Track Anomaly Detection with Rhythm of Life and Bulk Activity Modeling

R. O. Lane
QinetiQ
Malvern, UK
rlane1@QinetiQ.com

K. D. Copsey
Digital Barriers
Guildford, UK
Keith.Copsey@digitalbarriers.com

***Abstract*—This paper describes a model and algorithm for detecting anomalies in track data. The algorithm is general in the sense that it can be applied to tracks from any type of sensor. Two important enhancements to the standard algorithm are outlined. The first of these characterizes predictable temporal variations in behavior, the so-called rhythm of life. This allows the detection of unusual activity during one part of the day that would be considered normal at other times. The second development analyzes the bulk behavior of targets. Although tracks on their own may not be unusual, their combined actions could be suspicious. Also, bulk activity analysis allows the detection of missing expected behavior, which is not possible with many techniques. Application of the algorithms is demonstrated by analyzing the movement of people in a canteen.**

Keywords-anomaly detection; bulk activity analysis; gaussian mixture model; pattern of life; rhythm of life.

I. INTRODUCTION

In recent years there has been a heightened interest in security, which has led to the installation of a multitude of sensor-based surveillance systems. However, it is not possible to manually interpret the large amounts of available data in real time. This problem is likely to get worse in the future with an increasing availability of low-cost sensors. To avoid operator fatigue and overload, automated systems are required to help analyze the data.

At a glance, adversary activities may be difficult to distinguish from background activity. However, if it is possible to build an accurate model for normal patterns of life, then deviations from this model could indicate possible suspicious behavior. Anomaly detection algorithms automate this process and can be integrated into visual analytics or decision support systems. Rather than being presented with a screen showing hundreds or thousands of tracks, operators' attention will be focused on, for example, the five most suspicious individuals. Unattended systems could sound an alarm calling for human inspection.

The applications of anomaly detection algorithms range from countering piracy and organized crime, to insurgency and terrorism prevention. To maximize the effectiveness of a surveillance system a mix of sensor modalities is necessary. This is the case whether it is required to monitor activities near sensitive national infrastructure, such as airports, train stations, and harbors, or in military scenarios such as force protection [1][2].

The need to develop a large number of sensor-specific algorithms can be avoided by transforming the data into a common reference frame or format. One method for doing this is to run target detection algorithms on processed data and associate detections over time to form target tracks. Tracks can be defined in sensor coordinates or mapped to real-world coordinates. This paper proposes a generic track-based anomaly detection algorithm, including a method for modeling temporal variations in data. In addition to detecting anomalies in individual tracks, a process for analyzing the bulk behavior of groups of objects is described. Although tracks on their own may not be unusual, their combined actions could be suspicious. Also, bulk activity analysis allows the detection of missing expected behavior, which is not possible with many algorithms.

The remainder of this paper is structured as follows. Section II outlines related work in the field of track anomaly detection. Section III develops a track segment mixture model, including enhancements to characterize temporal variations in behavior and bulk activity. Section IV applies these algorithms to data and conclusions are given in section V.

II. RELATED WORK

A substantial amount of work has been carried out on track anomaly detection over the last ten years. The following discussion reviews various approaches, categorized by the type of algorithm.

A. Mixture Models

A pioneering example of mixture modeling for video surveillance tracks is contained in [3]. The feature vector consists of the history of recent target positions and the change in position at the latest time step. The density of these features is represented by a Gaussian mixture model (GMM). This allows the detection of anomalies through likelihood comparisons as well as the prediction of future behavior for partial tracks. There are many other examples of mixture models being used to model tracks. For instance, three separate GMMs can be used to model video-target position and velocity, length of track, and time spent while stationary [4]. Rather than estimating GMM parameters on batches of data, learning can be done online so that the model continually

This work was partially funded by the UK MOD

adapts to new data [5]. The model is improved by allowing new mixture components to be created or similar components to be merged. In a slight variation on track characterization, [6] splits feature vectors of car tracks into spatial and motion components, where the location of a target provides an index into a set of mixture models for the motion variables. Reference [7] uses adaptive kernel density estimation (KDE) to model the position and velocity of ships. This can be viewed as a type of GMM, with one mixture component per data point.

B. Markov Models

Markov models are used in a number of studies, where the target is assumed to exist in one of a finite set of discrete states. At each time step there is a certain probability of changing to each of the other states. In some models the state is not directly measurable but other variables that are a function of the state can be observed. One example of a hidden Markov Model (HMM) algorithm for ship anomaly detection is given in [8], where the hidden states are "cruising" or "maneuvering" and the observables are changes in measured variables such as speed and heading. Tracks with a low likelihood are declared as anomalous. In another maritime anomaly detection application, the series of ports visited by ships is characterized using a Markov model and unusual patterns of visitation can be detected [9]. A method for detecting slowly evolving anomalies in repetitive movements is symbolic dynamic filtering [10]. The state space described by a continuous variable and its derivatives is divided into separate regions to form discrete states. The system is monitored over a period of time for both normal and test conditions and high-order Markov models are trained on the data. Anomalies are declared when the state probability vector of the test model deviates significantly from the normal model. Reference [4] includes Markov modeling in a video surveillance application. Target attributes such as time, location and velocity are segmented to form a discrete state space. Other dimensions corresponding to the relationships between objects are also modeled.

C. Other Static Models

Several other approaches to modeling track data have been proposed. For example, [11] examines radar tracks of vehicles and classifies target behavior into nine activities. Sequences of activities are analysed using string processing techniques to recognize higher-level behaviors. Tracks are assigned to roads using map information and a set of rule-based anomalies are defined based on the high-level behavior and location. Reference [9] models standard ship routes as branches on a network of nodes defined by ports and coastal points. Deviation of measured tracks from ideal branches or the traversal of a non-standard sequence of branches indicates unusual behavior. Node locations can be determined from coastal maps and port databases or learnt from course changes and stoppages in the track data [12]. A Bayesian network can be used to combine this type of detector with other anomaly spotting algorithms, such as close approach detection and predicted entry of an alert zone, to provide an overall threat assessment [9].

The need to specify a parametric model for target motion can be avoided by using histograms on a discretized state space. The video motion anomaly detection system in [13] models velocity as speed and heading rather than using Cartesian coordinates as this allows a finer resolution at slow speeds without resorting to a complicated division of state space. Each location in the image has a separate velocity histogram to analyze target behavior. When a model depends on location, efficient tree-based spatial indexing systems can be used in large scale surveillance systems to speed up the calculation of parameters [14].

Strong assumptions about the distribution of data can be avoided using several methods. Support vector machines can be used to classify data as being either above or below a defined percentile of data normality, with the reporting of abnormal tracks to the user [15]. Another distribution-free algorithm is conformal anomaly detection [16]. Instead of estimating model parameters, this algorithm compares test tracks with historic tracks using a distance measure. New tracks that are sufficiently far from previously seen ones are declared as anomalous.

Visual analytics schemes avoid the need to design automated anomaly detection algorithms. Reference [17] outlines a system that performs various forms of data processing and provides selected ways for this to be presented on a screen. A human analyst can then examine the data and form their own situation awareness picture mentally.

D. Rhythm of Life Models

In general, the nature of vehicle or person activity is not constant over time; variations can depend on time of day, day of the week or time of year. For example, behavior that is considered normal during the day could be unusual if it is detected as night. It has been recognized that a single model is not appropriate for all times, see [5][8][17] for example, but explicit methods for modeling varying behavior are not often put forward. We mention here three cases where temporal change has been modeled. In an airport security application [4], time is represented as an extra dimension in the discrete state space. This allows events to be detected at different times at the expense of having a large number of state-space regions. Sports team performance analysis is carried out in [18]. A training set of player tracks for a particular team is manually labeled at each time instance, dividing the game into offense, defense, and time out phases. A GMM is used to model the data and new track sets are classified as belonging to one of these categories before further analysis takes place. Reference [19] learns daily traffic patterns in a communications network. If data relating to a particular day does not match one of the patterns it is labeled as anomalous.

III. TRACK SEGMENT MIXTURE MODEL

A. Basic Algorithm

A target trajectory is modeled as a time series $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T$, where $\mathbf{x}_t$ is the state of the target at time t. The examples presented in this paper use a 2D position state vector but in general, the state could include 3D position and velocity, and, depending on the sensor type, other variables such as transmitted frequency. The process is facilitated by modeling whole trajectories as a series of overlapping fixed-length track segments $\underline{\mathbf{x}}_t=\mathbf{x}_t,\mathbf{x}_{t-1},\ldots,\mathbf{x}_{t+1-h}$.

The statistical model of a trajectory takes the form of a Gaussian mixture model. The probability density function of a track segment $\underline{\mathbf{x}}$ is

$$p(\underline{\mathbf{x}}) = \sum_{r=1}^{R} w_r N(\underline{\mathbf{x}};\mu_r,\Sigma_r) \ , \qquad (1)$$

where R is the number of mixture components, w_r are the mixture component probabilities, and $N(\underline{\mathbf{x}};\mu_r,\Sigma_r)$ is the Gaussian density of the rth component with mean μ_r and covariance Σ_r. This model is similar to that used in [3] except that here, the first element of the state history vector is the latest target state rather than the difference between the latest state and the previous one.

The training phase formats complete trajectories from a historical database into track segments and uses them to estimate parameters of the model using the expectation maximization (EM) algorithm. The EM algorithm finds a local maximum of the likelihood function. To avoid convergence on a minor local maximum, the algorithm is run several times with randomly chosen starting points and the most likely parameter set is chosen. Details of the EM update equations can be found in [3].

The test phase compares new tracks to the model and if their likelihood is below a certain threshold they are flagged to the user as being anomalous. The algorithm does not have to wait for the tracks to be complete because the latest available track segment is used as an input to the algorithm. If it is desired to achieve a constant false alarm rate (CFAR), the likelihood threshold can be set adaptively by fitting a statistical model to the likelihood values of the most recent track segments. The threshold is calculated as the inverse cumulative distribution function of the requested false alarm rate. The model can be initialized by simulating track segments from the model and calculating their likelihood. For real-time operation, the CFAR model should then be updated in an online manner when a new track segment is received, rather than recalculating using all the data.

B. *Rhythm of Life*

A shortcoming with the single statistical model for trajectories described in the previous section is that it ignores temporal changes in the "rhythm of life"; normal patterns of behavior change with the time of day and the day of the week. For example, when considering a large office building most people enter the building at the start of the working day, and leave at the end of the day. A method for dealing with this is to have a suite of different models encompassing the time-varying patterns of behavior that occur.

The use of different models provides a number of advantages over attempting to build a single model that encompasses all patterns of behavior. Most importantly, a single model cannot allow for the fact that normal behavior in some time periods would be suspicious behavior in other time periods. For example, normal day-time behavior might be considered anomalous if taking place in the middle of the night, and someone entering a building at the same time as everyone else is leaving may also be suspicious. Secondly, use of a single large model can be computationally costly and difficult to train.

The models nominally cover a non-overlapping set of time periods. However, real-world entities do not undergo a sharp change in behavior at model boundaries and there may be some uncertainty as to the exact time of change between models. To accommodate this, a transition period at the beginning and end of each individual model period is defined where the assumed model is a linear combination of the two models either side of the transition phase. Thus the density model at time t is

$$p(\underline{\mathbf{x}}) = \frac{t_{i+1}-t}{t_{i+1}-t_i} M_i(\underline{\mathbf{x}}) + \frac{t-t_i}{t_{i+1}-t_i} M_{i+1}(\underline{\mathbf{x}}) \qquad (2)$$

where $M_i(\underline{\mathbf{x}})$ is the density of the ith model, t_i is the start of the transition period and t_{i+1} is the end of the transition period.

The rhythm of life algorithm proceeds in a similar manner to the basic algorithm. When historical track segments are processed, they are first assigned to the appropriate model depending on their time stamp. If the time stamp falls within a transition period, the track segment is assigned to one of the two models with a probability equal to the appropriate factor in (2). Once sufficient data has been allocated, each model is trained individually using the EM algorithm. During the test phase, the time stamp of track segments is used to select the correct model. If the time is within a transition period, (2) is used to determine the model density. Anomaly detection based on the likelihood of track segments is then carried out as before.

It may be the case that larger-scale shifts in the behavior-regime transition time occur due to extreme weather events or delays in transport systems, for example. To deal with this situation we outline a more adaptive approach to model transitions based on online monitoring of the fit of the models to the observed trajectories. As the fit becomes poorer the model is switched to an alternative one. By combining such an approach with monitoring of the pre-defined scheduled model we can adapt automatically to the rhythm of life.

It is assumed that during initial model training, an operator has identified the expected time sequence of models, known as the *a priori* or predicted schedule of models. Also, for each model we identify the potential successor models (behavior regimes which are likely to follow next), and the likely confusing or overlapping models. Then for any given time period during deployment we will have:

- The current active model denoted M_s.
- The pre-identified model for that time period, which may be the active model. This is denoted model $M_{s(0)}$.
- The previous active model. This is denoted model $M_{s(-1)}$.
- A set of the potential successor and confusing models. These are denoted as models $\{M_{s(i)}, i = 1,\ldots, n(s)\}$, where $n(s)$ is the number of such models.

Only the above models are considered in the analysis. The reason for not considering the whole set of models alongside the current active model is that the reduction in the model set reduces the computational cost of the procedure, and therefore allows a larger total set of models to be considered for finite computational cost. There is also the possibility that unexpected switches between different patterns of behavior are in themselves anomalous, and therefore of interest to the operator. Hence, such behavior should be indicated, even though a model for the trajectories may exist. The model succession lists could easily be adapted as time progresses.

The key to switching between models lies in monitoring the model fits over a time window. The model fits are given by the total log-likelihoods over the windowed data. The overall anomaly detection scheme is based upon identifying a few low-likelihood scores that need to be flagged to the operator. However, as the number of low-likelihood scores increases over a given time period, rather than observing an increasing number of anomalies, a more likely hypothesis is that the model is failing. It is that observation that underlies the switching procedure.

The following procedure is adopted:

1. Whilst deploying the system and declaring anomalies according to the procedures outlined in the previous section using model M_s, collect a window of the data measurements. When the window is full move to step 2.
2. Calculate the fit, in terms of total log likelihood, of the windowed data for the current active model M_s. If this fit is above a threshold then continue with that model, and move to step 7. If the fit is below the threshold then move to step 3.
3. Calculate the fit for the pre-identified model $M_{s(0)}$ and compare it to a threshold. If the fit is above the threshold select it as the new active model, and move to step 7, otherwise move to step 4.
4. Calculate the fits for the potential successor models $\{M_{s(i)}, i = 1,\ldots, n(s)\}$. Compare each fit to a threshold. Where more than one model is above the threshold select the model giving the best fit as the new active model. All above-threshold models are then added to the list of potential successor models for the new active model, and we move to step 7. If no likelihoods are above the threshold then move to step 5.
5. If the time since the last model change is below a threshold, calculate the fit for the previous active model $M_{s(-1)}$. If the fit is above the threshold then we select the model as the new active model, and move to step 7. Otherwise we move to step 6.
6. Alert the operator that no models provide a good fit to the data, select the least bad model as the active model and move to step 7.
7. Reset the data windowing, and return to step 1. In the extreme case, resetting the data window would involve removing just the first measurement in the window. However, for computational considerations a rolling window is unlikely to be appropriate, and separated non-overlapping windows could be used.

A further enhancement to the procedure can be adopted if the pre-identified schedule is expected to be fairly reliable. In such cases we switch from the current active model to the pre-identified model whenever it gives rise to a better fit.

To avoid the risk of ignoring large scale anomalies, each change of model should flagged to the operator, who is then aware of the changing environment. In fact, the selected models themselves may be of interest. Instances where the selected model does not agree with the predicted or *a priori* model may in themselves be anomalous, and should be presented to the operator for examination.

C. *Bulk Anomaly Detection*

The algorithms in the previous two sections are methods for analyzing the behavior of individual tracks of objects. Tracks may not be suspicious on their own but their group behavior could be considered as suspicious. For example, there may be a lot more people than normal on a road because they are fleeing an incident in a town. One may also normally expect a certain level of traffic on a particular road but find there are no people in the test data for that path. This could be because the local population knows an improvised explosive device (IED) has been planted there and they want to avoid it. The latter form of behavior requires a “lack of tracks” detector for this change to be noticed. For both forms of behavior change, groups of tracks must be analyzed over a period of time.

The bulk activity of targets is measured by examining the extent to which each component of the track segment mixture model describes the set of test data for a specified time period. The proportion of each component that explains a particular track segment is

$$u_{ij} = \frac{p_i(d_j)}{\sum_{i=1}^{n} p_i(d_j)} \quad , \tag{3}$$

where i=1,…,n is the component index, n is the number of components, d_j is the jth track segment and p_i is the probability density function of the ith component. At this stage of the calculation, knowledge of the component weights is not required. The effective number of times each component is used by a data set consisting of m track segments is thus

$$u_i = \sum_{j=1}^{m} u_{ij} \quad . \tag{4}$$

Assuming the data is correctly described by the mixture model, the distribution of u_i can be approximated by the binomial distribution $u_i \sim Bin(m,w_i)$ where w_i is the weight of the ith mixture component. Note that u_i is a continuous variable with $0<u_i<m$ but the binomial distribution is defined for discrete values. This is dealt with by rounding u_i away from the mean of the distribution $w_i m$ to the nearest integer. Since m is large, the discretization effect is small and tests on simulated data show that the approximation holds well. To determine

whether or not a measured value of u_i is anomalous, false alarm probabilities p_U and p_L are set for the upper and lower bounds of normal levels of activity, and a confidence region for values of u_i is calculated from the inverse binomial cumulative distribution function (CDF) of the probability-value (p-value) range $[p_U, 1\text{-}p_L]$. If u_i is greater than the confidence region maximum, this indicates the *i*th component has been used significantly more than expected and there has been excess activity in this area. If u_i is less than the confidence region minimum, this indicates the component has been used significantly less than expected and there is a missing set of expected activity.

Since testing each component of the mixture model gives another chance to generate a false alarm, the upper or lower false alarm probability for each component must be set to give an overall false alarm probability p_{TU} or p_{TL} for the whole model. This is calculated using:

$$p_{\{U,L\}} = 1 - \left(1 - p_{T\{U,L\}}\right)^{1/n} . \qquad (5)$$

As with single tracks, the bulk activity of targets varies with time. Thus the set of models generated by the rhythm of life algorithm should be used when testing new data for bulk anomalies.

Although the above process gives the desired false alarm rate for data that exactly matches the model, when data deviates from the model, even slightly, the false alarm rates can be much higher. This is understood by observing the CDF of p-values calculated from real data. There is a significant departure from the ideal uniform distribution, and this is particularly the case for p-values near 0 or 1. The achieved alarm rate is much higher than that requested, and the percentage difference increases as the requested alarm rate is reduced. Since low alarm rates are the regime operators are interested in, this presents a problem. The proposed solution is to use CFAR processing to reduce the number of alarms. The distribution of measured p-values is thus modeled by a beta distribution, the parameters of which can be estimated using recent past data. This distribution provides a good fit to measured data. The inverse CDF of the beta distribution is used to convert the requested total false alarm probability to an effective total false alarm probability p_T. This is then used in equation (5) to calculate individual component false alarm rates and set thresholds to detect anomalous activity for those components.

IV. Application to Data

A. Canteen Data Set

A representative set of movements of people through the canteen at QinetiQ Malvern has been used to develop and test the algorithms presented in this paper. The canteen data set comprises a series of tracks manually recorded by a human observer over a period of about six hours and transcribed into digital format for computer processing. An example from this data set is shown in Fig. 1 for five types of behavior out of 33 manually determined types.

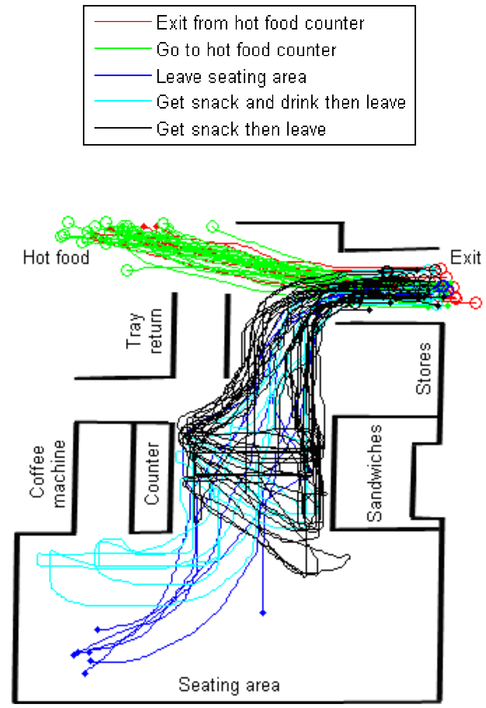

Figure 1. Five typical behavioral movement types in a canteen

For processing purposes, the finely sampled tracks of Fig. 1 were sub-sampled to generate track points that represent detections at fixed time intervals. The original tracks did not have time information recorded so the sub-sampling process assumed a constant speed and track-point separation of two seconds. In a deployed system, the tracks have time information and this step is unnecessary.

B. Rhythm of Life

The rhythm of life algorithm was tested by splitting track data into two sets of approximately 500 tracks each – one for training the model and another for testing it. The model was trained using five track points per segment and 50 mixture model components. The mixture model component centers of the trained models for lunch and non-lunch times are shown in Fig. 2. It can be seen that people only go to the hot food counter and tray return area at lunch times. Other areas of the canteen are visited at all times of the day. For comparison with the basic algorithm, a model was generated by grouping both the lunch and non-lunch tracks into one data set. The resulting mixture model is similar to the combination of the two rhythm-of-life models.

The track segment mixture model algorithm is able to characterize various types of behavior. For example, fast movement is modeled by stretched-out track segments whereas slow movement is represented by segments with closely bunched track points. Other types of movement, such as turning, are also modeled.

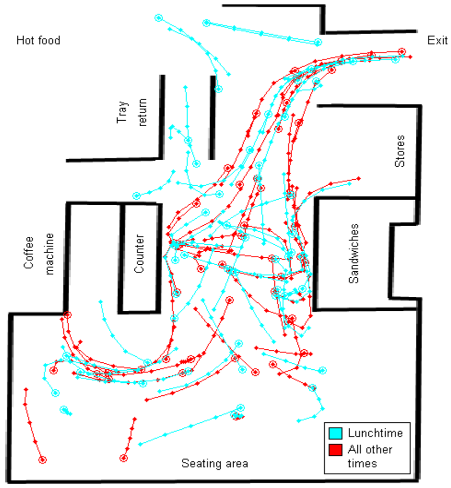

Figure 2. Five-point track-segment mixture model component centres

For testing, a set of artificially generated anomalous tracks were inserted into the data. One of the tracks represents someone scoping out the seating area in detail. Two other tracks represent someone placing an IED in a bin near the tray return area and then immediately going to the exit. These tracks were inserted into the lunch-time data. A number of normal lunch-time hot food counter and tray-return tracks were artificially added to the non-lunch time data. Other than these anomalies, the test data were similar to the training data. Once all the complete test trajectories were converted to track segments, 13,956 segments were available to test the model.

Fig. 3 shows the results of applying the anomaly detection algorithm to the test data during the lunch period, when trained using a whole-day model. In this figure grey tracks are the test tracks, red tracks are known anomalies that are detected, and blue tracks are naturally occurring tracks that have been flagged as anomalous by the system. Some of the blue tracks are false alarms, but most are genuine anomalies that have been discovered. The results achieved when testing this data against the lunch-time only model are similar to those in Fig. 3. This is because all types of behavior in the test set other than deliberate anomalies are present in the both the lunch-time and whole-day models.

Fig. 4 shows the results of the non-lunch test data compared to the whole-day model. A number of natural anomalies have been detected, which relate to tracks whose trajectories are somewhat jagged in certain places. However, the tracks near the hot food counter have not been detected even though they are unusual for this time of day.

Fig. 5 shows the result of applying the algorithm to the non-lunch test tracks using the rhythm of life model. Here we see that the tracks to the hot food counter and tray return area are correctly marked as anomalous.

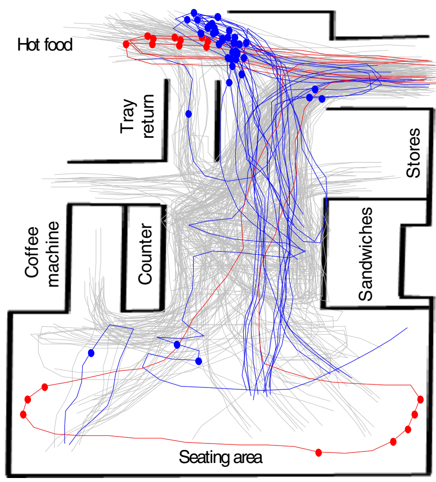

Figure 3. Tracks and multi-model anomalies at lunch time. Red: inserted anomalies, blue: discovered anomalies, grey: other tracks.

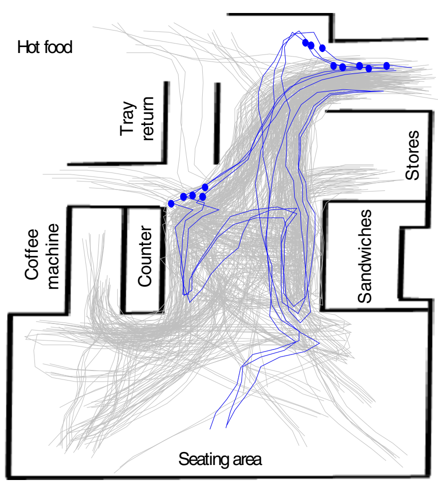

Figure 4. Tracks and single-model anomalies at times other than lunch. Blue: discovered anomalies, grey: other tracks.

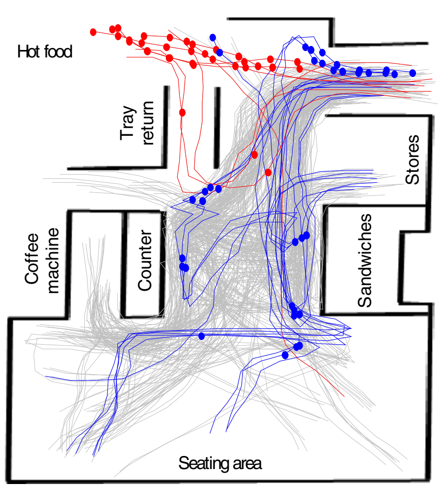

Figure 5. Tracks and multi-model anomalies at times other than lunch. Red: inserted anomalies, blue: discovered anomalies, grey: other tracks.

Results for the automatic model switching procedure are now given. For the testing of this algorithm the data has been divided into three different behavior regimes:

1. Majority of people going to the coffee seating area.
2. Majority of people entering and leaving the main hot food eating area.
3. All activity due to canteen staff at the end of the day.

The training phase consisted of learning a mixture model separately for each of the three regimes, using pre-separated training data examples from each of the regimes. The number of mixture components was set to 30.

Assessment of the model switching procedure requires simulation of a schedule for the observed test trajectories. The schedule divides the test data into a sequence of segments, with each segment containing trajectories from one regime only. The schedule used here selects trajectories from regimes 1, 2, 3 and 1 again in that order.

The pre-identified *a priori* schedule approximately follows the actual test schedule for the first three segments, switching slightly early in both cases, but is erroneous for the last segment, using regime 2 instead of regime 1. In other words, the *a priori* behavior regime for the fourth segment is not the actual behavior regime. There is therefore a requirement on the algorithm to switch to an unexpected regime for the fourth segment. The actual schedule is not made available to the algorithm.

The model switching results are shown in Fig. 6, which demonstrates the ability of the algorithm to switch between the models whilst monitoring the test data. The graph shows the actual regime in green, the pre-identified schedule of models in red, and the algorithm's estimate of the model regime in blue. The algorithm is able to switch to the correct regime in all three cases, albeit with a time lag at each transition. The time lags occur because a sufficient number of trajectories defined by the algorithm window need to have been collected before a change can be detected. Smaller assessment windows could be used, but would be more prone to erroneous model jumping.

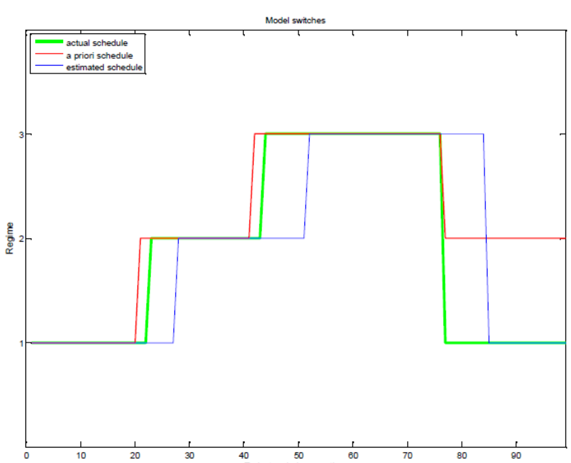

Figure 6. Model transitions in adaptive rhythm of life modeling

C. *Bulk Anomaly Detection*

To analyze performance of the bulk activity anomaly detection algorithm, the test data used in the previous section had one set of behaviors removed relating to movement from the entrance to the hot food counter. The requested false alarm probability for both excess and lack of activity was set to 1%. The tracks were randomly assigned into 20 subsets of 500 tracks each, and for each subset the anomalous components were reported. The results were that every subset correctly reported the missing behavior, no missing-behavior false alarms were generated, and seven excess-activity false alarms were generated. This shows that the missing-behavior process appears to be working as designed but the number of excess-activity false alarms is higher than expected.

It should be noted that when there is a lack of activity in one area associated with one mixture component then the proportion of data assigned to all other mixture components will increase slightly. This may explain the asymmetry in false alarms described above. The issue could possibly be addressed by using a multinomial approximation instead of a binomial one.

It should be noted that if track data arrives in batches from different sources, there may be long gaps between some components being observed. Thus if the analysis time window is too small, these components would be flagged as anomalous. Therefore the performance of the algorithm using real data should be found as a function of the window size. The training data requirements for the bulk activity algorithm are similar to those for rhythm of life.

V. CONCLUSIONS

We have outlined a method for detecting anomalies in both individual tracks and groups of tracks. Temporal variations in behavior are characterized using separate models for different time periods and the algorithm is able to select the appropriate model automatically by monitoring the fit between models and recently received tracks. The approaches have successfully been demonstrated on tracks relating to the movement of people in a canteen area. However, the design of the algorithms is such that they could be applied to any type of source data that is able to provide tracks.

ACKNOWLEDGMENTS

The author thanks colleagues at QinetiQ Malvern for their help in carrying out the work that led to the writing of this paper. Richard Penney implemented a real-time version of the basic algorithm and Anna Skeoch enhanced this to include the rhythm of life modifications.

REFERENCES

[1] S.J. Rouse, R.I. Young, B.D. McGrath, "Airborne multisensor demonstrator program for persistent wide-area surveillance", Proc. SPIE 7694, 76940A , April 2010.

[2] S.P. van den Broek, P. Hanckmann, M. Ditzel, "Situation and threat assessment for urban scenarios in a distributed adaptive system", 12th International Conference on Information Fusion, Chicago, IL, USA, 5-8 July 2011.

[3] N. Johnson and D. Hogg, "Representation and synthesis of behavior using Gaussian mixtures", Image and Vision Computing, 20:889-894, 2002.

[4] C. Brax, L. Niklasson, R. Laxhammar, "An ensemble approach for increased anomaly detection performance in video surveillance data", 12th International Conference on Information Fusion, Seattle, WA, USA, 6-9 July 2009.

[5] D. Garagic, B. J. Rhodes, N. A. Bomberger, and M. Zandipour, "Adaptive mixture-based neural network approach for higher-level fusion and automated behavior monitoring", NATO Workshop on Data Fusion and Anomaly Detection for Maritime Situational Awareness, La Spezia, Italy, 15-17 September 2009.

[6] P.L.M. Bouttefroy, A. Bouzerdoum, S.L. Phung, A. Beghdadi, "Abnormal behavior detection using a multi-modal stochastic learning approach", Int. Conf. on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Sydney, Australia, December 2008.

[7] B. Ristic, B. La Scala, M. Morelande, N. Gordon, "Statistical analysis of motion patterns in AIS data: anomaly detection and motion prediction", 11th International Conference on Information Fusion, Cologne, Germany, July 2008.

[8] M. Andersson, R. Johansson, "Multiple sensor fusion for effective abnormal behavior detection in counter-piracy operations", International Waterside Security Conference (WSS), Carrara, Italy, November 2010.

[9] R.O. Lane, D.A. Nevell, S.D. Hayward, T.W. Beaney, "Maritime anomaly detection and threat assessment", 13th International Conference on Information Fusion, Edinburgh, UK, 26-29 July 2010.

[10] C. Rao, A. Ray, S. Sarkar, M. Yasar, "Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns", Signal, Image and Video Processing 3(2):101-114, 2009.

[11] S.-K. Kim, A. Tsourdos, B. A. White, "Behavior monitoring on ground vehicles by UAVs", Mathematics in Defence 2011, Shrivenham, UK, 20 October 2011.

[12] M. Vespe, K. Bryan, P. Braca, I. Visentini, "Unsupervised learning of maritime traffic patterns for anomaly detection", 9th IET Data Fusion & Target Tracking Conference (DF&TT'12), London, UK, 16-17 May 2012

[13] E. Turkbeyler, C. Harris, "Anomalous movement detection from aerial surveillance", 7th EMRS DTC Technical Conference, Edinburgh, UK, 13-14 July 2010.

[14] J. Will, L. Peel, C. Claxton, "Fast maritime anomaly detection using Kd-tree Gaussian processes", Mathematics in Defence 2011, Shrivenham, UK, 20 October 2011.

[15] C. Piciarelli, C. Micheloni, G.L. Foresti, "Trajectory-based anomalous event detection", IEEE Trans. on Circuits and Systems for Video Technology 18(11):1544-1554, November 2008.

[16] R. Laxhammar, G. Falkman, "Sequential Conformal Anomaly Detection in Trajectories based on Hausdorff Distance", 12th International Conference on Information Fusion, Chicago, IL, USA, 5-8 July 2011.

[17] A. Malik, R. Maciejewski, B. Maule, D. S. Ebert, "A visual analytics process for maritime resource allocation and risk assessment", IEEE Conf. on Visual Analytics Science and Technology (VAST), Providence, RI, USA, October 2011

[18] M. Perše, M. Kristan, S. Kovačič, G. Vučkovič, J. Perš, "A trajectory-based analysis of coordinated team activity in a basketball game", Computer Vision and Image Understanding, 113(5):612-621, May 2009.

[19] P. Kumpulainen, K. Hätönen, "Compression of cyclic time series data", 12th IMEKO TC1 & TC7 Joint Symposium on "Man, Science and Measurement," Annecy, France, 3-5 September 2008