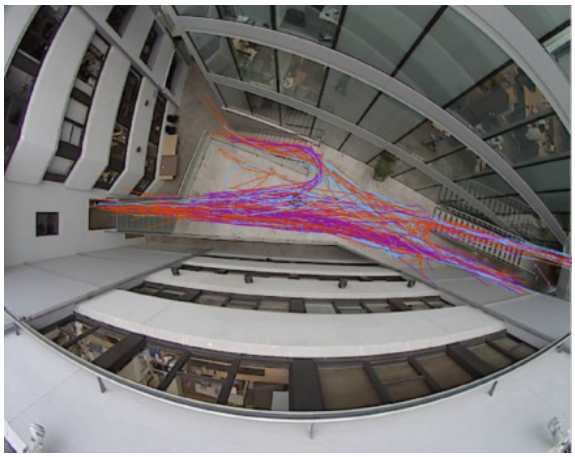


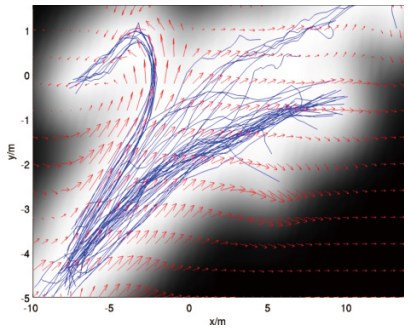
Figure 5. Trajectories observed from a camera over a period of a few hours. The colours of tracks (best seen in the online version) indicate a cluster assignment based on starting point.

### 5.1. On-line Adaptation

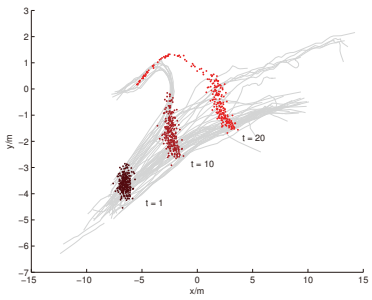
At present each model can only store a few thousand data points due to the quadratic complexity in sampling the long term predictions. By only using the most recently acquired trajectories as a prediction dataset, the model can adapt to changes in the environment. Re-clustering is inexpensive and so can be carried out online while the recomputation of $K^{-1}$ in equation 3 requires one $\mathcal{O}(n^3)$ operation when the dataset changes. The re-learning of hyper-parameters is slower, requiring recomputation of $K^{-1}$ at each optimisation step and is more suited to batch processing offline. This is not a problem, since we would not expect the general properties of the trajectories controlled by the hyper-parameters to change for a given scene. However, actual deviations in routes which are be subject to change can be handled by simply updating the data stored for use in the prediction model.

## 6. Evaluation

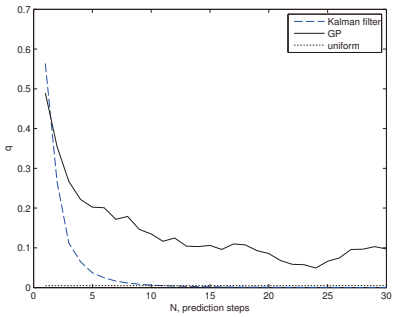
To evalutate the accuracy of the long term predictions made via the Monte Carlo sampling method, we performed leave-one-out cross-validation. For each of the 54 trajectories in the training set shown in Figure 6(a) a motion model is trained, excluding a trajectory from the training set. Long term prediction with 250 particles is used to predict the path after the $t = 5$ point along this trajectory. The observation space is divided into a grid of $0.5\text{m}^2$ cells. At each step $N$ of prediction the fraction of particles residing in the correct cell is recorded. A standard constant velocity model Kalman filter, with the same parameters as used for collecting the trajectory data, is also left to perform open loop predicition. The predicted state estimate is integrated over the correct cell for direct comparison with the correct particle fraction.




(a) Motion model for 54 trajectories starting on the left of figure 5. As in Figure 2, note how the bias $b$ in eqn 6 lets distant points assume non-zero velocities.




(b) Long term prediction showing the particle distribution estimate of $p(\mathbf{x}_t)$ for three snap shots.




(c) Comparison of long term prediction performance of the GP scene model (solid line) and a Kalman Filter (dashed). $q$ is the cumulative distribution over the correct cell where the target actually resides, and the value shown is averaged over all trials.

Figure 6(c) shows the average value of the probability of being correct over all the trials. The Kalman filter predictions quickly become too vague as the prediction covariance rapidly increases without any new measurements. The GP model makes use of the previously observed trajectories to make a more informative prediction, leading to a higher probability being assigned to the correct cell. Even after 30 time steps (7.5s), on average 10% of particles still lie within the correct cell. However, the Kalman filter estimate becomes less informative than a uniform distribution over the area under surveillance.

## 7. Conclusion and Future Work

We have presented a novel method for modelling common pedestrian motions through a scene and, using two different datasets, have given examples of how the model can be used to perform long term target position prediction. The key benefit comes through the use of Gaussian processes to explicitly model uncertainty in predictions such that the characteristic unpredictability in human motion can be accurately represented.

In future work we aim to use this prediction model directly to aid in planning a target re-acquisition strategy for a single active (pan, tilt, zoom) camera. It has been noted that each model is limited by the complexity of GP regression to contain only a few thousand data points. We are currently investigating the use of both global [9] and local [11] sparse approximations to allow the use of trajectory data over longer periods. Also, at present outliers (such as a trajectory going in the opposite direction to the others in a cluster) cause large errors in prediction and can confuse hyper-parameter learning. These can be avoided by using more advanced clustering algorithms, however it would also be interesting to compute an estimate of the joint distribution $p(\mathbf{x}, \Delta\mathbf{x})$ (*i.e.* incorporate velocity information into the inputs), which should alleviate the crossing paths problem, and then perform clustering over this space.

## References


[1] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *ECCV '08: Proceedings of the 10th European Conference on Computer Vision*, pages 1–14, Berlin, Heidelberg, 2008. Springer-Verlag.

[2] P. Baiget, E. Sommerlade, I. Reid, and J. Gonzàlez. Finding prototypes to estimate trajectory development in outdoor scenarios. In *Proceedings of the First International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS2008)*, September 2008.

[3] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[4] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1450–1464, Sept. 2006.

[5] N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing*, 14(8):609 – 615, 1996.

[6] J. Ko and D. Fox. GP-BayesFilters: Bayesian filtering using gaussian process prediction and observation models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3471–3476, Sept. 2008.

[7] D. Makris and T. Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20:895–903, 2002.

[8] D. Makris, T. Ellis, and J. Black. Bridging the gaps between cameras. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).*, volume 2, pages II–205–II–210 Vol.2, June-2 July 2004.

[9] J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *J. Mach. Learn. Res.*, 6:1939–1959, 2005.

[10] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, December 2006.

[11] E. Snelson and Z. Ghahrami. Local and global sparse gaussian process approximations. *Artificial Intelligence and Statistics 11 (AISTATS)*, 2007.