AVITRACK

Contract n° AST3-CT-2003-502818

D3.6- A- Deliverable

Complex Scene Tracking Report

Version 1.0 – Draft 1

DL_AVI_2_016

Contract Number :	AST3- CT- 2003- 502818
Document Title :	3.6- A- Complex Scene Tracking Report
Document version :	1.0
Document status :	Draft 1
Date :	11- January- 2006
Availability :	Restricted
Authors :	Mark Borg (UoR), David Thirde (UoR), James Ferryman (UoR), Josep Aguilera (PRIP), Horst Wildenauer (PRIP).

Abstract

This document describes the work performed for the AVITRACK project, WP3 Task 3.6 – Complex Scene Tracking.

Keyword List

Complex Scene Tracking, Motion Detection, Object Tracking, Data Fusion, Categorisation.

DOCUMENT CHANGE LOG

Document Issue.	Date	Reasons for change
1.0 – Draft 1	27- Sept- 2005	First draft.
1.0 – Draft 1	11- Jan- 2006	Modifications & First Release.

APPLICABLE AND REFERENCE DOCUMENTS (A/R)

A/R	Reference	Title
Please Refer to the Reference Section at the end of this document.		

Table of contents

1. INTRODUCTION

This document presents the work performed at the University of Reading and PRIP as part of Task 3.6 of "Work Package 3 – Complex Scene Tracking", of the AVITRACK project [1]. Complex scene tracking addresses issues such as:

- the tracking of individuals and vehicles for the recognition of complex servicing operations,
- enhancing the robustness of the frame to frame tracking modules and the data fusion module,
- addressing tracking and data fusion issues identified in tasks 3.2 to 3.5 (See [2- 4]),
- improvements to object categorisation,
- the use of context information, and
- other general modifications that improve scene tracking for airport aprons.

2. MOTION DETECTION

Motion Detection for AVITRACK is performed by the “Frame- to- Frame Tracking” module and the work on selecting and evaluating different motion detection algorithms for the apron environment is described in the “Motion Detection Report” [2]. A formal evaluation is given in [6]. The selected motion detection algorithm is the Colour Mean and Variance algorithm and it was selected because it achieves a compromise between real- time performance and sensitivity. In brief, the main characteristics of the motion detection algorithm as implemented in the AvitrackFrameTracker module are:

- Background subtraction method which represents each pixel by a single Gaussian distribution over the Normalised RGB colour space,
- An illumination handling component, based on the work of [7], has been added to the motion detector,
- A multi- layered background model is used to allow objects which become stationary for a short period of time, to be integrated into the background model,
- A coarse- to- fine quad- tree optimisation technique was added to improve efficiency.

The following sections describe the work performed on motion detection that has not already been described in the Motion Detection Report of task 3.1 (See [2] for a description of the work done in task 3.1).

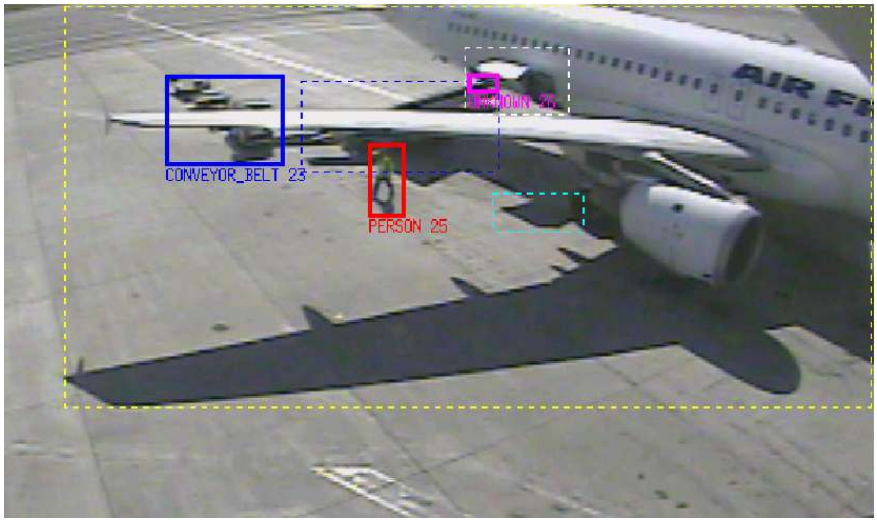
2.1. MULTI- LAYERED BACKGROUND MODEL

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. Personnel leave the vehicles, take out or move objects from the vehicles (e.g. cones), place or direct other objects on to vehicles (e.g. containers on to transporters), etc. To be able to differentiate between all these objects involved in these activities, a multi- layered background model needs to be adopted. This allows objects that become temporarily stationary to be 'put aside' and other objects moving in front of them to be identified and tracked correctly. Without such an approach, the tracker would end up with just one (useless) blob for the congested activity near the aircraft.

The use of multi- layer backgrounds for background subtraction algorithms has been used before in systems such as that described by Collins *et al*. [10]. Their method works on a pixel- 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. When this method was applied to the AVITRACK sequences, the results were poor, due mainly to stationary objects becoming fragmented into many layers for objects that remain stationary for long periods of time. This in turn results in different update rates to the layers and incorrect re- activation once the 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 representing the aircraft remains consistent throughout this time, its background layer gets updated uniformly and it is re- activated as a whole.

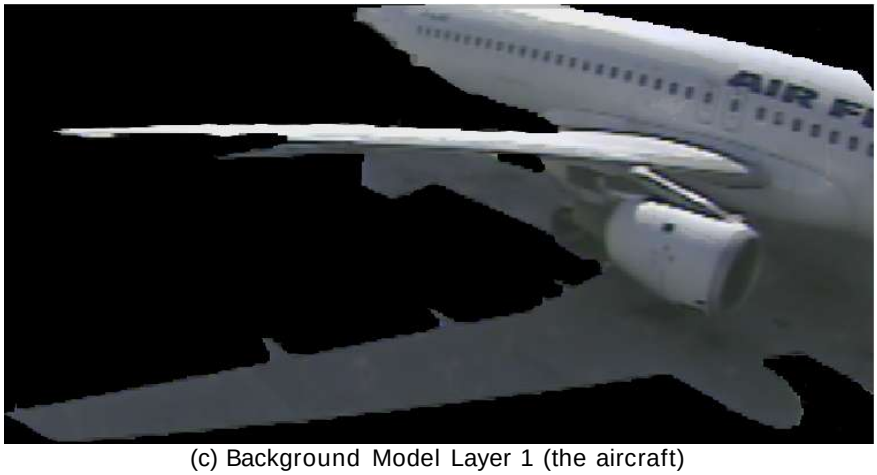
The method adopted for AVITRACK works at the region- level instead of the pixel- level and is handled by the tracker rather than at the motion detection phase, i.e. the tracker is the driver that determines when an object becomes stationary or starts moving again. The tracker will then inform the motion detector whether to create a new background layer for an object or not. The algorithm used by the tracking module is based on the KLT feature tracker [8]. Therefore, the motion information of the local features is used to determine whether a moving object has become stationary or not. In addition to using the local features, the tracker also uses a measure based on inter- frame pixel differences (and quite similar to the movement density measure described in section 2.3 below). This second measure is needed because the local features only provide a sparse sampling for the object. Combining the two together provides a robust way of detecting when object become stationary or start moving again.

Figure 1 below, illustrates an example from AVITRACK sequence S3- A320 Camera 2 and shows the back loading operation. Several stationary objects can be seen in Figure 1(a): the aircraft, the conveyor- belt vehicle, the rear hatch door and its shadow (shown with dotted bounding boxes); together with other objects such as the person moving in front, the partially- occluded driver of the conveyor- belt vehicle near the aircraft door, and a transporter vehicle in the far distance. Figure 1(b- e) shows the 5 layers making up the background model of the motion detector, in the order of how they were integrated into the background model. In particular note how the use of region- level analysis and tracking information allows the conveyor- belt vehicle to be treated as a single object when integrated into the background model – the aircraft wing partially occludes it and splits it into 2 parts. Relying only on pixel- level analysis or motion detection information, would have created 2 background layers, which in turn can cause synchronisation problems between the two parts when the conveyor- belt vehicle starts moving again.



(a) Tracking result for S3- A320 Camera 2 showing several stationary objects (dotted bounding boxes) and moving objects (solid bounding boxes) interacting together.

(b) Background Model Layer 0 (the main background layer). The top part of the image is masked out.



(c) Background Model Layer 1 (the aircraft)

(d) Background Model Layers 2 (the aircraft rear hatch door and reflection) and 3 (the hatch door's shadow on the ground).

(e) Background Model Layer 4 (the conveyor- belt vehicle, partially occluded by the aircraft's wing).

Figure 1: Multi- Layered Background Model

While background layers are ordered with respect to each other, with the most recent topmost layer hiding the layer below it, any background layer can be re- activated, even if it is not the topmost one. Background layers are kept up to date by the background update process – each visible portion of a background layer gets updated using the background update rate of the motion detector. For efficient processing, a layer of pointers, called the *active background layer*, is maintained. Each pointer points to the topmost pixel in the background model. Whenever a new layer is added or deleted, the pointers in the active background layer get updated accordingly.

Detecting Stationary Objects – Using Local Feature Movements

This method detects when objects are stationary by monitoring the movements of the individual local features f_i of an object from one frame to the next. The local features are maintained by the KLT feature tracker in the Tracking Module, and each feature is a 7x7 pixel window. If the feature's movement in the x- or y- direction is below a threshold T, then the feature is labelled as 'stationary' (T is set to be 1.0 pixels for AVITRACK):

$$|f_t(x) - f_{t-1}(x)| > T \qquad \text{or} \qquad |f_t(y) - f_{t-1}(y)| > T$$

This method works well and is very efficient. The disadvantage is that the local features are sparse and only a limited number of points in the object's region are checked; the number of local features per object is determined by a user- configurable feature density parameter, called ρ (see [2] for more detail).

Detecting Stationary Objects – Using Inter- frame Pixel Differences

The second method is based on a pixel- based difference measure from the current frame against the preceding frames. A sliding window of 6 frames (0.5 second) is used in AVITRACK. For each object blob *R:*

$$Stationary(R) = \frac{\sum_{x \in R} fd(x)}{area(R)} > T_S.$$

where: $fd(x) = 1$ if $|I_t(x) - I_{t-k}(x)| > T_1$ for any $k \in [0..6]$

Relaxation of the Stationarity Criteria

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); this case is quite common on the apron environment. Another example is when the conveyor- belt vehicle is moved slightly to and fro to align it with the aircraft hatch door while the door is in the process of being opened. This relaxation of the stationarity criterion allows the handling of partial motion as illustrated in Figure 2 below. It is performed by extending the Inter- Frame Pixel Difference method described above, to consider the overlap ratio between the bounding box of the non- stationary pixels to the bounding box of the object's blob (i.e. all the foreground pixels). If this ratio is below a certain threshold, the object is integrated into the background model as a new layer and the moving sub- part is set as a new object.

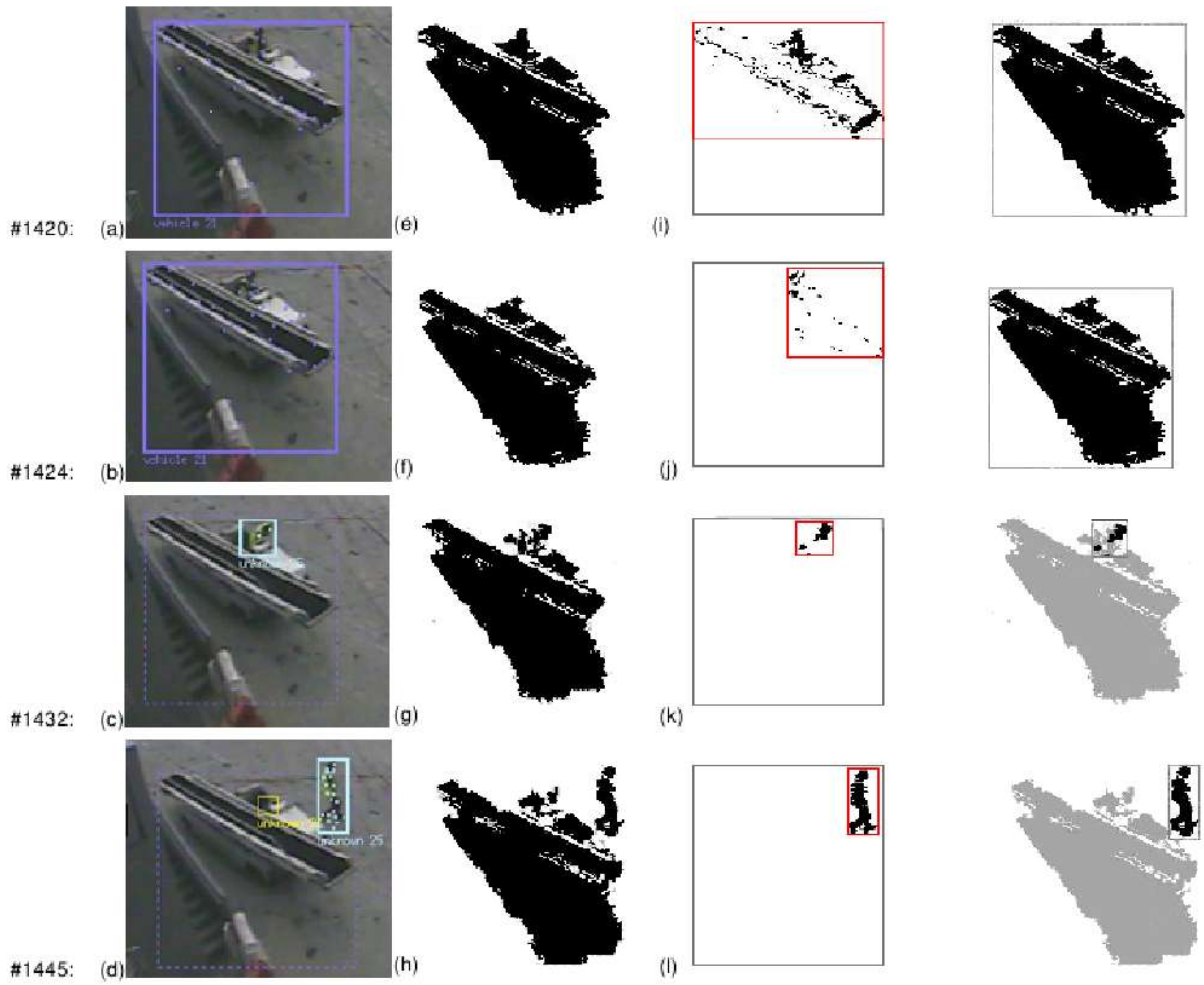
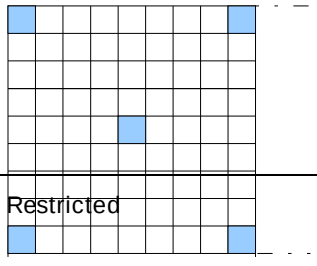


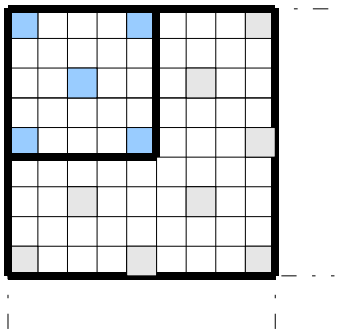
Figure 2: (a)- (d): several frames showing a conveyor- belt vehicle coming to rest, while its driver remains in motion and exits the vehicle. (e)- (h) shows the pixels labelled as foreground by the motion detector (in black). (i)- (l) the foreground pixels detected as non- stationary, using the second method (Inter- frame Pixel Differences), are shown in black. (m)- (p): the object's part in motion is shown in black, while the stationary part of the object is shown in gray. Relaxing the stationarity criteria allows the driver to be separated from the conveyor- belt vehicle in frame (c).

2.2. QUAD- TREE OPTIMISATION

To improve the real- time performance of the motion detection algorithm, a quad- tree optimisation technique has been implemented. Instead of performing motion detection at every pixel, the image is first divided into NxN pixel blocks. Motion detection is performed at the corner pixels of each block as well as the central pixel, labelling them individually as either: 'background', 'foreground', 'shadow', or 'highlight'.



If all 5 pixels have the same label, then it is assumed that all the pixels in the NxN block have the same value and no further motion detection is performed in the block. If the 5 pixels have different labels, then the NxN block is sub- divided into four (N/2)x(N/2) blocks and the process repeated for each of the four blocks:



If for any block, the corner and central pixels differ, then sub- division continues further until the blocks consist of just 1 pixel. The size of the initial blocks chosen for AVITRACK is 9x9 pixels (the default value for the configuration parameter MULTI_RESOLUTION_BLOCK_SIZE). This was found to achieve a good compromise between speed and detection sensitivity (objects of interest on the apron are normally larger than 9x9 pixels in size). In the best case scenario, a whole block has the same label, which means motion detection is evaluated at only 5 pixels from a potential of 9x9=81 pixels. Further, by overlapping the 9x9 blocks by 1 pixel, the number of pixels that need to be evaluated for each block is reduced to 2, as the top- left, bottom- left and top- right corners coincide with the corners of the surrounding blocks. Another optimisation, is to ignore blocks that have corner pixels consisting of a mixture of the labels 'shadow', 'highlight' and 'background' (i.e. having no 'foreground' label) – we don't care to know the exact shape of the highlight or shadow blob, and we set the whole 9x9 block to either 'shadow' or 'highlight', depending on the label with the highest votes.

After evaluating this quad- tree optimisation on motion detection results, it was found that it does not corrupt the boundaries/profiles of objects, while achieving an improvement in performance. Figure 3 below, shows the output from the motion detection module with quad- tree optimisation enabled and disabled for comparison, while Table 1 lists the average processing time per frame for the same video sequence.

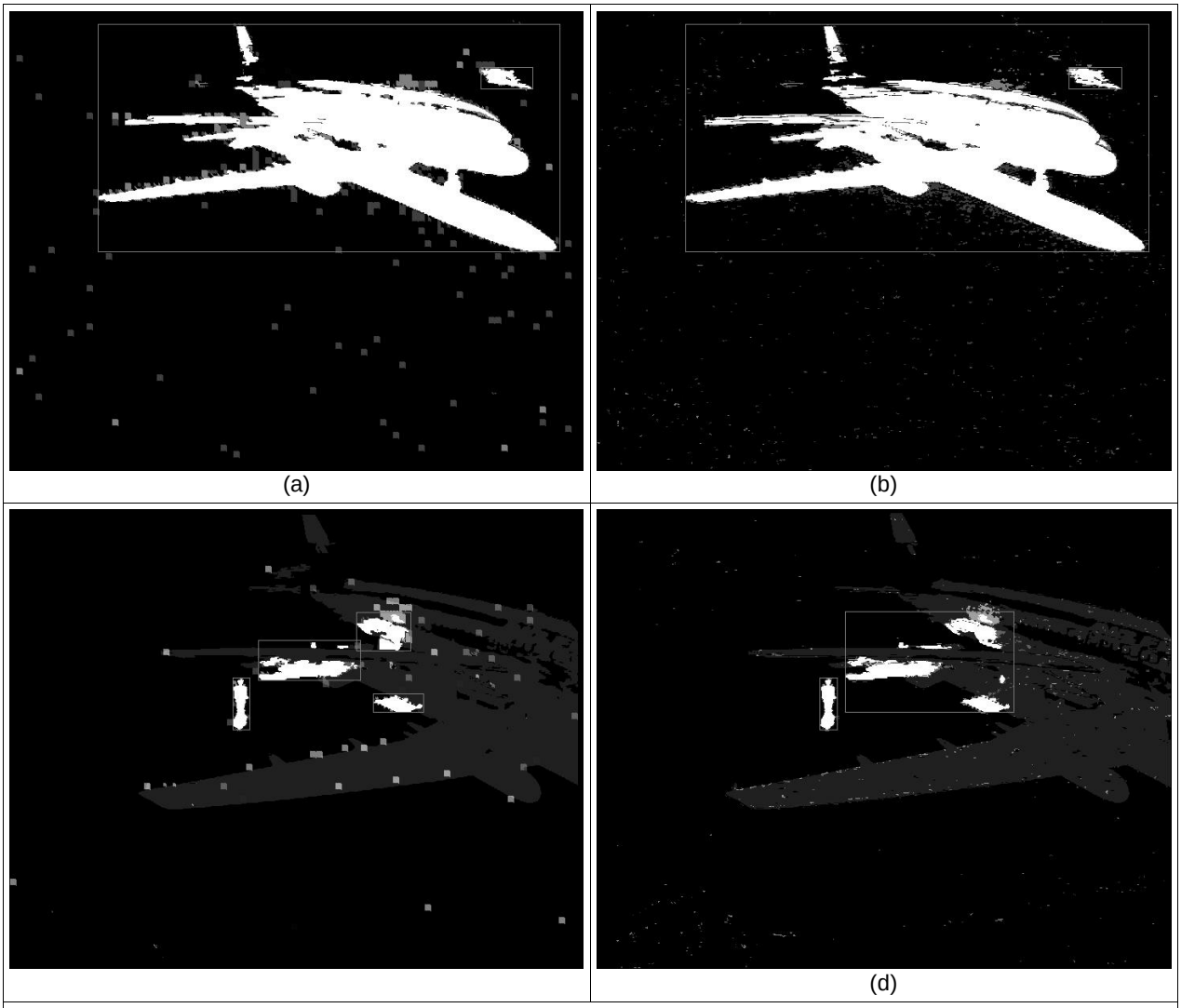


Figure 3: Motion detection results for frames 380, 1870 of sequence S3- A320 Camera 2: (a),(c) with quad- tree optimisation enabled, using 9x9 block size. (b), (d) with quad- tree optimisation disabled. Foreground pixels shown in white, highlight pixels in light gray, shadow pixels in dark grey, and pixels belonging to stationary objects (the aircraft) shown in darker grey. Note how the 9x9 block size does not corrupt the object outlines. Note also the appearance of the shadow and highlight blocks – these are not further sub- divided as these blobs are unused.

	average FPS	**min FPS**	**max FPS**
with 9x9 quad- tree optimisation	64.8	30.3	126.1
no optimisation	12.1	6.93	15.8

Table 1: Motion Detection speed (in Frames Per Second) with and without optimisation, for first 2000 frames of sequence S3- A320 Camera 2.

2.3. GHOST DETECTION

A *ghost* is defined as a set of connected pixels, detected as a mobile object but not corresponding to any real moving object.

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 are produced when parts of the background start moving.

A movement density measure introduced by Ruiz- del- Solar *et al* [9] is adopted to detect ghosts in the scene.

Movement Density Module

The movement density module receives the detections from the motion detection algorithm. Movement pixels are identified and connected by means of 8- connectivity into blobs. For each blob *b* is defined a movement density MD_b as:

$$MD_b = \frac{\sum_{x \in b} |I_k(x) - I_{k-1}(x)|}{Area(b)}$$

Where *x* are pixels belonging to a blob *b* and I_k is an image at frame *k.* The movement density measures for the blob the average change in the last frame. Ghosts should have a low movement density, while the moving objects should have a larger movement density. A threshold T_b is defined and set to 2.5. Blobs with a movement density measure under T_b are considered ghosts and are discarded.

In the next section, representative results of the movement density module for **S21** and **S28** datasets are presented.

Tracking Results without Ghost Detector	Tracking Results without Ghost Detector
S21 – Cam 8, # 1724	S21 – Cam 8, # 1772
	

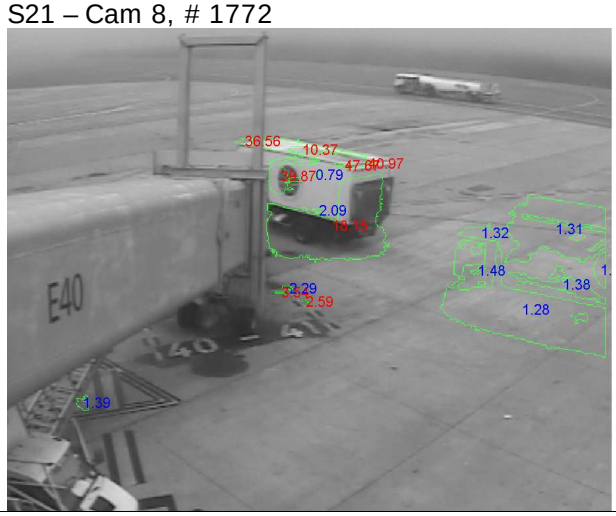
Movement Density Module Results	Movement Density Module Results
S21 – Cam 8, # 1724	S21 – Cam 8, # 1772
	

Figure 4

Tracking Results without Ghost Detector	Tracking Results without Ghost Detector
S21 – Cam 7, # 5860	S28 – Cam 5, # 1158
	

Movement Density Module Results	Movement Density Module Results
S21 – Cam 7, # 5860	S28 – Cam5, # 1158
	

Figure 5

In the tracking results without ghost detector at frame 1724 a person (at the bottom on the left side) starts to walk and at frame 1772 a catering vehicle starts to move. In addition, at frame 5860 a person (at the bottom on the right side) leaves the ground power unit (GPU) and at frame 1158 a container is unloaded from the aircraft.

In all previous cases, the moving objects produce a ghost which remains behind the previous object position.

In the movement density module images, blobs are depicted with colour green and the movement density blob values MD_b either with colour red or blue depending if the blob is detected as mobile object or as ghost respectively. Looking at the results of the movement

density module we can see that the ghost detector is able to identify correctly all previous miss-classified blobs detected as mobile objects.

2.4. THE PROBLEM OF REFLECTIONS

Reflections, as present in motion detection results, are pixels incorrectly labelled as 'motion' which arise due to changes in illumination. These can be caused by specular reflections, sudden brightening, light reflections due to presence of water, lens effects, etc.

For most illumination changes, the brightness- chromaticity test as implemented in the motion detector is adequate for eliminating false motion detections. This test is based on the observation that an illumination change will change the brightness component of a pixel's background model, but not the chromatic components (See [2] for more detail). But this method does not work for sudden large illumination changes or where the chromaticity varies as well.

In AVITRACK sequences, the reflections causing most problems are due to two effects: i) reflections caused by paint markings (so- called dry reflections) and ii) reflections caused by standing water present on the apron (wet reflections), normally from de- icing operations.

PAINT MARKING REFLECTIONS

The airport apron is covered by a network of reflective paint markings that reflect light from the sky and from objects passing nearby. The figure below illustrates how paint marking reflections create spurious motion detection blobs which add an error to the 3D localisation of objects.

Figure 6: (Left) A van moving on the apron near paint markings. (Right) Reflections from the paint markings create spurious blobs (shown in black) in the motion detection results that are merged with the van's blob, creating an error in the van's 3D localisation.

To tackle this problem for AVITRACK, it was decided to focus on cases where the reflection blob (s) are distinct from the object's blob(s), such as in the above example. This method is based on the idea that the gradient orientation of the current frame compared to the background image (of the motion detector) should not change for background pixels affected only by reflection. The gradient orientation at each pixel is obtained from the orientation angle θ of the 2D structured tensor. From this, one can then obtain the unit length of the orientation vector as:

$$n_1 = \sin(\theta \quad)$$

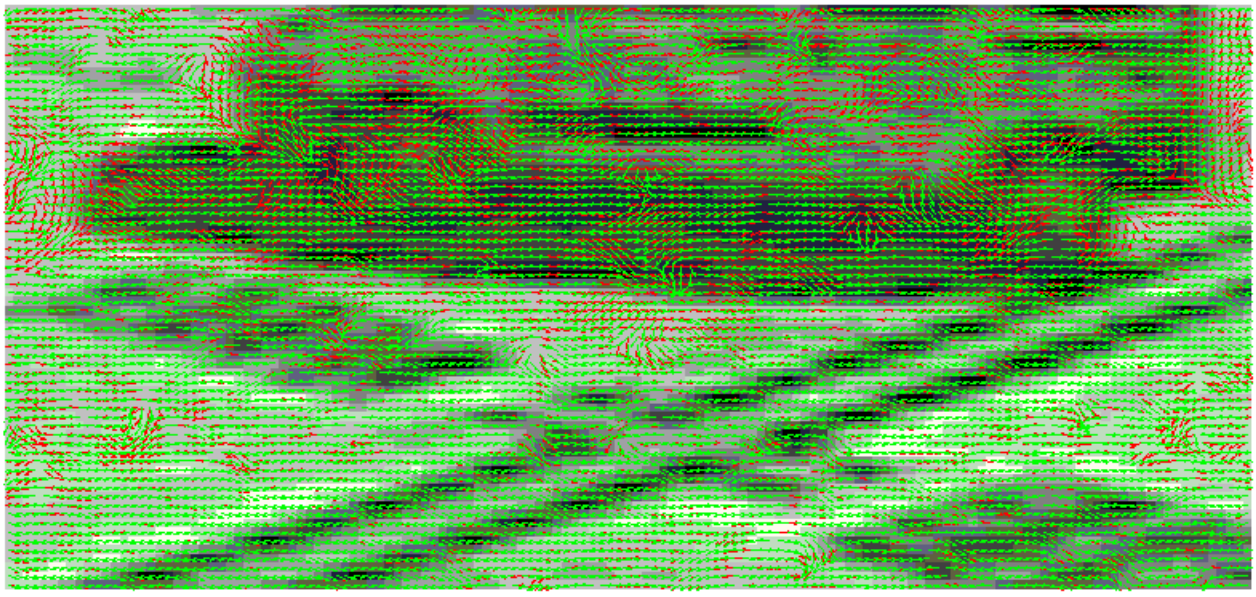
$$n_2 = -\cos(\theta \quad)$$

The difference *d* in the orientation between an image pixel and the background image pixel is estimated by:

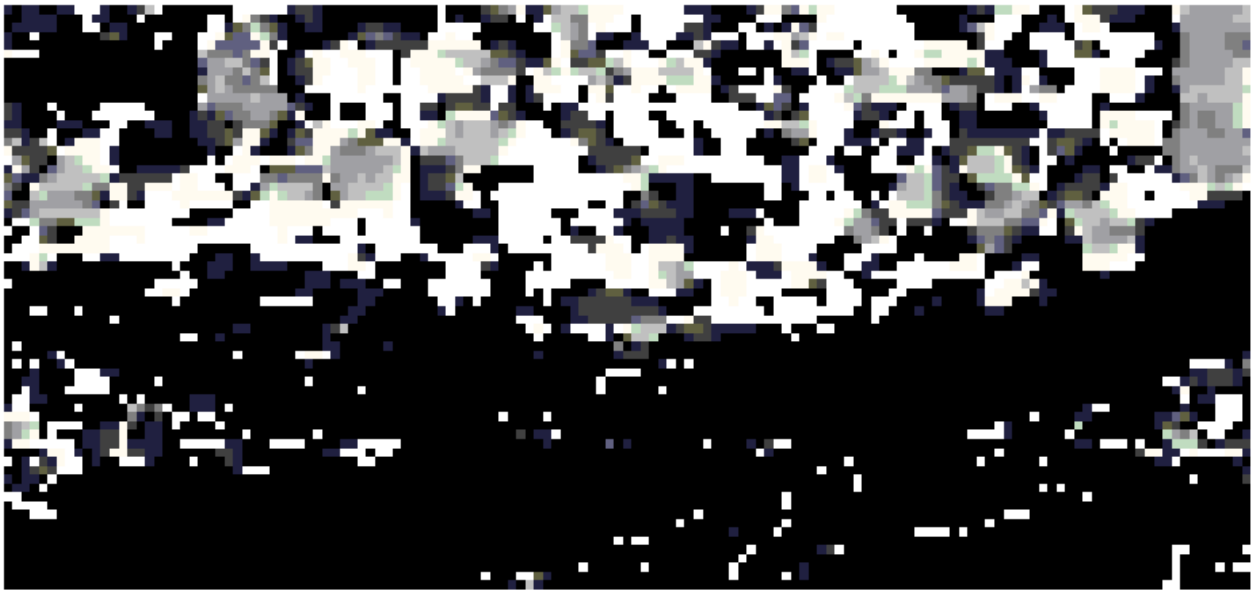
$$d = (n_{1,B} - n_{1,t})^2 + (n_{2,B} - n_{2,t})^2.$$

where $n_{1,B}$ is the value n_1 of the background pixel and $n_{1,t}$ is the value of the image pixel at time *t*. A threshold is then applied to the orientation differences. For pixels affected by paint marking reflections, the orientations of the background and image pixel should agree. Figure 7 below shows the result of this method.

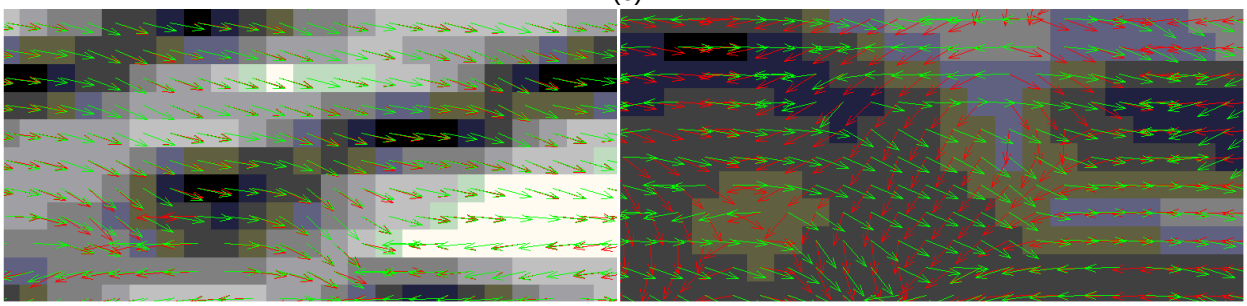
(a)



(b)



(c)



(d) (e)

Figure 7: Experimental results on part of the frame shown in the previous figure. (a) the image part containing object and reflections; (b) image showing gradient differences (red and green vectors) between background image and current frame; (c) after thresholding the differences, most of the reflection pixels initially detected as motion have been eliminated; (d,e) close- up regions of (b) showing how most of the gradient vectors align together in areas of reflections (d) and mis- alignment in areas belonging to the van (e).

The initial experimental results of this method appear to be able to solve the problem of paint marking reflections. As can be seen in the previous figure, a histogram- based mechanism or neighbourhood- based support is required in order to determine whether a blob is caused by reflections or not, i.e. to move from the pixel- level analysis to blob- level analysis. Initial results look promising. The work of integrating this method in the AvitrackFrameTracker is ongoing.

3. OBJECT TRACKING

Object Tracking for AVITRACK is performed by the "Frame- to- Frame Tracking" module and the work on selecting and evaluating different tracking algorithms for the apron environment is described in the "Scene Tracking Report" [3]. A formal evaluation is given in [6]. The selected object tracking algorithm is based on the KLT feature tracking algorithm [8]. In brief, the main characteristics of this algorithm as implemented in the AvitrackFrameTracker module are:

- Sparse local features are tracked and used to match object predictions to observations seen by the motion detector,
- Feature tracking is incorporated into a higher- level framework that handles complex object interactions, such as object merging, splitting, etc.

The following sections describe the work performed on object tracking that has not already been described in the Scene Tracking Report of task 3.4 & 3.5 (See [3] for a description of the work done in tasks 3.4 & 3.5) and addresses the issues and objectives mentioned under complex scene tracking.

3.1. MOTION SEGMENTATION

The first version of the AVITRACK frame- to- frame tracking module used the spatial information of sparse local features to track objects over time (e.g. spatial proximity and membership of features within a blob's area). The KLT algorithm was used together with a rule- based framework to handle complex object interactions such as objects appearing to merge, partially occlude each other, etc. This process is described in detail in the Scene Tracking Report [3].

In many of the aircraft servicing operations, such as the front- door loading operation, a lot of activity occurs in which objects remain merged together for extended periods of time. While in this merged state, the objects move relative to each other and new objects may appear (e.g. containers coming out of the aircraft). To be able to differentiate these objects, in addition to the spatial information of features, the motion information of the features was utilised.

This use of *motion segmentation* is based on the idea that features belonging to an object (or parts of it, if the object is articulated) should follow approximately the same motion (assuming rigid object motion). As objects move differently with respect to each other, they can be segmented out when in a merged state by analysing the motion of the local features. The motion of each individual object is robustly fitted to a *motion model* and then used to identify to which motion model (hence which object) a feature belongs to. Different motion models can be used such as a simple translational model, affine motion model, projective motion model, etc. For AVITRACK, the motion models implemented for the apron environment are the translational and affine motion models.

For each object, motion models are fitted to each group of *K* neighbouring features. This use of *K* neighbours helps to reduce the effects of outlier features (i.e. features incorrectly matched by the KLT algorithm from one frame to the next). For AVITRACK, *K* is set to 4. 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]. See Figure 8 below for an example of motion segmentation as performed on the features.

Figure 8: (Left) Sample frame from Dataset S28- A320 Camera 5 showing two vehicles (transporter on the left, loader on the right) that appear merged together for an extended period of time, while they move relative to each other. The transporter vehicle is moving slowly towards the camera, while the stationary loader vehicle is raising its platform. (Centre) Performing motion segmentation of the features of the 2 vehicles by fitting motion models and clustering. (Right) After performing clustering in the motion parameter space, 3 main motions are identified – a vertical motion displayed as red points that explains the upward motion of the loader's platform, a stationary motion (shown as white points) for the part of the loader vehicle that is stationary, and a forward motion (green) for the transporter vehicle.

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. Similarly new objects could be identified by their unexplained motion. Thus, though the underlying assumption is of rigid object motion (translational or affine), the use of a weighted list of motion models allows for some limited variation from rigid motion. This is especially useful to handle the different motions for articulated vehicles (e.g. the loader's platform).

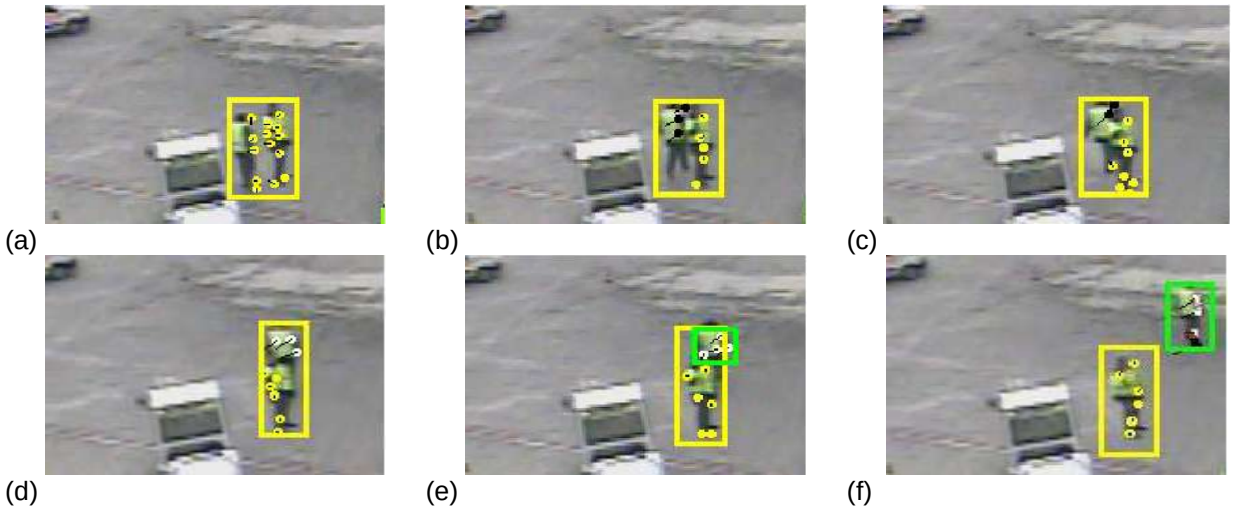


Figure 9: (a) shows two quasi- stationary persons merged together, with the features highlighted in yellow and explained by a single motion model; (b,c) as the person on the left starts moving, the motion of its features (shown in black) create a secondary motion with initially low confidence; (d) confidence in the secondary motion model increases (turning to white circles), until (e) the confidence is high enough for the secondary motion to trigger the creation of a new object; (f) the two persons are no longer merged.

AFFINE MOTION MODEL

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

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

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:

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

f_{nm} is obtained through a minimisation process based on eigen analysis, as described in [11]. The affine motion model is then represented in terms of the following 5 motion parameters, since the non- zero values f_{nm} of F are highly correlated and cannot be used directly.

$$\alpha = \arctan\left(\frac{-f_{13}}{f_{23}}\right)$$

$$\gamma = \arctan\left(\frac{f_{31}}{-f_{32}}\right)$$

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

$$\lambda$$

$$\theta = \alpha - \gamma$$

Clustering is then performed in the 5- dimensional motion parameter space $\{\alpha,\gamma,\rho,\lambda,\theta\}$ to get the list of the most significant motion models for the object.

TRANSLATIONAL MOTION MODEL

The second motion model used in AVITRACK is the translational motion in the image plane:

$$v_{translational} = w_t - w_{t-N}.$$

where w_t and w_{t-N} are the locations of feature w at time $t, t-N$. The motions are then plotted in a 2- dimensional space and clustering is performed to find the significant motions.

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. In the AvitrackFrameTracking module, it is possible to configure which of these motion models to use, by activating directives such as USE_AFFINE_MOTION_MODEL.

4. OBJECT CATEGORISATION

Object Categorisation for AVITRACK is performed by the "Frame- to- Frame Tracking" module and the work on the different methods adopted for classifying objects on the apron are described in the "Object Categorisation / Recognition Report" [5]. A formal evaluation is given in [6]. The selected classification method uses a hierarchical approach that combines bottom- up low- level classification with top- down model- based object recognition. In brief, the main characteristics of this algorithm as implemented in the AvitrackFrameTracker module are:

- Bottom- up object classification into the main categories (vehicle,person,aircraft,equipment,other) using a gaussian mixture model and descriptors such as 3D height and width, dispersedness, etc.,
- A top- down classifier used to recognise different vehicle types by 3D model- based recognition using appearance information.
- A hierarchical classification method that combines both the bottom- up classification method mentioned above and the top- down model- based classifier.

The following sections describe the work performed on object categorisation that has not already been described in the Categorisation/Recognition Report of task 3.3. (See [6] for a description of the work done in task 3.3).

4.1. FACET MODEL CLASSIFICATION

The facet model based classification method is quite computationally intensive because of the many models of vehicles that need to be recognised in the AVITRACK environment. As described in [6], the model- based classifier is combined in a hierarchical fashion with the bottom- up approach to ensure real- time performance. The current work has been aimed at further improving the performance of the facet model classifier.

CYLINDRICAL MODELS

The first version of the facet model classifier included in the FrameTracker module only allowed the modelling of planar vehicles. To handle the tanker vehicle and the aircraft, support for cylindrical models was added to AVITRACK. A facet approximation to cylinders is constructed so allowing the same *prim* file format to be used.

Figure 10: Use of cylindrical models for representing vehicles such as the refuelling tanker.

Cylinders are defined by a starting point and an end point – these two points define the cylinder's axis. Each of the end points has an associated radius. For a cylinder, the radii should be the same; if different, then a tapered cylinder (or cone) is generated. The user selects whether the cylinder has end caps or not – if the cylinder is open or closed. The user can also specify the number of facets used to approximate the curved side of the cylinder – the higher the number of facets, the smoother the cylinder is. Another option is whether to include lines in the primitive file between the edges of the facets used for approximating the curved side.

PARTIAL MODELS

Certain vehicles are quite complex and may require hundreds of facets to be modelled accurately enough for performing model fitting, e.g. the aircraft. For efficiency purposes, the facet model code was modified to allow the definition and the model fitting of partial models. Figure 11 below shows the partial model used for the aircraft, where only part of the fuselage and the two engines are defined. These parts are distinct enough to still give a high score and allow the aircraft to be discriminated from the other objects, as well as recovering an accurate pose for the aircraft.

(a) (b)

Figure 11: (a) the aircraft's partial model; (b) and as used for model fitting

FACET MODEL APPEARANCE LEARNING TOOL

To aid the AVITRACK user in constructing appearance models for different vehicles, a utility was created called *AppearanceModelInit*. This allows the user to interactively load a model, project it into a camera view (using that camera's calibration information), position it anywhere on the apron, and learn the facet's appearance model from the test images. Selecting different frames from a video sequence allows the model to be seen from different orientations, so that an appearance model can be learnt for all its facets. Figure 12 below shows a screenshot of this application. Further information about using this utility is given in [12].

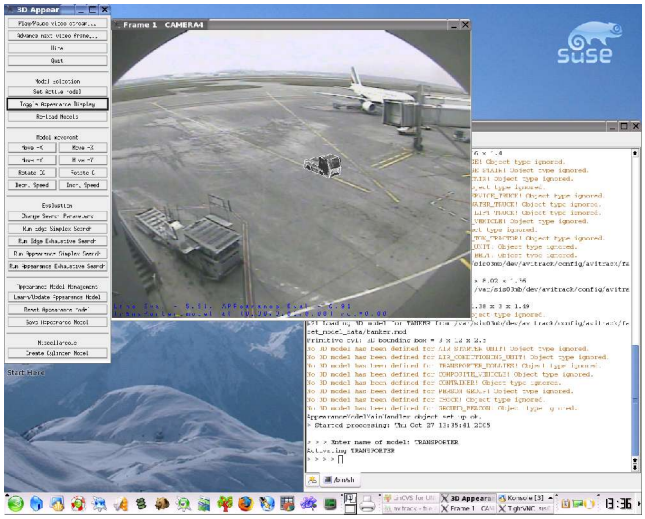


Figure 12: Screenshot of the *AppearanceModelInit* utility

SEARCH REGIONS FOR MODEL FITTING

To improve the speed of model fitting, some basic context information is used. This consists of allowing the user to specify search regions on the apron for certain vehicles. For example, the jet bridge object can be in only a small set of locations on the apron; its possible set of orientations is also restricted to a small range. The user can specify:

$$[x \ldots x \quad y \ldots y \quad \theta \qquad \ldots \theta \qquad].$$

where (x,y) are the position on the ground plane in terms of the calibration's world coordinate system and θ is the orientation range. The long- term aim is to make use of the complete static scene model that was built for the scene understanding module.

Figure 13: Using context-based search information (search region and orientation range) for specific objects. In this case the jet bridge can only be in a small set of positions. The X,Y search region on the ground plane is shown as a black rectangle on the apron.

USING DIFFERENT SEARCH METHODS

The first version of the facet model categorisation module used the SIMPLEX algorithm for finding the pose of a model that best fits the given image data. SIMPLEX is a fast algorithm, simple and quite robust. But its convergence is highly dependent on the initial search pose and so can easily converge to a local minimum instead of a global one (See [5] for more on this).

To improve the model fitting, the plan is to use different search algorithms and evaluate them to find the best performing one for AVITRACK. The search methods to be evaluated are:

- SIMPLEX algorithm [13],
- Simulated Annealing with SIMPLEX algorithm [13],
- Stochastic Diffusion Search (SDS) algorithm [14].

5. DATA FUSION

Fusion of tracking results for AVITRACK is performed by the “Data Fusion” module and the work describing the data fusion algorithm used for the apron environment is described in the “Data Fusion Report” [4]. A formal evaluation is given in [6]. The basic data fusion algorithm presented in the report is based on a discrete nearest neighbour Kalman filter technique. Advanced topics for data fusion that were introduced to address the issues and objectives required for multi-camera complex scene tracking are:

- Measurement confidence used to determine reliability of observations made in 2D frame-to-frame trackers.

- Association using an extended validation gate using spatial information, velocity and category information to improve tracking in congested apron regions.
- Epipolar data association with extensions to nearest neighbour association, allowing people to be tracked above the ground plane (e.g. on stairs)
- Implementation of standard (i.e. spatial) JPDA filter.
- Improved estimated object properties using filtering and prior knowledge
- Added contextual information (basic) for challenging scenarios (aircraft, jetbridge)

The following sections describe in more detail this work, which has not already been described in the Data Fusion Report of task 3.2. (See [4] for a description of the work done in task 3.2).

5.1. MEASUREMENT CONFIDENCE

To improve reasoning in the data fusion module, we introduce a confidence measure that the 2-D measurement represents the whole object. Localisation is generally inaccurate when clipping occurs at the left, bottom or right- hand image borders when objects enter/exit the scene. The confidence measure $\psi \in [0,1]$ is estimated using a linear ramp function at the image borders (with $\psi = 1$ representing `confident' i.e. the object is unlikely to be clipped). A single confidence estimate Ψ_{o_i} for an object O_i is computed as a product over the processed bounding box edges for each object.

5.2. EXTENDED VALIDATION GATE

The validated set of measurements are extracted using a validation gate [15], this is applied to limit the potential matches between existing tracks and observations. In previous tracking work the gate generally represents the uncertainty in the spatial location of the object; in apron analysis this strategy often fails when large and small objects are interacting in close proximity on the congested apron, the uncertainty of the measurement is greater for larger objects hence using spatial proximity alone larger objects can often be misassociated with the small tracks. To circumvent this problem we have extended the validation gate to incorporate velocity and category information, allowing greater discrimination when associating tracks and observations.

The observed measurement is a 7-D vector $\boldsymbol{Z} = [x, y, \dot{x}, \dot{y}, P(p), P(v), P(a)]^T$ where $P(\ldots)$ is the probability estimate that the object is one of the three main taxanomic categories (p = Person, v = Vehicle, a = Aircraft). This extended gate allows objects to be validated based on spatial location, motion and category, which improves the accuracy in congested apron regions. The effective volume of the gate is determined by a threshold t on the normalised innovation squared distance between the predicted track states and the observed measurements:

$$d_k^2(i,j) = \left[\mathbf{H}\hat{\mathbf{X}}_k^-(i) - \mathbf{Z}_k(j)\right]^T \mathbf{S}_k^{-1} \left[\mathbf{H}\hat{\mathbf{X}}_k^-(i) - \mathbf{Z}_k(j)\right]$$

where

$$\mathbf{S}_k = \mathbf{H}\hat{\mathbf{P}}_k^-(i)\mathbf{H}^T + \mathbf{R}_k(j)$$

is the innovation covariance between the track and the measurement; this takes the form:

$$\mathbf{S}_k = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & 0 & 0 & 0 & 0 & 0 \\ \sigma_{yx} & \sigma_y^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}} & 0 & 0 & 0 \\ 0 & 0 & \sigma_{\dot{y}\dot{x}} & \sigma_{\dot{y}}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{P(p)}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{P(v)}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{P(a)}^2 \end{bmatrix}$$

For the kinematic terms the predicted state uncertainty $\hat{P}_k^-$ is taken from the Kalman filter and constant *a priori* estimates are used for the probability terms. Similarly, the measurement noise covariance $\boldsymbol{R}$ is estimated for the kinematic terms by propagating a nominal image plane uncertainty into the world co- ordinate system using the method presented in [16]. Measurement noise for the probability terms is determined \emph{a priori}. An appropriate gate threshold can be determined from tables of the chi- square distribution[15]. For epipolar data association the validation gate also includes estimated height information.

The performance is shown in Figure 14 where estimated objects on the ground plane are shown for the two test sequences. It is clear to see that by extending the validation gate to include velocity and category, as well as the use of measurement confidence in the fusion process, the extended NNDA filter out- performs the standard (i.e. spatial validation and fusion) process. Many more objects estimated by the extended filter are contiguous, with less fragmentation and more robust matching between measurements and existing tracks. It can be seen that the extended filter is robust against objects that are not on the ground- plane (e.g. the containers on the loader in S28). This is achieved by using camera line- of- sight to determine that the container observations do not agree between the cameras and hence the estimated object is given a lower confidence.

The results are encouraging, for many scenarios the extension of the validation gate provides much greater stability, especially when objects are interacting in close proximity. It is noted that the track identity can be lost when the object motion is not well modelled by the Kalman filter or when tracks are associated with spurious measurements. The filter currently has no contextual information about the 3D geometry of the scene, therefore the camera line- of- sight cannot be accurately determined. Due to this factor, objects can have lower than expected confidence since some camera measurements cannot be made due to occlusions. The addition of contextual information would also allow the tracking of large objects when they are off the ground- plane (e.g. the containers in S28). For larger objects epipolar analysis is not practical, therefore contextual information about the loader vehicle would be required to position the container objects correctly.

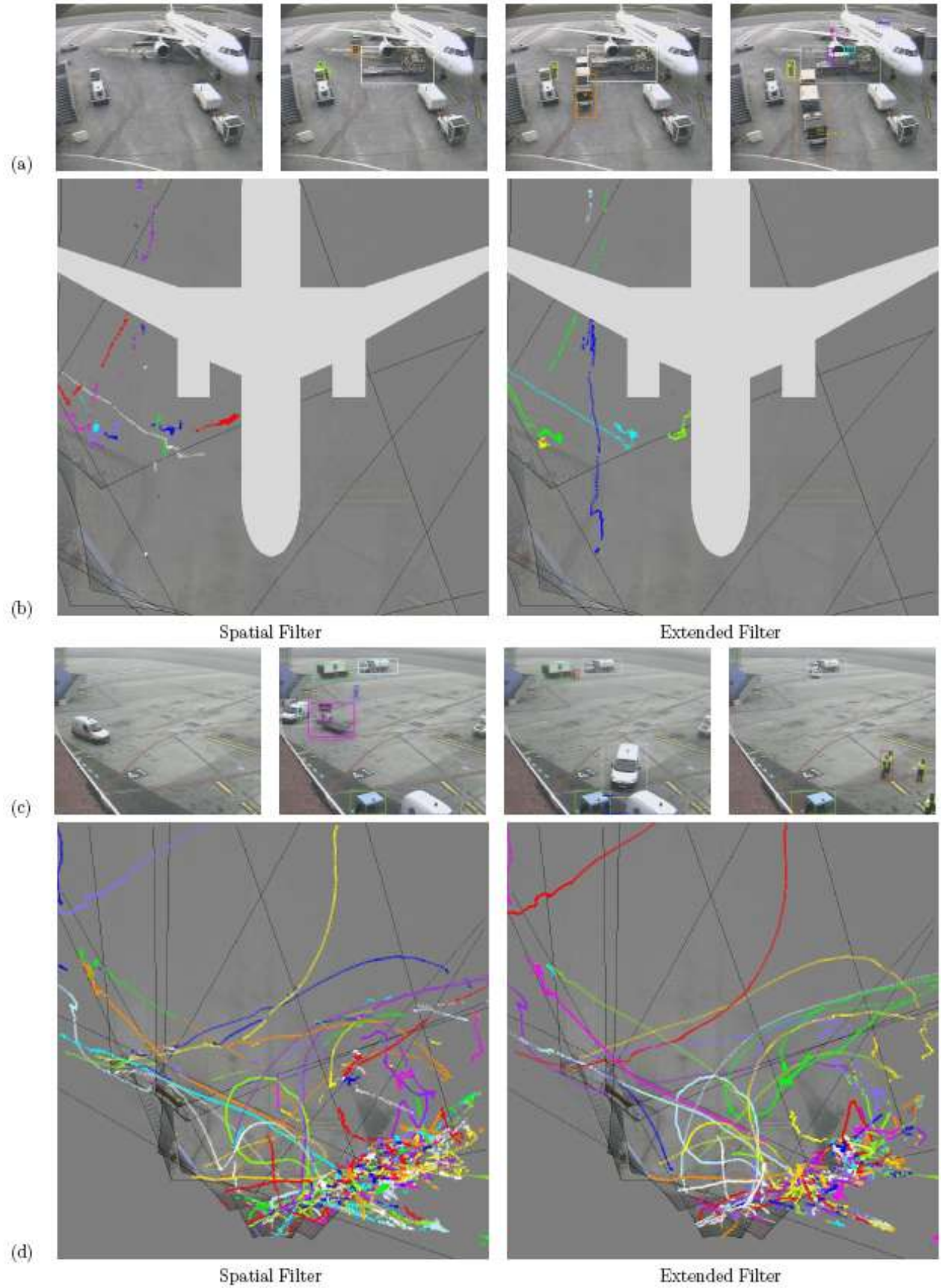

(a)
(b)
Spatial Filter
Extended Filter
(c)
(d)
Spatial Filter
Extended Filter

Figure 14: Results of the multi- camera tracking module showing extracted object locations on the ground- plane for two data sets. The track colour is derived from the object ID, limited to eight colours for visualisation. (a) S28 - All cameras frames 0, 500, 750, 1000. (b) Objects tracked by the NNDA filter with (Extended Filter) and without (Spatial Filter) the extended validation gate and confidence based fusion. The aircraft is added for illustrative purposes. (c) S21 - All cameras frames 0, 6000, 7000, 9000. (d) Objects tracked by the NNDA filter with (Extended Filter) and without (Spatial Filter) the extended validation gate and confidence based fusion.

5.3. EPIPOLAR DATA ASSOCIATION

To track objects that cannot be located on the ground plane we have extended the tracker to perform epipolar data association (based on the method presented in [16]), this can either be run in standalone mode or as an extension to the ground- plane tracking system. The epipolar data association method is a technique for associating per- camera observations *independent* of the existing objects. This method is performed as follows:

1. Associate per- camera observations using the epipolar plane constraint.
2. The associated measurements are formed into fused observations using a method based on the covariance intersection approach, the estimated intersection point of the epipolar lines is used to locate the fused observation.
3. Associate the fused observations with the existing tracks (since this relationship is not known), this is achieved using a variant of the NNDA filter.

A representative result for the epipolar based data association method is shown in Figure 15. In further experiments this method has been demonstrated to track off the ground plane and partially occluded objects, although can be prone to noise due to least squares solution. The extension of the NNDA filter is difficult since the epipolar tracker detects some of the objects filtered out by the NNDA. Further work is required to robustify the epipolar method to noisy measurements, while retaining the flexibility to detect objects that are off the ground plane or occluded. The handover between both types of filter appears to be sufficiently handled using the validation gate with height information.

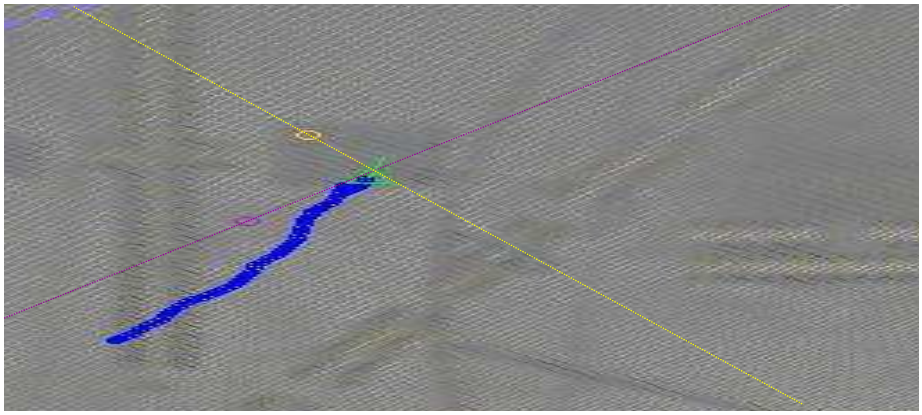


Figure 15: S4 tracking result showing epipolar based data association for a person object. The yellow and purple lines show the epipolar lines from each camera, the yellow and purple circles are the observations (well separated due to poor localisation). The track location is marked by the green triangle, at the intersection of the epipolar lines.

5.4. JPDA IMPLEMENTATION

The discrete nature of the NNDA filter leads to a degradation of performance in the presence of noise, where the chance of misassociation is increased. To improve the robustness in the presence of noise the JPDA filter analyses the neighbourhood of the track estimate to compute the joint probability of association between each measurement and track combination. Briefly, the JPDA filter is performed as follows (see [15] for further details):

1. Cluster tracks into extended validation regions using the intersection of validation gates.
2. For each extended validation region, generate all feasible hypotheses of track to measurement associations. The feasibility constraint requires that each track generates at most one measurement and that each measurement corresponds to only one track.
3. Compute the probabilities of the feasible hypotheses
4. Find the association probability between a track and a measurement by summating the hypothesis probabilities for all hypotheses in which the measurement occurs.
5. Compute the combined innovation for use in the sequential Kalman filter update using the standard PDA filter expressions.

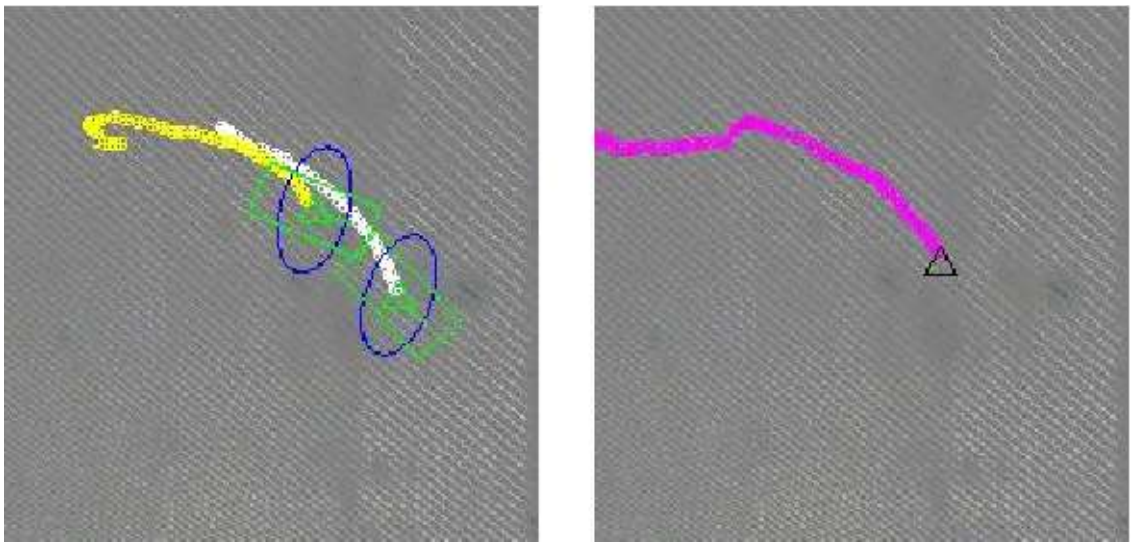


Figure 16: S4 tracking result (Left) NNDA filter with spatial validation gate (Right) JPDA filter with spatial validation gate. The NNDA validation regions (blue) and vehicle estimates (green) are not displayed for the JPDA filter.

A representative result comparing JPDA and NNDA filters is shown in Figure 16. In this example a single vehicle has been misdetected as two separate racks for this camera, since the nearest neighbour performs discrete association two objects are created. For the JPDA filter the most likely hypothesis was that these belonged to a single track, and the vehicle is tracked successfully. Interestingly, this example highlights a potential problem with the data fusion on the apron. The vehicle to be tracked is articulated, and the two objects detected by the NNDA are the vehicle and the trailer. A common problem with the JPDA is that closely interacting objects may be merged into single tracks. The balance between tracking in the presence of noise and tracking in violation of the assumptions of the data association filters needs to be explicitly addressed.

5.5. ESTIMATED OBJECT PROPERTIES

In the NNDA and EDA filters the matched observations are combined to find the fused estimate of the object, this is achieved using covariance intersection. This method estimates the fused uncertainty $\boldsymbol{R}_{fused}$ for the N matched observations as a weighted summation:

$$\mathbf{R}_{fused} = \left(w_1\mathbf{R}_1^{-1} + \ldots + w_N\mathbf{R}_N^{-1}\right)^{-1}$$

where $w_i = \frac{w_i'}{\sum_{j=1}^{N} w_j'}$ and $w_i = \psi_i^n$ is the confidence of the i 'th associated observation (made by camera c) estimated using the method in Section 5.1. Sequential Kalman filter update was used in the JPDA filter to estimate the object states from associated measurements (using the standard equations in [15]).

If tracks are not associated using the extended validation gate the requirements are relaxed such that objects with inaccurate velocity or category measurements can still be associated. Remaining unassociated measurements are fused into new tracks, using a validation gate between observations to constrain the association and fusion steps.

For object category estimation the object category estimates for all fused measurements are averaged, weighted by the confidence of the observation. The velocity, speed and orientation are computed from the estimated location of the object. The category and velocity information is filtered in an alpha- beta IIR filter of the form $E^+ = \alpha E^- + (1-\alpha)O$ where E^- is the previous estimate of the value, O is the observed value and E^+ is the updated (filtered) estimate.

5.6. CONTEXTUAL INFORMATION

For certain events predefined contextual rules have to be added to the tracking module e.g. during aircraft arrival a global association of the per- camera aircraft tracks is made to circumvent the problem that no single camera observes the whole object. Contextual global rules are implemented in the data fusion for the aircraft and jetbridge objects.

Further contextual knowledge is applied to replace the estimated dimensions of an object (w,h,l) with the *a priori* known dimensions taken either from the facet modeller (in the case of vehicles) or hard- coded into the classification system (for aircraft, vehicle and equipment). Using the preset dimensions of the objects allows accurate 3D information even when the observability is poor over all cameras (e.g. the aircraft).

Figure 17 shows the aircraft arrival tracking result for sequence S3. The aircraft is associated globally over all the cameras (i.e. regardless of validation regions). The recovered track (blue) is offset from the centre of the apron due to the false positive detection of the aircraft shadow (which is cast to the left of the image). The aircraft dimensions are correctly output from the system even with poor observability of the aircraft object, due to the use of prior knowledge. More work is required to improve the robustness of the contextual global rules situations where, for example, more than two aircraft are identified. This could be achieved by re- incorporating the validation gate

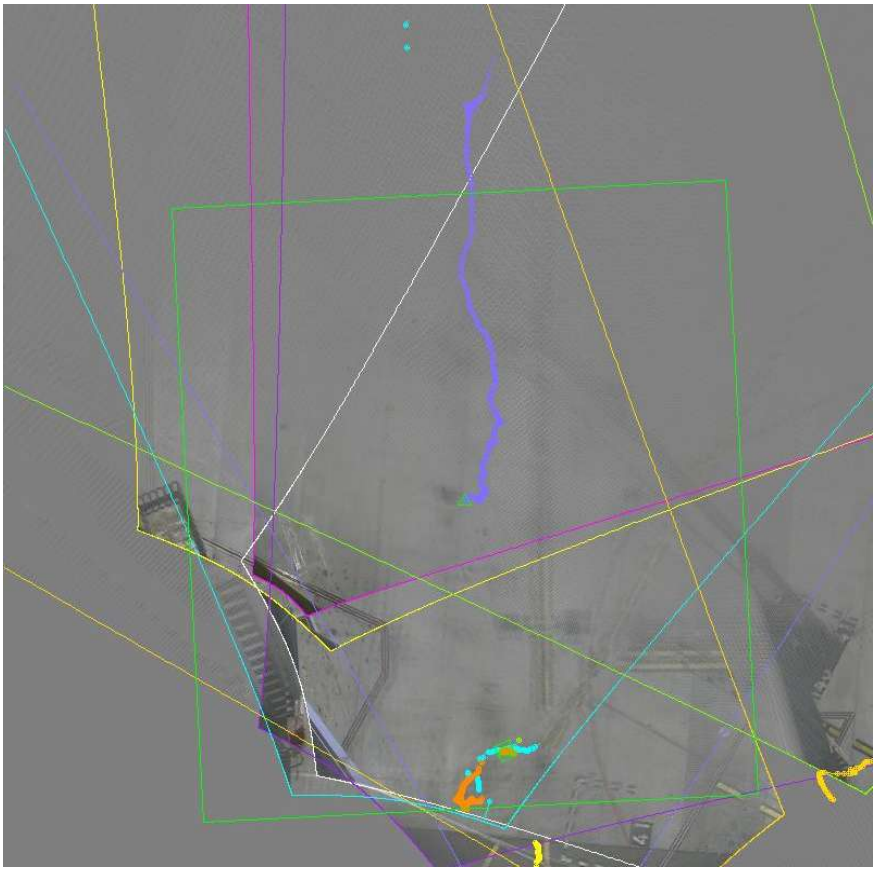


Figure 17: S3 aircraft tracking result shown on the apron surface. The blue track represents the track of the aircraft as it arrives on the apron (using global data association) and the green box represents the orientation and the size of the aircraft. The size of the aircraft is defined using prior knowledge of the dimensions of, in this case, an Airbus A320.

6. REFERENCES

[1] "AVITRACK Technical Annex 1", AST3- CT- 3002- 502818.

[2] "Motion Detection Report – Deliverable D3.1", Version 1.0, DL_AVI_2_006, The University of Reading, 22 Sept 2004.

[2] "Scene Tracking Report – Deliverable D3.4A", Version 2.0, DL_AVI_2_011, The University of Reading, September 2005.

[4] "Data Fusion Report – Deliverable D3.2A", Version 2.0, DL_AVI_2_014, The University of Reading, September 2005.

[5] "Object Categorisation / Recognition Report – Deliverable D3.3, DL_AVI_2_015, The University of Reading, September 2005.

[6] "Prototype Scene Tracking Evaluation – Deliverable D6.1A", Version 1.0, DL_AVI_6.1_10, PRIP, 11 Dec 2004.

[7] Horprasert T., Harwood D., and Davis L., "A Statistical approach for real- time robust background subtraction and shadow detection". In Proc. IEEE ICCV FRAME- RATE Workshop, Greece, Sept 1999.

[8] Shi J., and Tomasi C., “Good Features to Track”. In IEEE Conf. CVPR, pp. 593-600, 1999.

[9] Ruiz-del-Solar J., and Vallejos P. A., “Motion Detection and Tracking for an AIBO Robot using Camera Motion Compensation and Kalman Filtering”. *RoboCup* 2004: 619-627

[10] Collins R., Lipton A., Kanade T., Fujiyoshi H., Duggins D., Tsin Y., Tolliver D., Enomoto N., Hasegawa O., Burt P., and Wixson L., “A System for Video Surveillance and Monitoring”. In *Technical Report CMU-RI-TR-00-12*, May 2002.

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

[12] “Appearance 3D Model Utility – Internal Technical Note”, Version 1.0, IN_AVI_2_017, University of Reading, 27 Oct 2005.

[13] “Numerical Recipes in C”, 2nd edition, Cambridge University Press.

[14] Bishop, J.M., “Stochastic Searching Networks”. In Proc. 1st IEE Conf. on Artificial Neural Networks, pp. 329-331, London, 1998.

[15] Y. Bar-Shalom and X. R. Li, *“Multitarget-Multisensor Tracking: Principles and Techniques”*. YBS Publishing, 1995.

[16] J. Black and T.J. Ellis,``Multi Camera Image Measurement and Correspondence.'' *Measurement - Journal of the International Measurement Confederation* 35(1) July, pp 61--71, 2002.