Airborne Monitoring of Ground Traffic Behaviour for Hidden Threat Assessment

Seungkeun Kim, Rafal W. Żbikowski, Antonios Tsourdos, and Brian A. White
Department of Informatics and Sensors
Cranfield University, Defence Academy of the UK
Swindon, UK
s.kim; r.w.zbikowski; a.tsourdos; b.a.white@cranfield.ac.uk

Abstract – *This paper focuses on development of behaviour recognition technique for airborne monitoring of ground traffic to detect hidden threats. To enhance tracking accuracy, sensor fusion and smoothing are applied with Kalman filter. To tackle behaviour recognition, trajectory approximation and classification methodology is proposed using differential geometric quantities and string matching. Simulation on a ground vehicle is done to verify the feasibility of the proposed algorithms.*

Keywords: Sensor fusion, tracking filter, trajectory classification, behaviour recognition, string matching.

1 Introduction

The focus of this study is airborne monitoring of ground traffic behaviour in order to detect disguised threats and then to notify the human commander about the potentially dangerous vehicles. Within this traffic the most difficult challenge is to recognise behaviours of the potentially dangerous vehicles, disguised as legitimate traffic. Most of these activities of the dangerous vehicles are characterised by occasional deviations from motion characteristics of the legitimate traffic. As an initiative, we intend to draw upon experience of airborne law enforcement, especially air support tasking for vehicle pursuit and air search at scenes of crime. Fot this, the developed techniques should be able to provide continuity of tracking of the vehicles of interest and thus enable positive identification of suspects. Since typical ground traffic in an urban environment is quite dense and involves numerous vehicles, achieving the proposed capabilities by a single sensor platform is unlikely to be feasible. Moreover, long endurance surveillance is required, with good spatial coverage, which would be difficult to accomplish with manned aircraft. Therefore we propose to investigate a swarm of airborne sensor platforms endowed with an appropriate level of autonomous decision making to support the human commander. It requires the sensor swarm to maximise information by sensor fusion and to manage mobility and reconfiguration in case of failure of individual sensor platforms. The swarm should also decide its collective and individual actions such as path planning and control using different pieces of information. In this paper, we deal with tracking filter design for ground vehicles, sensor fusion, trajectory approximation, behaviour classification and recognition based on differential geometric quantities and symbolic dynamics. The position estimation accuracy of the ground vehicle is enhanced by the sensor fusion based on covariance intersection and optimal smoothing. Analytical curvature and acceleration histories of the ground vehicle are obtained through trajectory approximation using a polynomial function. Based on this differential geometrical quantities, trajectory classification methodology is proposed. Lastly, simulation on a ground traffic vehicle is done to verify the feasibility of the proposed algorithms using an commercial off-the-shelf traffic simulation program.

2 Tracking filter and sensor fusion

2.1 Target modelling

In this study, acceleration dynamics [1] is applied to tracking of the ground traffic vehicle. Discretised system and measurement equations are normally:

$$X(k+1) \quad = \quad F(k)X(k) + \eta(k), \tag{1}$$

where $X = [x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}]$. The state transition matrix $F(k)$ can be represented as follows [1]:

$$F(k) = \begin{bmatrix} F_1 & \underline{0} \\ \underline{0} & F_1 \end{bmatrix}, F_1 = \begin{bmatrix} 1 & T & \phi \\ 0 & 1 & (1-e^{-\alpha T})/\alpha \\ 0 & 0 & e^{-\alpha T} \end{bmatrix}, \tag{2}$$

where $\phi = (e^{-\alpha T} + \alpha T - 1)/\alpha^2$, a sampling time $T = 0.5$, and a correlation parameter $\alpha = 0.6$. The covariance matrix of the process noise $\eta(k)$ can be modeled as follows:

$$Q(k) = 2\alpha\sigma_m^2 \begin{bmatrix} Q_1 & \underline{0} \\ \underline{0} & Q_1 \end{bmatrix}, Q_1 = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix} \tag{3}$$

where σ_m is the standard deviation related to noise of target acceleration, and a definition of q_{ij} is skipped due to space limit (refer to [1]).

2.2 Sensor modelling

The measurement equation can be represented as:

$$Z(k+1) = H(k)X(k+1) + v(k+1) \quad (4)$$

where $H(k) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$. The measurements in the Cartesian coordinates are generated from the relation with measurements in the spherical coordinates:

$$M_x = R\cos\varphi\cos\theta, M_y = R\cos\varphi\sin\theta. \quad (5)$$

The measurement covariance matrix is computed by considering transformation from the spherical to Cartesian coordinates as follows:

$$R(k) = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}, \quad (6)$$

where a definition of r_{ij} is skipped due to space limit (refer to [1]).

2.3 Discrete Kalman tracker

The initial state estimates can be obtained using the kinematics of the first three measurements. When αT is small, the covariance matrix can be initialized in the simplified form:

$$P(0) = \begin{bmatrix} P_1 & \underline{0} \\ \underline{0} & P_1 \end{bmatrix}, P_1 = \frac{\sigma_x^2}{T^4}\begin{bmatrix} T^4 & T^3 & T^2 \\ T^3 & 2T^2 & 3T \\ T^2 & 3T & 6 \end{bmatrix}. \quad (7)$$

For reference, we now summarise the algorithm of a general discrete Kalman filter [2]. Firstly, extrapolation of the states and error covariance matrix should be:

$$\begin{aligned} \hat{X}(k)^- &= F(k)\hat{X}(k-1)^+ \\ \hat{P}(k)^- &= F(k)\hat{P}(k-1)^+F(k)^T + Q(k). \end{aligned} \quad (8)$$

Then correction of the states and error covariance matrix is:

$$\begin{aligned} \hat{X}(k)^+ &= \hat{X}(k)^- + K(k)[Z(k) - H(k)\hat{X}(k)^-] \\ \hat{P}(k)^+ &= [I - K(k)H(k)]\hat{P}(k)^-, \end{aligned} \quad (9)$$

where $K(k) = P(k)^-H(k)^T[H(k)P(k)^-H(k)^T + R(k)]^{-1}$ is the Kalman gain controlling weight between extrapolation and measurement correction.

2.4 Fusion using Covariance Intersection

The focus in this section is sensor fusion using a Covariance Intersection algorithm. Here the concept of the sensor fusion and the Covariance Intersection will be briefly introduced based on the reference [3]. Data fusion is a well-known algorithm in which any two pieces of data are fused together in order to get a new estimate of the underlying information. Since the input sources are always corrupted by noise, it can be represented as random variables a and b, respectively. The real statistics of these variables a and b are assumed to be unknown; they, in turn, provide the only information of consistent estimates of the means and the covariances of these variables. In order to define the consistency of these two variables it is assumed that the means of these variables are $\bar{a}$ and $\bar{b}$. So the deviations for these assumed means are, $\tilde{a} \triangleq a - \bar{a}$ and $\tilde{b} \triangleq b - \bar{b}$. The information from these variables a and b are combined together in order to get a new estimate $\{\bar{c}, P_{cc}\}$. The Kalman filter uses a linear update rule of the form of $\bar{c} = W_a\bar{a} + W_b\bar{b}$ in order to calculate the covariance as:

$$P_{cc} = W_a(P_{aa}W_a^T + P_{ab}W_b^T) + W_b(P_{ba}W_a^T + P_{bb}W_b^T). \quad (10)$$

The trace of P_{cc} is minimised by W_a and W_b. However, this calculation is done using the assumed covariance, but the actual covariance is:

$$\bar{P}_{cc} = W_a(\bar{P}_{aa}W_a^T + \bar{P}_{ab}W_b^T) + W_b(\bar{P}_{ba}W_a^T + \bar{P}_{bb}W_b^T). \quad (11)$$

If the assumed and the actual variables are uncorrelated ($P_{ab} = \bar{P}_{ab} = 0$) the consistency of a and b allows for the consistent update. However, if $\bar{P}_{ab} \neq 0$ then it is hard to generate a consistent update. This problem led to the development of the Covariance Intersection method [4] which is a data fusion algorithm which has a convex combination of the means and the covariance in the information space. The basic intuition behind the covariance intersection is to form a geometric interpretation of the above equation. When the P_{cc} is within the intersection between P_{aa} and P_{bb} for any possible choices of P_{ab}, then an update strategy in the intersection region P_{cc} must be consistent. The tighter the updated covariance fits the region of intersection, the more will be the information used. This intersection is characterised by the convex combination of the covariances and this CI algorithm is formulated as:

$$P_{cc}^{-1} = \omega P_{aa}^{-1} + (1-\omega)P_{bb}^{-1} \quad (12)$$

$$P_{cc}^{-1}\bar{c} = \omega P_{aa}^{-1}\bar{a} + (1-\omega)P_{bb}^{-1}\bar{b} \quad (13)$$

where $\omega \in [0, 1]$. Free parameter ω manipulates the convex weights which are assigned to a and b. Different choices of ω can be used to optimise the covariance update with respect to different performance criteria, e.g. minimising the trace or the determinant of P_{cc}. The cost functions are convex with respect to ω and will have a unique optimum in the range of $0 \leq \omega \leq 1$. This CI algorithm is detailed in the references [4] and [5].

3 Trajectory approximation

3.1 Optimal smoother

Because behaviour description for the ground vehicle requires its trajectory history for a specific length of time, we can use the past state estimates to enhance the tracking accuracy. In this case, an optimal fixed-interval smoother [2] can be applied with the Kalman tracker. Generally the optimal smoothing algorithm is composed of a forward filter and a backward filter. The basic idea of the optimal smoothing is that if the measurements between t and $T_f(> t)$ are available, the estimates of the forward filter at time t, $\hat{x}_f(t)$,

can be adjusted based on the estimates of the backward filter at that time, $\hat{x}_b(t)$. The detail derivation of the optimal smoother can be found in [2]. The forward filter generally adopt the Kalman filter already described in the previous section. In case of the backward filter, let us assume the following variable transformation.

$$S_k(+) \triangleq P_k^b(+)^{-1},\ \hat{y}_k(+) \triangleq S_k(+)\hat{x}_k^b(+) \quad (14)$$
$$S_k(-) \triangleq P_k^b(-)^{-1},\ \hat{y}_k(-) \triangleq S_k(-)\hat{x}_k^b(-) \quad (15)$$

If the last discrete sampling time is N, the transformed estimates and error covariance matrix can be initialized as $\hat{y}_N(-) = 0$ and $S_N(-) = 0$. The backward filter performs the measurement update first contrary to the forward filter.

$$\hat{y}_k(+) = \hat{y}_k(-) + H_k^T R^{-1} z_k \quad (16)$$
$$S_k(+) = S_k(-) + H_k^T R^{-1} H_k \quad (17)$$

Then, the time update is done as the following equations.

$$K_k^b = S_k(+)(S_k(+) + Q^{-1})^{-1} \quad (18)$$
$$S_{k-1}(-) = F_k^T(I - K_k^b)S_k(+)F_k \quad (19)$$
$$\hat{y}_{k-1}(-) = F_k^T(I - K_k^b)\hat{y}_k(+) \quad (20)$$

Finally, the Kalman gain, the error covariance, and the state estimates can be newly defined as follow by the optimal smoothing algorithm.

$$K_k = P_k^f(+)S_k(-)(I + P_k^f(+)S_k(-))^{-1} \quad (21)$$
$$P_k = (I - K_k)P_k^f(+) \quad (22)$$
$$\hat{x}_k = (I - K_k)\hat{x}_k^f(+) + P_k\hat{y}_k(-) \quad (23)$$

where the subscript f means the forward filter. As a result, the estimates of the forward Kalman filter is corrected by the weighted sum with the estimates of the backward filter.

3.2 Cubic polynomial approximation

Curvature of the ground vehicle trajectory that will be used for behaviour classification can be obtained using the estimated states from the tracker:

$$k = \frac{\dot{r} \times \ddot{r}}{\|\dot{r}\|^3} = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \quad (24)$$

where $\dot{r} = [\dot{x}, \dot{y}]$, and $\ddot{r} = [\ddot{x}, \ddot{y}]$ are vectors composed of the states from $\hat{x}$. The curvature history computed numerically at every sampling time(0.5 seconds) is not good enough to be used for the behaviour description and classification because of its frequent peak phenomenons by numerical error and coarse sampling time. To tackle this, we propose a moving-window based trajectory approximation using a polynomial function versus time. This kind of technique has been recently used for approximating the position history acquired by visual sensor system in machine vision area. Assuming r_k indicates a two-dimensional position vector, let a moving-window trajectory be the N-sampling position vector set.

$$p_i = r_i = (\hat{x}_i, \hat{y}_i)^T, k - N + 1 \le i \le k \quad (25)$$

The x components $\hat{x}_i$ can be approximated by a third order polynomial function $f_x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$. In Ref. [6], the second order polynomial function was used for approximating the car trajectory acquired by a camera in an urban parking space where the car could not have a complicated shape over the size of their moving-window(1.6 seconds). In our case, the second order approximation is not enough since the ground vehicle can perform the lane change with the relatively high speed. Now the coefficients $a_i, 0 \le i \le 3$ are chosen such that $\min \sum_{i=k-N+1}^{k}(f_x(t_i) - \hat{x}_i)^2$. To minimize the above performance index in the least-square sense, we can get the following matrix equation:

$$\underline{\hat{x}} = T\underline{a} \quad (26)$$

where

$$\underline{\hat{x}} = (\hat{x}_{k-N+1}, \hat{x}_{k-N+2}, \ldots, \hat{x}_k)^T \quad (27)$$
$$\underline{a} = (a_0, a_1, a_2, a_3)^T \quad (28)$$
$$T = \begin{bmatrix} 1 & t_{k-N+1} & t_{k-N+1}^2 & t_{k-N+1}^3 \\ 1 & t_{k-N+2} & t_{k-N+2}^2 & t_{k-N+2}^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_k & t_k^2 & t_k^3 \end{bmatrix}. \quad (29)$$

The coefficients a_i of the cubic polynomial function $f_x(t)$ can be obtained by pseudo inversion of the above matrix equation: $\underline{a} = T^+\underline{\hat{x}}$.

The most advantage of this approximation is that we can get the analytic function of the position history. Even better, the analytic function of the velocity and acceleration histories can be easily derived using the position function as follows.

$$x(t) = f_x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (30)$$
$$\dot{x}(t) = \dot{f}_x(t) = a_1 + 2a_2 t + 3a_3 t^2 \quad (31)$$
$$\ddot{x}(t) = \ddot{f}_x(t) = 2a_2 + 6a_3 t \quad (32)$$

$f_y(t)$ and its coefficients b_i and the corresponding $y(t)$, $\dot{y}(t)$, and $\ddot{y}(t)$ can be also computed in a similar way. Substituting the above equations into Eq.(24) gives the analytical curvature history. Another advantage of this approach is that it is possible to virtually increase the sampling frequency.

4 Classification of driving behaviour

To recognize an drivers irregular behaviours, it is needed to classify the moving-window trajectory acquired in the previous chapter. The driving maneuver does not happen for a single sampling time, rather they happen for a specific length of time. For instance, let us suppose an aggressive driver repeat weaving, i.e. changing lanes with a short interval. To detect that, we have to store that vehicles trajectory classification history, i.e. driving mode, and then must perform decision-making on whether the vehicle does worrying behaviour and what type of behaviour that is. To conform these requirements, a decision-making flowchart

is considered as shown in Figure 1. The tracker, the optimal smoother, and the trajectory approximation were already done up to the previous section. Let us move on to the trajectory classification involving how to sort out driving behaviours. Fraile and Maybank [6] divided the driv-

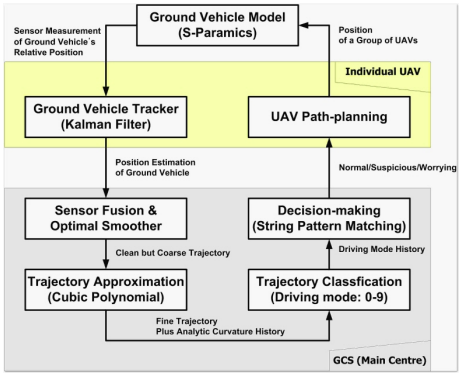

Figure 1: An overall flowchart for decision-making

ing mode of the ground vehicle into four categories: Ahead, Left/Right Turn, Stopping. Oliver and Pentland [7] categorized this as eight manoeuvres: Passing, Turning Right/Left, Changing Lanes Right/Left, Starting, and Stopping. These classifications are not suitable to our scope because focusing on the car manoneuvres in a urban parking space with slow speed. While the latter looks more detailed, they used various sensors equipped by a smart car, e.g. analog speed, acceleration, steering, and digital break/gear indicators. Those sensor signals are unavailable to the UAV flying in the air. Besides, their works [6, 7] are not detailed enough to depict the forward driving precisely. In this study, we propose a new driving-mode classification based on the trajectory approximation proposed in the previous section. In [6], the least speed and the greatest rate of orientation change of the vehicle was used to distinguish stopping and turning from driving ahead, respectively.

This approach is applied to the moving-window-based approximated trajectory in Eqs. (30)-(32). Let us assume a new time sequence, $0 < T_n < 2T_n < \ldots < (N-1)T_s, T_s = cT_n$ where T_s is a original sampling time of sensor data acquistion, T_n is a new virtual sampling time, and N is the number of samplings. In this study, it is assumed that $N = 4$, $T_s = 0.5$, and $c = 5$, and thus the new virtual sampling time is 0.1 seconds. The selection of N=4, i.e. 1.5 seconds' moving window reflects that the bandwidth for lane changing is at least 1.0Hz according to the reference [8]. Over the new time sequence, Eqs. (30)-(32) can yield the analytical velocity $(\dot{x}, \dot{y})$ and acceleration $(\ddot{x}, \ddot{y})$ histories. Using these, the least speed u of the car can be provided in the time inverval of the moving window as follows.

$$u = \min v(t) = \min \sqrt{\dot{x}^2(t) + \dot{y(t)}^2} \tag{33}$$

The rate of change of orientation can be obtained:

$$\theta = v(t)k(t) = \sqrt{\dot{x}^2(t) + \dot{y}^2(t)} \frac{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}{(\dot{x}^2(t) + \dot{y}^2(t))^{3/2}} \tag{34}$$

Using Eqs. (33)-(34), the thresholds were proposed for classifying a car trajectory in the parking space in the reference [6]. Although this threshold is not compatible with our application, it gives us a good starting point of our trajectory classification. With the background discussed so far, we propose the following nine driving modes based on the minimum speed u in Eq. (33), the rate of orientation change θ in Eq. (34), the acceleration in Eq. (32), and the curvature in Eq. (24).

Stopping (0) $u < 1$: Since 1 m/s equals to 3.6 km/h, we can say that the car does not move or is about to stop or start moving.

Left turn (1) $\max(\theta)\min(\theta) > 0$ and $\max(\theta) > 0.8$: In Ref.[6], they assumed only the maximum rate of orientation change, but we introduced inspection of its sign change to distinguish the pure turning maneuver from the lane changing.

Right turn (8) $\max(\theta)\min(\theta) > 0$ and $\max(\theta) < -0.8$

Left lane change (2) $\max(\theta)\min(\theta) < 0$, $\max(|\theta|) > 0.8$, and $\theta(0) > 0$: The only difference to the left turn of this condition is the sign change of the rate of orientation change. As you can see in Figure 2, the sign of curvature transits from positive to negative in case of the left lane change. The rate of change of orientation also shows the same tendendency.

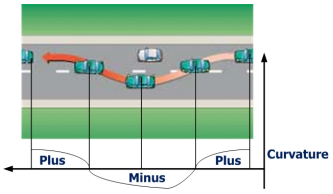

Figure 2: The sign change of curvature when changing lanes

Right lane change (7) $\max(\theta)\min(\theta) < 0$, $\max(|\theta|) > 0.8$, and $\theta(0) < 0$: The only difference to the right turn of this condition is the sign change of the rate of orientation change. As can be seen in Figure 2, the sign of curvature transits from negative to positive in case of the right lane change.

Closing gap (6) $\max(a_f)\min(a_f) < 0$, and $a_f(0) > 0$ where $a_f = \ddot{x}\cos\psi + \ddot{y}\sin\psi$ is the forward acceleration in direction of velocity, and $\psi = \tan^{-1}\dot{x}/\dot{y}$ is the heading angle from the North.: The acceleration profile can be used for classification of the forward driving mode. When the driver wants to close gap to the preceding vehicle, the sign of acceleration transit from positive to negative as shown in the left plot of Figure 3 even if the real profile is not necessarily constant.

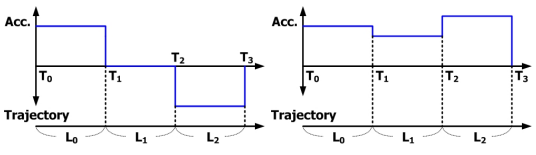

Figure 3: Acceleration history when closing gap (L) and accelerating ahead (R) to the preceding vehicle

Widening gap (3) $\max(a_f)\min(a_f) < 0$, and $a_f(0) \leq 0$: Contrary to the case of closing gap, the sign of acceleration transit from negative to positive when the driver wants to widen gap to the preceding vehicle.

Accelerating ahead (4) $\max(a_f)\min(a_f) > 0$, and $a_f(0) > 0$:The sign of acceleration keeps positive as shown in the left plot of Figure 3.

Decelerating ahead (5) $\max(a_f)\min(a_f) > 0$, and $a_f(0) \leq 0$: The sign of acceleration keeps negative contrary to the case of the accelerating ahead.

Figure 4 shows a full flowchart of the car trajectory classification proposed in this study.

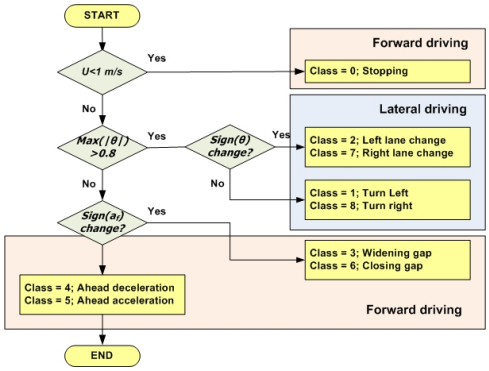

Figure 4: A flowchart of trajectory classification

5 Detection using string matching

5.1 Exact matching

The essence of the classification was to categorise characteristic manoeuvres associated with forward or lateral driving by assigning them to one of nine classes: $\{0, \ldots, 8\}$, as dealt with in the previous section. This allows not only to recognise characteristic fragments of the trajectories, but to enable recognition of ground traffic behaviour in a theoretically sound, intuitive, robust, computationally-efficient and flexible way. The key tools for this detection scheme are symbolic dynamics and string matching. The mathematical subject of symbolic dynamics originally arose in the theory of dynamical systems and was motivated by the qualitative approach to dynamics in which the character of trajectories is more important than their numerical values. String matching theory is a well-developed area of text processing. String matching consists in finding all the occurrences of a string (called a pattern) in a text where the pattern is a string x of length m, while the text is a string y of length n. A good example of string-matching practice with strong theoretical underpinnings is the Boyer-Moore algorithm which is considered as the most efficient string matching algorithm in usual applications. A simplified version of the Boyer-Moore scheme, or the entire algorithm, is often implemented in text editors for the search and replace commands. There are widespread and flexible software implementations of string matching algorithms. In this study, using the alphabet $\{0, \ldots, 8\}$, a symbolic time series y_i is generated for each $i = 1, \ldots, N$ using a N sampling time trajectory by target-tracking and classification. The definitions of normal, suspicious and worrying behaviours are expressed as strings x_n, x_s and x_w, made up of the letters of the nine-symbol alphabet.

5.2 Approximate matching

In the previous section, the task of concern is to find the string pattern strictly identical to one of the possible classes. However, to find the exact string matching to a certain behaviour class is not technically easy and computationally burdensome. Let us assume we are interested in a reference pattern (driving mode sequence) '145048', which means that a ground vehicle turns left, accelerates ahead, decelerates ahead, stops, accelerates ahead again, and lastly turns right. Sometimes, the test pattern appeared as the driving mode sequences, '145548' or '145448' cannot be ignored in the detection scheme, whose fourth element of the string might be one of the following forward driving modes: '3'; '4', '5', '6', not '0'. However, it is not efficient to run the string pattern matching repeatedly after regarding all the possible driving-mode sequences as the reference pattern. Typically this problem arises in speech recognition. When someone says 'beauty', the test pattern may be sensed as 'beety' or 'beaut' due to errors in the reading sensor or speaking speed difference of the speaker [9]. In this case, we need to define a cost measuring the distance or the similarity between the reference patterns and the test patterns, as which this study adopts a concept of Edit distance.

The Edit distance between two patterns is defined as the cost to convert one pattern to the other. If the patterns are of the same length, then the cost is directly related to the number of symbols that have to be changed in one of them so the the other pattern results. In case the two patterns are not of equal length, symbols have to be either deleted or inserted at certain places of the test string [9]. Although this problems arises in automatic editing and text retrieval applications, it is worth considering for detection of driving mode sequences similar but not exactly matching to the predefined interested sequences. The Edit distance [10, 9] between two string pattern S_1 and S_2 is defined as the minimum total number of changes C, insertions I, and deletions R required to change pattern S_1 into S_2:

$$D(S_1, S_2) = \min_j [C(j) + I(j) + R(j)] \qquad (35)$$

where j runs over all possible combinations of symbol variations in order to obtain S_2 from S_1 [9]. A dynamic programming methodology is employed to compute the required minimum j. For this, let us form the grid by placing the symbols of the reference pattern in the horizontal axis and the test pattern in the vertical axis as shown in Figure 5. For

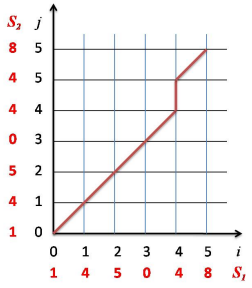

Figure 5: Two dimensional grid example for computing Edit distance

optimal path searching using dynamics programming, consider the following constraints [9].

- D(0,0)=0: The cost $D(0,0)$ is zero.
- A complete path is searched.
- Each node (i,j) can be reached only through three allowable predecessors: $(i-1,j),(i-1,j-1),(i,j-1)$.

The cost by the above three transitions can be defined for diagonal transitions and horizontal/vertical transitions, respectively.

$$d(i,j|i-1,j-1) = \begin{cases} 0 \text{ if } r(i)=t(j) \\ 1 \text{ if } r(i)\neq t(j) \end{cases} \quad (36)$$

$$d(i,j|i-1,j) = d(i,j|i,j-1) = 1 \quad (37)$$

Using these constraints and the Edit distance as the performance index, the dynamic programming can be applied.

The proposed detection scheme has good theoretical basis on symbolic dynamics, is intuitive, robust within threshold bounds, computationally efficient and flexible since string patterns to search for can be easily adapted to operational context.

6 Simulation result

6.1 Assumptions and conditions

The vehicle trajectory data (2Hz) in a S-Paramics traffic model of Devizes, Wiltshire, UK, were used to generate measurements composed of relative range, azimuth, and elevation angle with respect to a position of UAV as shown in Figure 6. The vehicle starts at the western side of Devizes and traverses a part of the town centre and then turns back, so that the journey ends at the northwestern side of Devizes. To verify the performance of the sensor fusion algorithm discussed so far, we carried out a simulation of cylindrical station-keeping of two UAVs flying with $90km/h$ at the different altitudes, $h_1 = 500m$ and $h_2 = 800m$, as

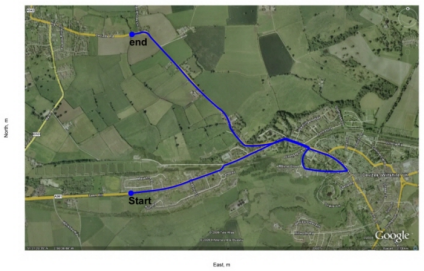

Figure 6: Vehicle trajectory within the Devizes road network with GIS satellite data overlaid thanks to Google Map

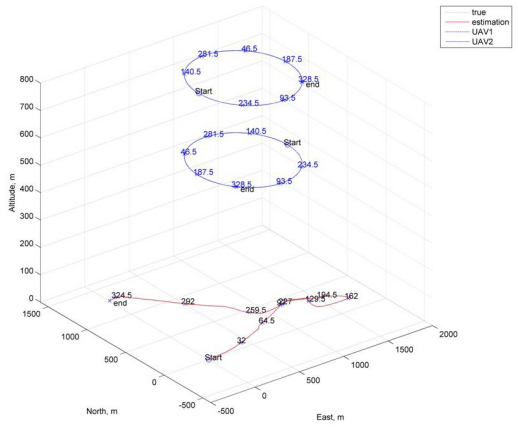

Figure 7: Car trajectory by covariance intersection

shown in Figure 7. The diameter of the cylinder is $1km$, and its centre is at $(1000m, 700m)$. In order effectively to assess the performance of the Covariance Intersection method, different noise characteristics is considered for each UAV: $\sigma_{r1} = 10m$, $\sigma_{\psi 1} = 5 \cdot 10^{-3}$ and $\sigma_{\phi 1} = 5 \cdot 10^{-3}$ for the UAV1 and $\sigma_{r2} = 10m$, $\sigma_{\psi 2} = 7 \cdot 10^{-3}$ and $\sigma_{\phi 2} = 3 \cdot 10^{-3}$ for the UAV2.

6.2 Tracking and sensor fusion

Using the estimations of the Kalman trackers, we then carry out the Covariance Intersection and the optimal smoothing. Figure 7 displays the results of ground vehicle tracking by Covariance Intersection. Also, Table 1 shows the tracking accuracy is enhanced by the Covariance Intersection and smoothing. Actually, the tracking error of smoothing is about 50% compared to the Kalman tracker. Figure 8 shows

	mean (m)	max (m)
Kalman tracker(UAV1)	(3.63,2.89)	(16.25,13.80)
Kalman tracker(UAV2)	(3.48,2.89)	(16.25,13.25)
Cov. Intersection	(3.11,2.35)	(13.57,10.89)
Smoothing with CI	(1.64,1.57)	(8.41,7.77)

Table 1: Analysis of estimation error (x_e, y_e)

how a new covariance ellipse (green one) is computed using the Covariance Intersection algorithm.

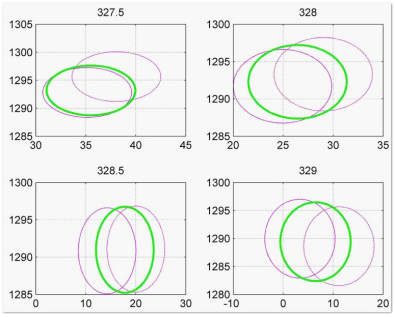

Figure 8: Covariance ellipses at the final four steps

6.3 Trajectory approximation/classification

Figure 9 shows a moving-window histories using the position estimates between 122 and 124 seconds. In this study, we assume the moving-window is composed of four sampling times($N = 4$, 1.5 seconds), and the sampling frequency virtually can be increased from 2Hz to 10Hz. In the upper figure, the mark o denotes the real position estimation, and the mark x does the virtual estimation computed through the approximated functions. The lower figure shows the corresponding analytical acceleration, curvature and rate of change. Then, the history of ground vehicles driving-

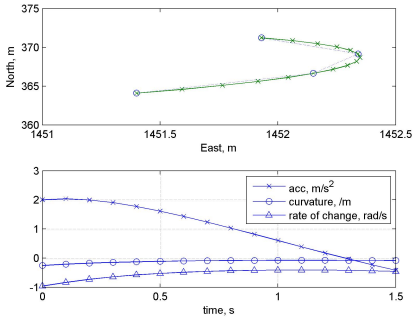

Figure 9: Trajectory approximation for four steps

mode is found using the classification algorithm proposed in Section 4 as shown in Figure 10. In addition, we performed

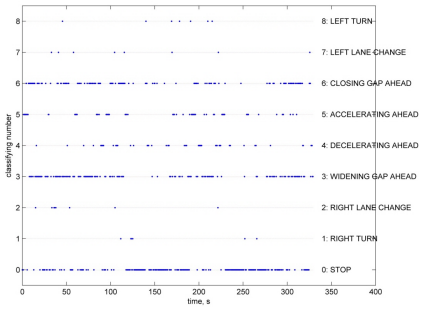

Figure 10: Driving-mode classification

a simulation to detect the suspicious behaviour based on the trajectory classification. Because it is impossible to interrupt a vehicle behaviour in the S-Paramics, an lane change for 15 seconds is added in the off-line as shown in the left of Figure 11. At 31.5 seconds, the vehicle carries out the left lane change, and comes back four seconds later to the original lane. Again at 39 seconds, it performs the right lane change, and comes back four seconds later. This manoeuvre is called as weaving classified as one of the most common dangerous behaviour. Based on the minimum speed and the maximum rate of orientation, the accurate behaviour classification is accomplished as shown in the right of Figure 11.

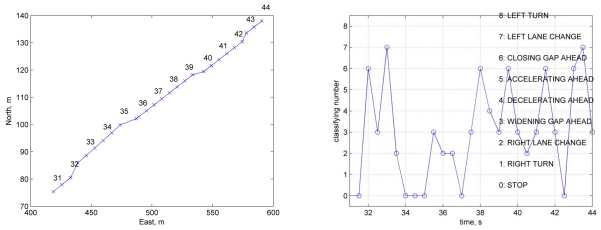

Figure 11: Trajectory(L) and classification(R) of weaving

6.4 Behaviour recognition

Finally, for string matching, the size of symbolic time series y_i is set as $N = 30$ equivalent to 15 seconds. To catch lane changing, the number of string matching with 2(right lane change) or 7(left lane change) is first searched as a decision for suspicious behaviour x_s, and then 27 or 72 for worrying behaviour x_w. The upper figure in Figure 12 shows the number of exact string matching with the abovementioned two behaviours. Based on this information, the weaving behaviour is declared as a worrying behaviour if the number of lane change is over three and the number of rapid lane change is not less than one.

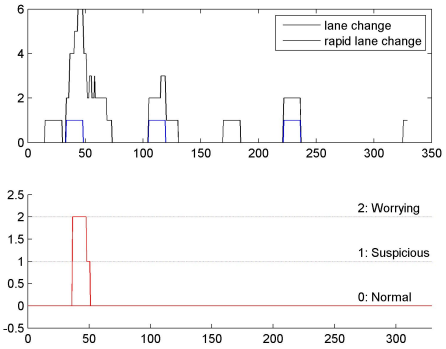

Figure 12: String matching and recognition of weaving

The approximate matching using the Edit distance is also applied to the same scenario as the previous case. The reference string pattern is assumed as '4742' which represents

that a ground vehicle shows accelerating forward, changing a lane right, accelerating forward again and then changing a lane left, successively. Figure 13 shows the result in which the exact matching with the reference pattern '4742' is applied. As can be seen in the figure, the exact matching

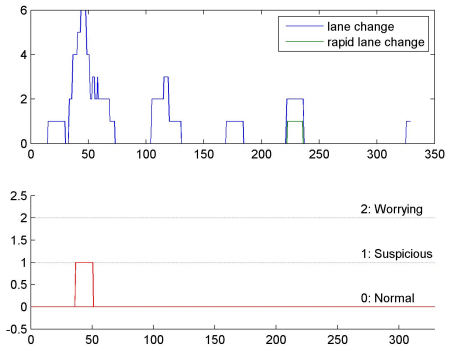

Figure 13: Behaviour recognition w/ exact matching

does not detect any during the time when abrupt lane changing happens near 50 seconds. While, for the approximate matching, the test string pattern is assumed as every five samples of string obtained from the trajectory classification. Because actually the behaviour pattern '4742' could appear as '47442', '43742, and so on, it is promising to compare the test pattern having a longer length than that of the reference pattern. In this simulation, the test pattern is regarded as the similar pattern to the reference pattern '4742' if its Edit distance from the reference pattern is not higher than 3. As can be seen in Fig. 14, the approximate matching robustly detects the similar test patterns during the time when abrupt lane changing happens near 50 seconds. Then, the

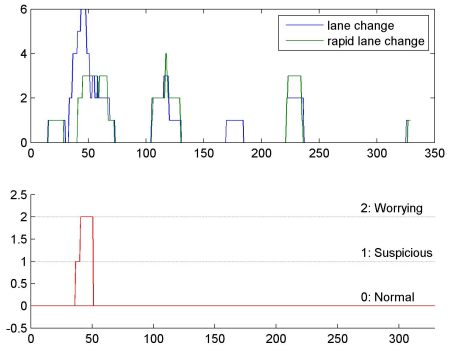

Figure 14: Behaviour recognition w/ approximate matching

decision-making algorithm should regard this vehicles future behaviours as suspicious or worrying. Simultaneously it should notify a human operator about this danger, and the UAV has to carry out a new path planning to track this vehicle. Although this simulation seems simple, it gives us the feasibility of the proposed approaches including tracking filter with sensor fusion, trajectory approximation and overall trajectory classification and recognition algorithm.

7 Conclusions

This paper presented a novel approach on sensor fusion and information recognition for airborne monitoring of ground traffic behaviours. This study is still going on under Competitions of Ideas held by UK Ministry of Defence, and currently decision making on path planning and control of the UAV swarm has been developed. As a future work, detection scheme proposed in this paper should be integrated with the decision-making on UAV path planning/control and could be applied to various anomalous behaviour scenarios in view of both military and civil applications.

References

[1] K. Mehrotra and P. R. Mahapatra. A jerk model for tracking highly maneuvering targets. *IEEE Transactions on Aerospace and Electronic Systems*, 33(4):1094–1105, 1997.

[2] F. L. Lewis. *Applied optimal control and estimation*. Prentice Hall Englewood Cliffs, NJ, 1992.

[3] S. B. Lazarus. *Navigation and mapping using multiple autonomous vehicles*. PhD thesis, Cranfield University, 2009.

[4] S. J. Julier and J. K. Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the American Control Conference*, volume 4, pages 2369–2373. Citeseer, 1997.

[5] P. O. Arambel, C. Rago, and R. K. Mehra. Covariance intersection algorithm for distributed spacecraft state estimation. In *American Control Conference, 2001. Proceedings of the 2001*, volume 6, 2001.

[6] R. Fraile and S. J. Maybank. Vehicle trajectory approximation and classification. In *British Machine Vision Conference*, volume 698, page 702. Citeseer, 1998.

[7] N. Oliver and A. P. Pentland. Graphical models for driver behavior recognition in a smartcar. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 7–12, 2000.

[8] C. F. Lin and A. G. Ulsoy. Calculation of the time to lane crossing and analysis of its frequency distribution. In *American Control Conference, 1995. Proceedings of the*, volume 5, 1995.

[9] S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Academic Press, San diego, USA, 2006.

[10] E. J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.