Evaluation of Object Tracking for Aircraft Activity Surveillance

David Thirde[1], Mark Borg[1] Josep Aguilera[2]
James Ferryman[1], Keith Baker[1] Martin Kampel[2]

[1] Computational Vision Group, The University of Reading, UK
{D.J.Thirde, M.Borg, J.Ferryman,K.Baker}@rdg.ac.uk http://www.cvg.rdg.ac.uk/

[2] Pattern Recognition and Image Processing Group, Vienna University of Technology, Austria
{agu,kampel}@prip.tuwien.ac.at http://prip.tuwien.ac.at/

Abstract

This paper presents the evaluation of an object tracking system that has been developed in the context of aircraft activity monitoring. The overall tracking system comprises three main modules — Motion Detection, Object Tracking and Data Fusion. In this paper we primarily focus on performance evaluation of the object tracking module, with emphasis given to the general 2D tracking performance and the 3D object localisation.

1. Introduction

This paper describes work undertaken on the EU project AVITRACK. The main aim of this project is to automate the supervision of commercial aircraft servicing operations on the ground at airports (in bounded areas known as *aprons*). A combination of visual surveillance algorithms are applied in a multi-camera environment to track objects and recognise activities predefined by a set of servicing operations. More details of the system are given in [9]. Each camera agent performs real-time detection and tracking of scene objects, the output is transmitted to a central server where data association and fused object tracking is performed.

The tracking of moving objects on the apron has previously been performed using a top-down model based approach [8] although such methods are generally computationally expensive. On a standard workstation (2×3Ghz pentium-4 processors with 2Gb RAM running Suse Linux 9.1) we have found the model based method to fit one textured wireframe model in 0.25 seconds. In the apron environment there are 28 object categories, which would therefore result in a prohibitive frame-rate when tracking multiple objects. An alternative approach, bottom-up scene tracking, is a process that comprises the sub-processes *motion detection* and *object tracking*; the advantage of bottom-up tracking is that it is more generic and computationally efficient compared to the top-down method.

Motion detection is normally the first process that is performed in a tracking system, and it attempts to locate connected regions of pixels that represent the moving (foreground) objects within the scene that are of interest to the application. Several different methods can be used to achieve this: frame to frame differencing, background subtraction and motion analysis (optical flow based) techniques. Background subtraction methods (like [10, 5]) represent the static scene by a background model, normally learnt over a period of observation; motion detection is then performed by finding regions that do not match this background model. Because of environmental and lighting changes, especially in outdoor environments like AVITRACK, background subtraction methods must update the background model over time.

The output of the motion detection process is normally passed to object tracking algorithms that use trajectory or appearance based analysis to predict, associate and update previously observed objects in the current time step. One such method is the Kanade-Lucas-Tomasi (KLT) feature tracker [7], which combines a local feature selection criterion with feature-based matching in adjacent frames; this method has the advantage that objects can be tracked through partial occlusion when only a sub-set of the features are visible.

The problems that tracking algorithms have to deal with include motion detection errors and complex object interactions; e.g. objects appear to merge, occlude each other, fragment, undergo non-rigid motion, etc. Apron analysis presents further challenges due to the size of the vehicles tracked (e.g. the aircraft size is $34 \times 38 \times 12$ metres), therefore prolonged occlusions occur frequently throughout apron operations. The apron can also be congested with

objects (most of the activity occurs near the aircraft), so enhancing the difficulty of associating objects with regions.

The goal of this paper is to evaluate the performance of a KLT-based tracker that addresses the challenges of aircraft activity modelling. The choice of the KLT algorithm and other tracking algorithms implemented for AVITRACK are described in [9]. To improve the computational efficiency of the KLT tracker, motion segmentation is not performed globally to detect the objects. Instead, the KLT features are used in conjunction with a rule based approach to provide the correspondence between connected foreground regions; in this way the KLT algorithm *simultaneously* solves the problems of data association and tracking without presumption of a global motion for each object. Spatio-temporal reasoning is applied during object interactions (e.g. partial occlusions) to allow object prediction / matching and feature continuity during these complex events; this reasoning primarily takes the form of spatial and motion based analysis of the (merged) connected foreground region features to allow objects to be successfully predicted and matched.

The goal of performance evaluation is to characterise a system using an unbiased and clearly defined set of metrics. A set of metrics for evaluating object tracking were proposed by Black *et al* [2], these metrics are formed from a selection of previous work on the performance evaluation of tracking systems. These metrics, like many conventional approaches to performance evaluation, require the definition of ground truth tracking data against which the performance of the system can be quantitatively evaluated; there are a selection of open source tools that allow efficient manual generation of ground truth data, for example the ViPER [4] framework.

The remainder of this paper is organised as follows: Section 2 reviews the per camera motion detection. Section 3 introduces the per camera (2D) object tracking. Section 4 describes how objects are localised in the 3D world and how a confidence measure is derived for the measurement and Section 5 contains evaluation of the object tracking and localisation procedures.

2. Motion Detection

For the AVITRACK project, 16 motion detection algorithms were implemented and evaluated on various apron sequences under different environmental conditions (sunny conditions, fog, etc.). More detail about the algorithms, the evaluation process and the selection criteria can be found in [1, 9]. The algorithm selected for AVITRACK is the colour mean and variance [10], which represents the background model by a pixel-wise Gaussian distribution $N(\mu, \sigma^2)$ over the normalised RGB space. In addition, a shadow/highlight detection component based on the work of Horprasert *et al* [5], handles illumination variability.

3. Object Tracking

Real-time object tracking can be described as a correspondence problem, and involves finding which object in a video frame relates to which object in the next frame. Normally, the time interval between two successive frames is small, therefore inter-frame changes are limited, thus allowing the use of temporal constraints and/or object features to simplify the correspondence problem.

The KLT algorithm considers features to be independent entities and tracks each of them individually. Therefore, it is incorporated into a higher-level tracking process that groups features into objects, maintain associations between them, and uses the individual feature tracking results to track objects, taking into account complex object interactions. For each object O, a set of sparse features S is maintained. The number of features per object (i.e. $|S|$) is determined dynamically from the object's size and a configurable feature density parameter $\boldsymbol{\rho}$. If $\boldsymbol{\rho} = 1.0$, $|S|$ is the maximal number of features that can spatially cover object O, without overlap between the local feature windows.

The KLT tracker takes as input the set of observations $\{M_j\}$ identified by the motion detector. Here, an observation M_j is a connected component of foreground pixels, with the addition of a nearest neighbour spatial filter of clustering radius r_c, i.e., connected components with gaps $\leq r_c$ are considered as one observation. Given such a set of observations $\left\{M_j^t\right\}$ at time t, and the set of tracked objects $\left\{O_i^{t-1}\right\}$ at $t-1$, the tracking process is summarised as:

1. Generate object predictions $\{P_i^t\}$ for time t from the set of known objects $\left\{O_i^{t-1}\right\}$ at $t-1$, with the set of features $S_{P_i^t}$ set to $S_{O_i^{t-1}}$.
2. Run the KLT algorithm to individually track each local feature belonging to $S_{P_i^t}$ of each prediction.
3. Given a set of observations $\left\{M_j^t\right\}$ detected by the motion detector, match predictions $\{P_i^t\}$ to observations by determining to which observation M_j^t the tracked local features of P_i^t belong to.
4. Any remaining unmatched predictions in $\{P_i^t\}$ are marked as missing observations. Any remaining unmatched observations in $\left\{M_j^t\right\}$ are considered to be potential new objects.
5. Detect any matched predictions that have become temporarily stationary. These are integrated into the background model of the motion detector as a new background layer.
6. Update the state of those predictions in $\{P_i^t\}$ that were matched to observations and replace any lost features. The final result is a set of tracked objects $\{O_i^t\}$ at time t. Let $t = t + 1$ and repeat step 1.

In step 3 above, features are used in matching predictions to their corresponding observations in 2 ways — using the spatial information and the motion information of the features. Spatial rule-based reasoning is applied to detect the presence of merging or splitting foreground regions; in the case of merged objects the motion of the individual features are robustly fitted to (predetermined) motion models to estimate the membership of features to objects. If the motion models are not distinct or unreliable then the local states of the features are used to update the global states of the merged objects. The spatial rule-based reasoning is described in more detail in Section 3.1, while the motion-based segmentation method is described in Section 3.2. Section 3.3 describes the technique in step 5 above, for detecting and handling moving objects that become temporarily stationary.

3.1. Using Spatial Information of Features

This method is based on the idea that if a feature belongs to object O_i at time $t-1$, then the feature should remain spatially within the foreground region of O_i at time t. A match function is defined which returns the number of tracked features w of prediction P_i^t that reside in the foreground region of observation M_j^t:

$$f\left(P_i^t, M_j^t\right) = \left|\left\{w : w \in S_{P_i^t}, w \in M_j^t\right\}\right| \qquad (1)$$

In the case of an isolated (non-interacting) object, (1) should return a non-zero value for only one prediction-observation pair; ideally $f\left(P_i^t, M_j^t\right) = \left|S_{P_i^t}\right|$ – this is normally less due to lost and incorrectly-tracked features. For interacting objects, such as objects merging, occluding each other, undergoing splitting events, etc., a table of score values returned by (1) is constructed, and a rule-based approach is adopted to match predictions to observations.

The first rule handles the ideal matches of isolated objects, i.e. one-to-one matches between predictions and observations:

$$\begin{array}{l} f\left(P_i^t, M_j^t\right) > 0 \quad \text{and} \\ f\left(P_k^t, M_j^t\right) = 0, \quad f\left(P_i^t, M_l^t\right) = 0 \quad \forall k \neq i, l \neq j \end{array} \qquad (2)$$

The second rule handles the case when an object at time $t-1$ splits into several objects when seen at time t. This occurs when several observation regions match with a single prediction P_i^t - in other words, the set of observations is partitioned into two subsets: the subset $M1$ of observations that match only with P_i^t and the subset of those that do not match with P_i^t:

$$\begin{array}{l} f\left(P_i^t, M_j^t\right) > 0 \quad M_j^t \in M1 \subseteq M,\ |M1| > 1 \quad \text{and} \\ f\left(P_k^t, M_j^t\right) = 0, \ \forall M_j^t \in M1, k \neq i \quad \text{and} \\ f\left(P_i^t, M_l^t\right) = 0, \ \forall M_l^t \notin M1 \end{array} \qquad (3)$$

The prediction is then split into new objects, one for each of the matched observations in $M1$. The features of the original prediction P_i are assigned to the corresponding new object depending on whether they reside within its observation region or not. In this way, features are maintained throughout an object splitting event.

The third matching rule handles merging objects. This occurs when more than one prediction matches with an observation region:

$$\begin{array}{l} f\left(P_i^t, M_j^t\right) > 0 \quad P_i^t \in P1 \subseteq P,\ |P1| > 1 \quad \text{and} \\ f\left(P_i^t, M_k^t\right) = 0, \ \forall P_i^t \in P1, k \neq j \quad \text{and} \\ f\left(P_l^t, M_j^t\right) = 0, \ \forall P_l^t \notin P1 \end{array} \qquad (4)$$

In this case the state of the predictions (such as position and bounding box) cannot be obtained by a straightforward update from the observation's state, since only one combined (merged) observation is available from the motion detector. Instead, the known local states of the tracked features are used to update the global states of the predictions. The prediction's new centre is estimated by taking the average relative motion of its local features from the previous frame at time $t-1$ to the current one. This is based on the assumption that the average relative motion of the features is approximately equal to the object's global motion - this may not always be true for non-rigid objects undergoing large motion, and may also be affected by the aperture problem due to the small size of the feature windows. The sizes of the bounding boxes of the predictions are also updated in order to maximise the coverage of the observation region by the combined predictions' bounding boxes. This handles cases where objects are moving towards the camera while in a merged state and hence their sizes increase. If not done, the result is parts of the observation region that are not explained by any of the predictions.

3.2. Using Motion Information of Features

The motion information obtained from tracking the local features of a prediction P_i is also used in the matching process of step 3 above. Features belonging to an object should follow approximately the same motion (assuming rigid object motion). Motion models are fitted to each group of k neighbouring features of P_i. These motion models are then represented as points in a motion parameter space and clustering is performed in this space to find the most significant motion(s) of the object [11]. A weighted list is maintained per object of these significant motions and the list is updated over time to reflect changes in the object's motion - if a motion model gains confidence its weight is increased; if a new motion model is detected, it is added to the list, or replaces an existing lower probable one. The motion models are used to differentiate the features of merged objects by checking whether a feature belongs to one motion model or the other. This allows tracking through merging/occlusion

and the replenishment of lost features. The motion models of an object are also used to identify object splitting events – if a secondary motion becomes significant enough and is present for a long time, splitting occurs. Although the underlying assumption is of rigid object motion, the use of a weighted list of motion models should allow for the identification of the different motions for articulated vehicles; future work will address this issue.

Two types of motion models have been used for AVITRACK – affine and translational models. The affine motion model is generated by solving for:

$$w_t^T F w_{t-N} = 0 \tag{5}$$

where w_t and w_{t-N} are the locations of feature w at time t, $t-N$, and F is the fundamental matrix representing the motion. For the affine case, F has the form:

$$F = \begin{bmatrix} 0 & 0 & f_{13} \\ 0 & 0 & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \tag{6}$$

F is obtained through a minimisation process based on eigen analysis, as described in [11]. The affine motion model is then represented in terms of 5 motion parameters: $v_{affine} = \langle \boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\rho}, \boldsymbol{\lambda}, \boldsymbol{\theta} \rangle$, where:

$$\boldsymbol{\alpha} = arctan(\frac{-f_{13}}{f_{23}}) \tag{7}$$

$$\boldsymbol{\gamma} = arctan(\frac{f_{31}}{-f_{32}}) \tag{8}$$

$$\boldsymbol{\rho} = \sqrt{\frac{f_{31}^2 + f_{32}^2}{f_{13}^2 + f_{23}^2}} \tag{9}$$

$$\boldsymbol{\lambda} = \frac{f_{33}}{\sqrt{f_{13}^2 + f_{23}^2}} \tag{10}$$

$$\boldsymbol{\theta} = \boldsymbol{\alpha} - \boldsymbol{\gamma} \tag{11}$$

Clustering is performed in the motion parameter space to get the list of most significant motion models for the object.

The second motion model is simply the translational motion in the image plane:

$$v_{translational} = w_t - w_{t-N} \tag{12}$$

When tested on AVITRACK sequences, it was found that perspective and lens distortion effects cause the affine motion models to become highly dispersed in the motion parameter space and clustering performs poorly. The translational model, as can be expected, also suffers from these problems and affine motion effects, but the effect on clustering is less severe. This motion 'fragmentation' for the translational model is mitigated somehow by the use of the weighted list of motion models for each object. At present, the translational model is performing better than the affine model; future work will look into improving the affine model and using perspective motion models.

3.3. Stationary Objects

For the apron environment, activity tends to happen in congested areas near the aircraft with several vehicles arriving and stopping for short periods of time in the vicinity of the aircraft, creating occlusions and object merging problems. To allow objects to be differentiated and the tracking of moving objects in front of stopped objects, the motion detection process described in Section 2 was extended to include a multiple background layer technique. The tracker identifies stopped objects by one of two methods: by analysing an object's regions for connected components of foreground pixels which have been labelled as 'motion' for a certain time window; or by checking the individual motion of local features of an object. The accuracy of the second method depends on the sparseness of the features, and hence on the density parameter $\boldsymbol{\rho}$ introduced in Section 3. Stationary objects are integrated into the motion detector's background model as different background layers.

This technique is similar in idea to the Temporal Layers method described by Collins *et al* [3], except that their method works on a pixelwise level, using intensity transition profiles of pixels to classify them as 'stationary' or 'transient'. This is then combined with pixel clustering to form moving or stationary regions. This method performed poorly when applied to AVITRACK sequences, due mainly to stationary objects becoming fragmented into many layers as the duration objects remain stationary increases. This results in different update rates to the layers and incorrect re-activation once an object starts moving again. In the case of AVITRACK, the aircraft can remain stationary for up to half an hour - it is imperative that the object remains consistent throughout this time, its background layer gets updated uniformly and it is re-activated as a whole. The method adopted for AVITRACK works at the region-level and is handled by the tracker rather than at the motion detection phase, where the motion information of the local features can provide robust information on an object's motion. This use of region-level analysis helps to reduce the creation of a large number of background layers caused by noise.

The criterion used for checking stationarity was modified to take into account cases where as an object comes to rest, a sub-part of it remains in motion (e.g. a person emerging from a vehicle while it is slowing down to a stop). But the relaxation of this criterion, and the use of background layers in general, can result in ghosts (false positives) being detected when part of the background is uncovered. A method based on the movement density, i.e. the average change in a region, is used to detect such ghosts. Figure 1 illustrates the use of a multi-layered background model to distinguish overlapping objects. The matching of predictions to observations described in Sections 3.1 and 3.2 then takes into account the interaction that occurs between objects that become temporarily stationary and moving objects.

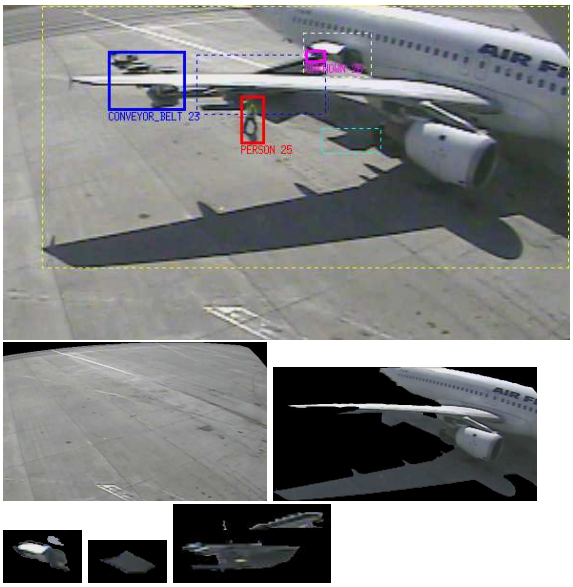


Figure 1: (Top) Frame 2352 of sequence S3-A320 showing overlapping stationary and moving objects. (Centre-left) The basic (full image) background layer. Other background layers (in order of creation) representing stationary objects: (Centre-right) the aircraft, (Lower-left) aircraft door, (Lower-centre) aircraft door shadow, and (Lower-right) partially-visible conveyor-belt vehicle.

4. Object Localisation

The localisation of an object in the context of visual surveillance generally relates to finding a location in the world co-ordinates that is most representative of that object. This is commonly taken to be the centre of gravity of the object on the ground plane and it is this definition that we adopt here. With accurate classification and detection, the localisation of vehicles in the 3D world can be reduced to a 2D geometrical problem. For state of the art algorithms accurate classification and detection is not reliable enough to apply such principled methods with confidence. For the AVITRACK project we therefore devised a simple, but effective, vehicle localisation strategy that gives good performance over a wide range of conditions.

The first step of the strategy is to categorise the detected objects as *person* or *non-person* using a supervised Gaussian mixture model of the estimated object width and height in world co-ordinates. The motivation behind this is that people generally have negligible depth compared to vehicles and hence a different strategy is required to locate each type. For the person class of objects the location is taken to be the bottom-centre of the bounding box of the detected object, this location estimate for people is commonplace in visual surveillance systems.

For vehicles many researchers arbitrarily choose the centroid of the bounding box / detected pixels to locate the object in the world. This method has the drawback that for objects further away from the camera the bottom of the bounding box is a better approximation of the object location than the centroid. To alleviate this problem we compute the angle made between the camera and the object to estimate an improved location. For a camera lying on the ground plane the location of the object will be reasonably proximal to the bottom centre of the bounding box, whereas for an object viewed directly overhead the location of the object will be closer to the measured centre of the bounding box.

Using this observation we formulated a smooth function to estimate the position of the centroid using the (2-D) angle to the object. Taking α to be the angle measured between the camera and the object, the proportion p of the vertical bounding box height (where $0 \leq p \leq 1/2$) was estimated as $p = 1/2(1 - \exp(-\lambda a))$; the parameter $\lambda \equiv ln(2)/(0.15 \times 1/2\pi)$ was determined experimentally to provide good performance over a range of test data. The vertical estimate of the object location was therfore taken to be $y_lo + p * h$ where y_lo is the bottom edge of the bounding box and h is the height of the bounding box. The horizontal estimate of the object location was measured as the horizontal centre-line of the bounding box, since this is generally a reasonable estimate. Examples of estimated vehicle centroids are shown in Figure 2, it can be seen that the estimate is closer to the actual object location than simply using the centroid of the bounding box.

Figure 2: Detected object locations (red circles) shown for 3 vehicles in the near, mid and far-field of sensor 5 for Dataset 4.

5 Experimental Results

The Scene Tracking evaluation assesses the performance of both the object tracking and the 3D localisation components on representative test data. The evaluation of the components strongly depends on the choice of the video sequences. We have chosen video datasets containing realistic conditions for an objective evaluation. All sequences are stored at a size of 720x576 pixels, and at a frame rate of 12.5 fps.

5.1 Local Feature Tracking Method

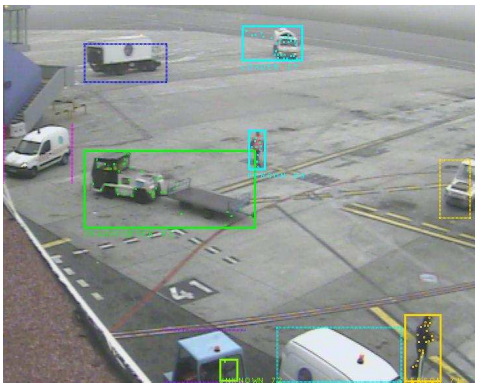
To evaluate the performance of the local feature tracking method two apron datasets were chosen. Both sequences were taken under a wide range of disturbing conditions such as illumination changes, occlusions and shadows. Dataset 1 (2400 frames) contains the presence of fog whereas Dataset 2 (1200 frames) is acquired on a sunny day. The datasets have been manually annotated using ViPER annotation tool [4]. ViPER (Video Performance Evaluation Resource) is a semi-automatic framework designed to facilitate and accelerate the creation of ground truth image sequences and evaluate performance of algorithms. The ViPER's performance evaluation tool has been used to compare the result data of the local feature tracking method with the ground truth in order to generate data describing the success or failure of the performance analysis. At first, the evaluation tool attempts to match tracked objects (TO) to ground truth objects (GTO) counting objects as matches when the following metric distance is less than a given threshold.

$$D_i(t,g) = 1 - 2Area(t_i \wedge g_i)/(Area(t_i) + Area(g_i)) \quad (13)$$

Where t_i and g_i define the bounding-box of the tracked objects and ground truth objects at frame i respectively. Once the tracked and ground truth objects have been matched true positives, false negatives and false positives objects are counted and summed up over the chosen frames. The following metrics defined by Black et al. [2] were used to characterise the tracking performance:

- Tracker detection rate (TRDR): $TP_t/(TP_t + FN_t)$
- False alarm rate (FAR): $FP_t/(TP_t + FP_t)$
- Track detection rate (TDR): $TP_o/(TP_o + FN_o)$
- Track fragmentation (TF): Number of TO matched to GTO

Where TP, FN and FP are either the total number t or the number for object o of true positives, false negatives and false positives respectively. The *TRDR* and the *FAR* metrics characterise the performance of the tracker. The *TDR* metric determines the completeness of individual ground truth objects. The *TF* metric determines the number of object label changes. It is desiderable to have a *TF* value of one. Representative results of the local feature tracking method are presented in Figure 3. Strong shadows are detected and tracked as part of the mobile objects such as the tanker from Dataset 1 and the transporter with containers from Dataset 2 (See Figure 3 (a, b)). In Figure 3 (a) a person (at the bottom on the right side) leaves the ground power unit (GPU) and in (b) a container is unloaded from the aircraft. Both objects produce a ghost which remains behind the previous object position. An object is integrated into the background when becomes stationary. In these cases, ghosts are created when stationary objects start to move again. Furthermore, ghosts



(a)

(b)

Figure 3: The results obtained from the local feature based tracking algorithm. Image (a) has been chosen from Dataset 1 and image (b) from Dataset 2.

Object	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
TP	289	551	827	601	274	200	207	72
FN	0	17	10	6	54	10	11	0
TDR	1.00	0.97	0.99	0.99	0.83	0.95	0.95	1.00
TF	3	2	3	2	3	3	1	1

Table 2: Individual object performance results of the local feature tracking algorithm for Dataset 2.

are produced when parts of the background start moving. Objects in the scene such as the container from Figure 3 (b) are partially detected due to the achromaticity of the scene. Therefore, fragmentation is presented in objects with the same colour as background.

At first, the track detection rate *TDR* and the track fragmentation *TF* were computed separately for each ground truth object. The results of the performance evaluation are depicted in Table 1 for Dataset 1 (eighteen GTO) and in Table 2 for Dataset 2 (eight GTO). Two ground truth objects were not matched to tracked objects (See Table 1, object 17 and 18). These two objects were partially detected due to their colour similarity with the background. Most of the objects from Dataset 1 present a track detection rate between

Object	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
TP	333	94	33	426	944	166	391	77	125	108	143	209	116	124	113	33	0	0
FN	3	5	10	19	2	6	32	6	32	7	6	4	9	7	3	10	310	65
TDR	0.99	0.95	0.83	0.96	0.99	0.95	0.92	0.93	0.80	0.94	0.96	0.98	0.93	0.95	0.97	0.77	0	0
TF	1	1	1	1	1	1	3	1	2	1	1	1	1	1	3	1	0	0

Table 1: Individual object performance results of the local feature tracking algorithm for Dataset 1.

Dataset	TP	FP	FN	TRDR	FAR
1	3435	275	536	0.87	0.07
2	3021	588	108	0.97	0.16

Table 3: Performance results of the local feature tracker.

92% and 99%. All ground truth objects from Dataset 2 (See Table 2) have been matched to tracked objects. Dataset 2 contains several dynamic occlusions causing tracked object label changes (See *TF* in Table 2).

In addition, the tracker detection rate *TRDR* and the false alarm rate *FAR* were calculated for whole frames. The results of this evaluation are depicted in Table 3. The presence of fog in Dataset 1 together with the achromatic nature of the scene cause a considerable number of false negatives provoking the decrease in *TRDR* (87%). Dataset 2 contains ghosts and reflections causing the increase in *FAR* (16%).

5.2 Localisation Module

For the evaluation of the 3D localisation module an individual person and vehicle have been considered. Sequence S27 (all cameras) contain individuals walking on well known trajectories along the grid of the apron. S26 (cameras 3,4,5,6) contains a single services vehicle driving on the apron for which EGNOS positional measurements were recorded[1]. To allow the comparison between the apron grid lines and the trajectories, we consider the trajectories defined by the object as paths along the apron.

3D localisation output data (e.g. Info3D(X, Y, Z=0) has been generated for each of the test cameras installed at the airport's apron. The co-ordinate Z is equal to 0 because the objects are constrained to lie on the known ground plane. For each location along the individual path the shortest Euclidean distance (in metres) is computed between the point and the associated grid line. The following performance statistics metrics are applied to the results [6]: Mean, standard deviation, minimum and maximum.

For the person class, it can be seen that person (Left) trajectory (See Figure 4) is broken due to occlusions. Occlusions lead to loss of 3D data information causing errors on 3D trajectory reconstruction. In Figure 4 the second person (Right) walks along the y=-15 grid line. The accuracy of the localisation module depends on the distance between the camera and the object due to the perspective effect and the uniform quantisation of sensor pixels. Reflections provoke errors on the reconstruction of 3D trajectories. Table 4 shows the statistic results for the eight cameras; these results demonstrate that the accuracy of the person localisation is approximately 1 metre average over all cameras, this is to be expected due to detection or calibration error. Due to the general innaccuracy in the far-field of all cameras these results show that the use of multiple overlapping cameras is justified for this surveillance system to ensure the objects are accurately located on the airport apron.

For the evaluation of the vehicle trajectory we only consider a single trajectory estimate made by the 'best' camera. The reasoning for this is that the EGNOS data was captured over a large area, and several cameras can view this trajec-

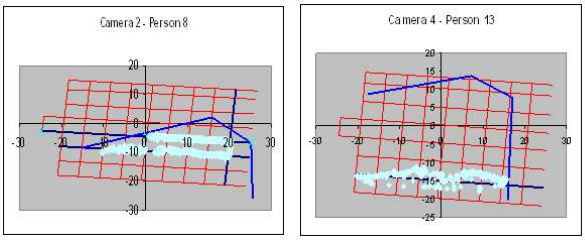


Figure 4: 2D trajectory graph for the Person object (S27, camera 2, person 8 (Left) and camera 4, person 13 (Right)). The light (red) lines represent the patching lines and the light (blue) lines represent the camera field of view.

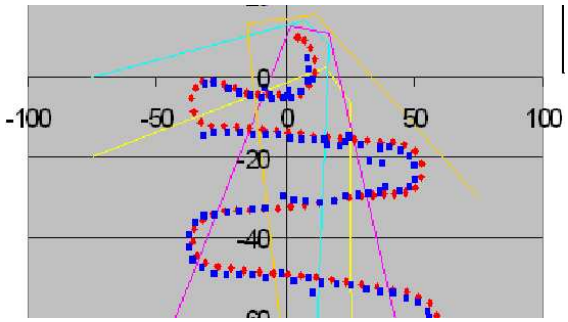


Figure 5: Vehicle 2D trajectory graph showing (Red) the EGNOS trajectory and (Blue) the estimated location on the apron. The scale is measured in metres and the camera fields of view are shown.

[1]The EGNOS measurements were kindly provided by the ESA project GAMMA (http://www.m3systems.net/project/gamma/); the EGNOS system gives an estimated accuracy of 2-3m for 95% of measurements.

Metric	C1-P27	C2-P8	C2-P12	C3-P10	C4-P10	C4-P13	C5-P8	C5-P13	C6-P27	C7-P8	C7-P25	C8-P5
Frames	148	842	501	361	432	416	419	336	431	265	164	87
Mean	0.83	0.31	0.96	0.73	0.48	1.42	0.93	0.18	2.3	0.34	0.23	0.68
STD	0.48	0.2	0.66	0.52	0.55	0.8	0.74	0.13	2.85	0.59	0.36	0.7
Min	0.14	0.02	0	0	0.01	0	0.003	0	0.01	0.001	0	0.001
Max	1.8	4.4	2.25	2.29	2.13	3.3	3.6	0.62	12.6	2.92	1.93	2.37

Table 4: 3D localisation statistics results.

tory. Therefore, at each time step, the size of the tracked object is measured in the four cameras and the one with the largest viewable object is chosen to make the trajectory estimate. In this way we are able to compare the estimated for the entire EGNOS measurement sequence.

The results, shown in Figure 5, demonstrate that the estimated vehicle location is reasonably accurate close to the camera sensors (at the top of the figure). In the far field the estimate diverges from the measured EGNOS signal due to the perspective effect and the uniform quantisation of the sensor pixels. Quantatively, the mean distance between the EGNOS signal and the estimated location was found to be 2.65 metres $+/-0.34$. The minimum deviation was found to be 0.58 metres and the maximum was found to be 4.64 metres.

6. Discussion and Future Work

The tracking and localisation of objects within a scene is a challenging problem in computer vision. In this paper we have introduced an extension of the KLT tracker that is designed to overcome some of the challenges associated with apron analysis; we also show a method for localising these objects in the world co-ordinates.

The evaluation of these methods demonstrates that the object tracking module detects a high proportion of the objects in the scene and these objects are tracked over extended time periods. Under severe partial partial occlusions we have found that the tracks become fragmented and lose the track ID. Track localisation has been shown to be accurate for vehicles and people, although naturally the accuracy reduces further from camera sensor.

Future work on the object tracker is to improve the prediction of the bounding boxes when object are undergoing occlusion and to retain the object ID's during this period. We also plan to work on the reducing the influencing of ghosts and reflections on the tracking procedure.

Acknowledgements

This work is supported by the EU, grant AVITRACK (AST3-CT-3002-502818).[2]

[2]This paper does not necessarily represent the opinion of the EU; the EU is not responsible for any use which may be made of its contents.

References

[1] J. Aguilera, H. Wildernauer, M. Kampel, M. Borg, D. Thirde, and J. Ferryman. Evaluation of motion segmentation quality for aircraft activity surveillances. In *Proc. Joint IEEE Int. Workshop on VS-PETS, Beijing*, Oct 2005.

[2] J. Black, T. Ellis, and P. Rosin. A Novel Method for Video Tracking Performance Evaluation. In *Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS), Nice, France*, pp. 125–132, 2003.

[3] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, and L. Wixson. A system for video surveillance and monitoring. In *Tech. Report CMU-RI-TR-00-12*, May 2002.

[4] D. Doermann and D. Mihalcik. Tools and Techniques for Video Performance Evaluation. In *Proceedings of the International Conference on Pattern Recognition, Barcelona, Spain*, vol 4, pp. 167–170, Sept 2000.

[5] T. Horprasert, D. Harwood, and L.S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE ICCV'99 FRAME-RATE Workshop, Kerkyra*, Sept 1999.

[6] C.J. Needham and R.D. Boyle. Performance evaluation metrics and stadistics for positional tracker evaluation. Technical report, University of Leeds, School of Computing, Jan 2003.

[7] J. Shi and C. Tomasi. Good features to track. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600, 1994.

[8] G. D. Sullivan. Visual interpretation of known objects in constrained scenes. In *Phil. Trans. R. Soc. Lon.*, vol B, 337, pp. 361–370, 1992.

[9] D. Thirde, M. Borg, V. Valentin, F. Fusier, J.Aguilera, J. Ferryman, F. Brémond, M. Thonnat, and M.Kampel. Visual surveillance for aircraft activity monitoring. In *Proc. Joint IEEE Int. Workshop on VS-PETS*, Beijing, Oct 2005.

[10] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. In *IEEE Transactions on PAMI*, vol 19 num 7, pp. 780–785, 1997.

[11] G. Xu and Z. Zhang. *Epipolar Geometry in Stereo, Motion and Object Recognition: A Unified Approach*. Kluwer Academic Publ., 1996.