

An Integrated Vision System for Aircraft Activity Monitoring

Mark Borg, David Thirde, James Ferryman
Computational Vision Group
The University of Reading
Whiteknights, Reading, RG6 6AY, UK

Florent Fusier, François Brémond, Monique Thonnat
ORION Team
INRIA
SOPHIA-ANTIPOLIS, FRANCE

Abstract

This paper presents work in progress on automatic scene interpretation of airport aprons based on a multi-camera video surveillance system. The Scene Tracking and Scene Understanding modules are described and preliminary results and evaluation are presented.

1 Introduction

This paper introduces work in progress on the EU project AVITRACK [5]. 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*). Figure 1 shows a birds-eye view simulating the expected positions of various vehicles that are integral to the task of servicing the aircraft. A combination of visual surveillance algorithms in a multi-camera environment are applied to track semantically meaningful objects and recognise activities predefined by a set of servicing operations.

The architecture employed is a decentralised eight camera tracking system with overlapping fields of view (FOV) [3]; the cameras are marked by blue triangles in Figure 1 with each having a stationary FOV. Each camera agent performs per frame detection and tracking of scene objects, and the output data is transmitted to a central server where data association and fused object tracking is performed. This tracking result is subsequently fed to a video event recognition module where spatial and temporal events relating to the servicing of the aircraft are detected and analysed.

1.1 Scene Tracking and Understanding

The application of signal processing theory to detect and track objects in dynamic scenes is a well researched problem; the application of such methods to robust apron activity analysis presents new challenges that must be overcome. For example, the system must be capable of monitoring and recognizing the activities and interaction of numerous vehicles and personnel in a dynamic environment over an extended period of time. Another constraint is that it must operate in real-time (at 12.5 FPS with resolution

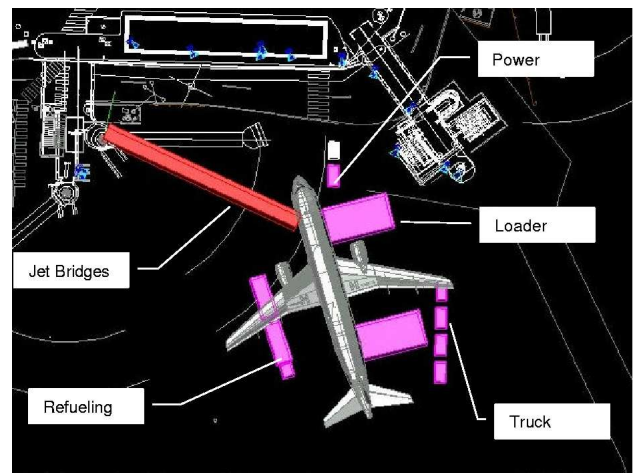


Figure 1: The distribution of equipment around a parked A320 aircraft in apron echo-40 at Toulouse Airport.

720 × 576). The tracking of moving objects on the apron has previously been performed using a top-down model based approach [15] although such methods are generally computationally expensive and are less generic than bottom-up approaches. Video event recognition algorithms analyse tracking results both spatially and temporally to automatically recognise the activities occurring in the scene; for aircraft servicing analysis these activities occur around parked aircraft in apron areas. Recent work by Xiang *et al* [14] applied a hierarchical dynamic Bayesian network to model and hence recognise scene events; however, such models are incapable of recognising simultaneous complex scene activities in real-time over extended time periods.

Section 2 details the scene tracking module comprising per-camera motion detection, bottom-up feature-based per-camera object tracking and finally fused object tracking using the combined object tracking results from the camera agents. Section 3 describes the video event recognition module including both the representation of video events and the video event recognition algorithm itself applied to apron monitoring.

2 Scene Tracking

The scene tracking module is responsible for the detection and tracking of moving objects from individual cameras; the tracked object locations are subsequently transformed into the 3D world co-ordinates. The data fusion algorithm determines single measurements in the world co-ordinates for each object from the multiple camera observations.

2.1 Motion Detection

The output of a motion detector is generally used to find foreground regions (e.g. connected components), which can then be used by the tracking algorithms to track objects of interest across multiple frames. The airport apron, being an outdoor environment, provides a number of challenges to motion detection. It must handle a wide range of environmental conditions, illumination changes and weather. The illumination changes can be long-term, such as the diurnal cycle, or short-term, caused by cloud movements, reflections, etc.

16 motion detection algorithms were implemented and qualitatively evaluated for the project. The performance criteria for the evaluation were ‘Susceptibility to Noise’, ‘Robustness to Illumination Changes’, ‘Detection Sensitivity’ and ‘Speed’ with all algorithms implemented in C++ and evaluated on dual 2.8Ghz pentium workstations with 2GB RAM running Suse Linux 9.1. Five algorithms that were found to perform adequately on a range of test data and the evaluation is presented here. The specific test sequence evaluated was ‘Airport03062004’, a frame of which is shown in Figure 2. The five algorithms evaluated were:

1. Linear prediction [8]
2. Mixture of Gaussians [9]
3. Colour and edge fusion [6]
4. Kernel density estimation [7]
5. Colour mean and variance

All these methods had acceptable susceptibility to noise, although detection noise was encountered on thin object components (e.g. the aircraft wing edge) perhaps due to aliasing or JPEG artifacts. All the algorithms were reasonably robust to illumination changes, the algorithms were modified with a shadow/highlight detection component based on the work of Horprasert *et al* [10]. The colour and edge fusion technique was found to be the most sensitive for detection in regions with low contrast; linear prediction was also found to have a good detection sensitivity, often finding moving object regions undetected by the other techniques. The most efficient algorithms with regards to processing requirements were the colour and mean variance and the Gaussian mixture model; it was found after

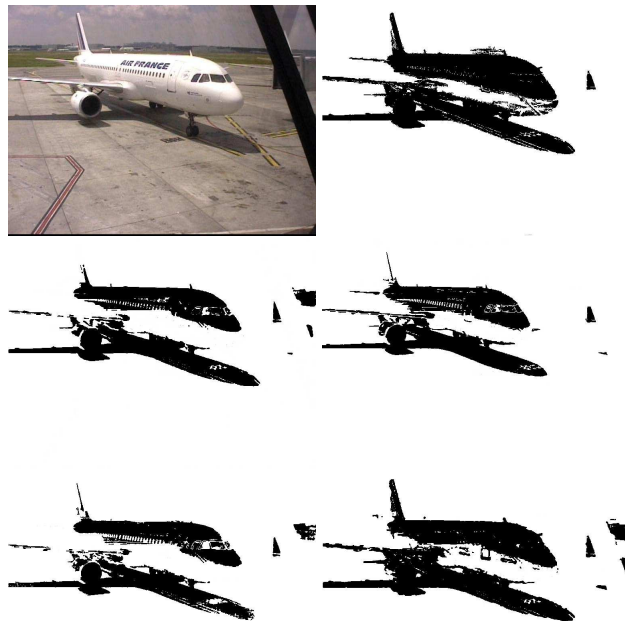


Figure 2: (Clockwise from top-left) Observed video frame and the detection results for the motion detection algorithms 1–5 (see enumerated list) for frame 1500 taken from the sequence ‘Airport03062004’, Camera 3.

this performance evaluation that the colour and edge fusion and kernel density estimation algorithms could be made significantly faster by simplifying some of the routines and assumptions made. A representative set of results from the motion detection algorithms is shown in Figure 2; interestingly, the performance of all the algorithms appears to be of a similar level with relatively minor response differences between them.

Figure 2 demonstrates the common failing of the detection algorithms tested; with false negatives detected due to the similar appearance (at pixel level) of the aircraft body and the background and false positives detected in shadowed background regions. The false negatives detected generally have negligible effect on the estimated bounding box dimensions, since the edges of moving object regions are mostly detected. The false positives (caused mainly by strong shadow in this example) present a much greater challenge since existing shadow/highlight detection methods generally rely on colour information to find such regions. The AVITRACK datasets contain predominantly achromatic regions (both moving and stationary); this results in failure for existing shadow/highlight detection methods due to unreliable colour information in such regions.

2.2 Object Tracking

Real-time object tracking can be described as a correspondence problem, and involves finding which object in an image frame relates to which object in the next frame of a video stream. Normally, the time interval between two successive frames is small, meaning that the changes visible from one frame to the other should be limited. This allows the use of temporal constraints and/or object features that can help to simplify the correspondence problem. Tracking algorithms have to deal with problems such as motion detection errors and the interactions between objects as viewed by a camera; they can appear to merge together, occlude each other, fragment, undergo rigid or non-rigid motion, etc. Three approaches to bottom-up tracking of detected objects have been applied in the AVITRACK project. These are based on the tracking of local features, colour information and difference image clusters.

The Kanade-Lucas-Tomasi (KLT) feature tracker (described in [4]) combines the Lucas-Kanade method for matching local features in adjacent frames with the Shi-Tomasi feature selection criterion. The colour based tracker uses an object's histogram as the global colour model for tracking, and is an adaptation of the CamShift algorithm [12] (which in turn is an extension of the Mean Shift algorithm). The difference image clusters algorithm is based on the work of Pece [13], and uses a probabilistic generative model for detecting and tracking objects.

The KLT algorithm considers features to be independent entities and tracks each of them individually. Therefore, it must be incorporated into a higher-level tracking process that is able to group features into objects, maintain associations between the features and the objects, and use the individual tracking results of the features to track objects from one frame to the next, taking into account interactions between objects, such as merging situations, object splitting, stationary objects, etc. For each object O being tracked, a set S of features is maintained. For a list of known objects $\{O_i\}$ at time $t - 1$, the tracking process for time t can be summarised as:

1. Generate object predictions $\{P_i\}$ for time t from the list of known objects $\{O_i\}$ at $t - 1$.
2. Run the KLT algorithm to track the features of $\{P_i\}$ individually.
3. Given a list of measurement objects $\{M_j\}$ detected by the motion detector at time t , match predictions $\{P_i\}$ to the measurement objects.
4. Any remaining unmatched predictions in $\{P_i\}$ are set as missing observations. Any remaining unmatched measurement objects in $\{M_j\}$ are considered as potential new objects.

5. Update the state of those predictions in $\{P_i\}$ that were matched to measurement objects and replace lost features. The final result is a list of tracked objects $\{O_i\}$ at time t .
6. Let $t = t + 1$ and repeat step 1.

The predicted objects are obtained from the set of tracked objects at frame $t - 1$ i.e. $\{P_i\}_t = \{O_i\}_{t-1}$. The measurement objects $\{M_j\}$ are connected regions of foreground pixels detected by the motion detector at time t . A match function is defined which returns the total number of features W of a prediction P_i that reside in the foreground region of a measurement M_j i.e.

$$f(P_i, M_j) = |\{W : W \in S_{P_i}, W \in M_j\}| \quad (1)$$

In the ideal case, given a prediction P_i , (1) should return a non-zero value for only one of the measurements that matches P_i . But objects, as viewed by a camera, can interact with each other in complex ways - for example, objects can appear to merge, an object may split into parts, be occluded, etc. A rule-based approach is adopted to handle such cases. The first rule handles ideal matches, i.e. one-to-one matches between predictions and measurements:

$$\begin{aligned} f(P_i, M_j) &> 0 \quad \text{and} \\ f(P_k, M_j) &= 0, \quad f(P_i, M_l) = 0 \quad \forall k \neq i, l \neq j \end{aligned} \quad (2)$$

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 measurement regions match with a single prediction P_i - in other words, the set of measurements is partitioned into two subsets: the subset $M1$ of measurements that match only with P_i and the subset of those that do not match with P_i :

$$\begin{aligned} f(P_i, M_j) &> 0 \quad M_j \in M1 \subseteq M, |M1| > 1 \quad \text{and} \\ f(P_k, M_j) &= 0, \quad \forall M_j \in M1, k \neq i \quad \text{and} \\ f(P_i, M_l) &= 0, \quad \forall M_l \notin M1 \end{aligned} \quad (3)$$

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

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

$$\begin{aligned} f(P_i, M_j) &> 0 \quad P_i \in P1 \subseteq P, |P1| > 1 \quad \text{and} \\ f(P_i, M_k) &= 0, \quad \forall P_i \in P1, k \neq j \quad \text{and} \\ f(P_l, M_j) &= 0, \quad \forall P_l \notin P1 \end{aligned} \quad (4)$$

In this case the state of the predictions (such as position and bounding box) cannot be obtained by a straightforward update from the measurement’s state, since only one combined (merged) measurement 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 measurement 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 measurement region that are not explained by any of the predictions.

When objects become stationary, they are integrated into the motion detector’s background model. A multi-layered model is used to handle overlapping stationary objects and allow re-activation once they start moving again. The criteria used for checking whether an object has become stationary or not was relaxed so that if only a small localised part of the object remains in motion, the whole object is integrated into the background and a new object is created. For example, a person emerging from a vehicle just as it becomes stationary. The same approach is used for the re-activation criteria. The set of rules used to match predictions to measurements was extended to include cases when stationary objects and moving objects interact with each other.

Tracking results from the local feature algorithm are shown in Figure 3. Qualitative evaluation has been performed, and the local feature tracking method was found to give the best results so far. It can suffer from loss of object identity if the local features are lost during merged or occluded states and cannot be replenished fast enough. It also requires objects to be textured in order for good features to be selected. The results of the difference image clusters method look promising, but it suffers from tracking failures when clusters merge together and it requires a higher-level process to preserve identities. The colour tracking method performs quite badly because of the achromatic nature of most of the objects present in the AVITRACK datasets.

2.3 Data Fusion

The main advantages of using a multi-camera tracking system are:

- Occlusion Minimisation - If a target becomes occluded

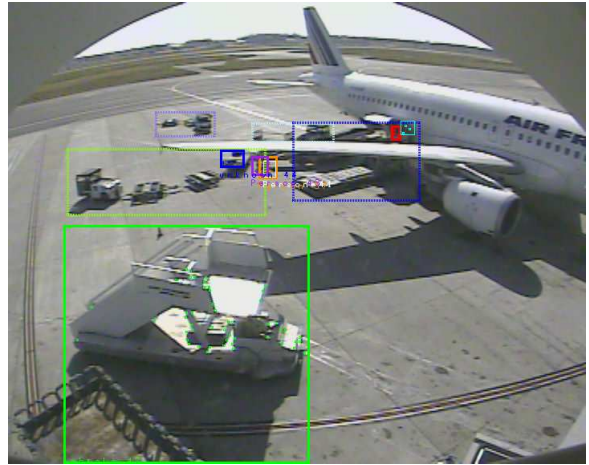


Figure 3: The results obtained from the local feature based tracking algorithm.

in a camera view, there is a higher probability of observing the target unoccluded with a different camera viewpoint.

- A larger Visible Area - This consists of the combined area (i.e. network field-of-view) observed by all the cameras from their respective viewpoints. This results in targets being potentially observed for a longer period of time over a wider area.
- Improved 3D localisation - A more accurate and reliable 3D position can be computed for targets observed by multiple cameras.

The data fusion module combines the tracking data seen by each of the individual cameras to maximise the useful information content of the scene being observed and hence achieving the above-mentioned advantages over single-camera systems. Data fusion also helps to minimise the volume of data generated by the many cameras and helps to reduce the bandwidth needed to send information to later processing modules. Spatial registration of the cameras is performed using per camera calibration and the camera streams are synchronised temporally across the network by the central server to prevent temporal drift between the image frames acquired by each camera, which will affect the fusion accuracy.

The current method for data fusion is based on a nearest neighbour Kalman filter approach [3]. A measurement uncertainty for each camera is used in the fusion of the data (approximated as $1/d_i^2$, where d_i is the distance from the observing camera i); the Kalman filter assumes a constant velocity model with a state vector $\mathbf{X}_t = [x \ y \ \dot{x} \ \dot{y}]^T$. This work is in the early stages of development; future work is envisaged to evaluate the covariance based measurement uncertainty method adopted by Ellis *et al* [11] and the JPDAF / multi-hypothesis techniques described in [3].

3 Scene Understanding

The goal of video event recognition is to perform a high level interpretation of the scene observed through video sequences by detecting video events occurring in the scene. The approach to detect video events uses cognitive vision techniques based on spatio-temporal reasoning, *a priori* knowledge of the observed environment and a set of predefined event models. The Video Event Recognition module takes as input the tracked mobile objects from a Scene Tracking module and generates as output the events which have been recognised. The *a priori* knowledge of the environment corresponds to the empty scene model. It contains the list of all the contextual objects (e.g. equipments, zones of interest, airport walls, jet-bridge) in the scene. The contextual objects are characterised by their 3D geometry (approximate shape) and their semantics (e.g. how they interact with operating people).

The set of event models is defined by the experts of the domain (e.g. managers of handling companies) using a video event description language described in [1].

3.1 Video Event Representation

The aim of video event representation is to make explicit all the knowledge necessary for the system to be able to detect events occurring in the scene. The description of this knowledge has to be declarative and intuitive (in natural terms), so that the experts of the aircraft activity monitoring in this case can easily define and modify it. Thus, the video event recognition uses mainly the knowledge represented by experts through event models.

This representation of knowledge needs to be clear, rich, intuitive and flexible. The proposed model of a video event E is composed of five components:

- a set of Physical Object variables corresponding to the physical objects involved in E (any contextual objects including static objects (equipment, zones of interest) and mobile objects (people, vehicles,...))
- a set of temporal variables corresponding to the components (sub-events) of E
- a set of forbidden variables corresponding to the components that are not allowed to occur during the detection of E
- a set of constraints (symbolic, logical, spatial and temporal constraints including Allen's interval algebra operators) involving these variables
- a set of decisions corresponding to the tasks predefined by experts that are needed to be executed when E has been detected (e.g. to launch an alarm or to display a message on a window)

There are four types of video events: primitive states, composite states, primitive events and composite events.

A state describes a situation characterising one or several physical objects defined at time t or a stable situation defined over a time interval. A primitive state (e.g. a person is inside a zone) corresponds to a vision property directly computed by the Scene Tracking module. A composite state corresponds to a combination of primitive states.

For example, Figure 4 describes the model of the composite state "Vehicle_Stopped_Inside_Zone". This state involves two physical objects : a vehicle and a zone of interest. Two components must be detected to recognise this state : state c1, the vehicle v1 is stopped (named "Vehicle_Stopped") and state c2, the vehicle is located inside a zone of interest z1 (named "Inside_Zone"). There is one temporal constraint indicating that c2 occurs during c1.

```
CompositeState(Vehicle_Stopped_Inside_Zone,
PhysicalObjects((v1 : Vehicle), (z1 : Zone))
Components((c1 : PrimitiveState Inside_Zone(v1, z1))
(c2 : PrimitiveState Vehicle_Stopped(v1)))
Constraints( (c2 during c1)))
```

Figure 4: The model of the composite state "Vehicle_Stopped_Inside_Zone" is composed of two components c1 and c2

An event is an activity containing at least a change of state values between two consecutive times (e.g. a vehicle enters a zone of interest : it is outside the zone and then inside the zone). A primitive event (as shown in Figure 5) is a change of primitive state values and a composite event is a combination of states and/or events.

```
PrimitiveEvent(Changes_Zone,
PhysicalObjects((m1 : MobileObject), (z1 : Zone), (z2 : Zone))
Components((c1 : PrimitiveState Inside_Zone(m1, z1))
(c2 : PrimitiveState Inside_Zone(m1, z2)))
Constraints( (c1 meet c2)))
```

Figure 5: The model of the primitive event "Changes_Zone" is composed of two components c1 and c2

3.2 Video Event Recognition

The video event recognition algorithm has to detect which events are occurring in a stream of mobile objects tracked by the Scene Tracking module.

The algorithm to recognise a primitive state consists in a loop of two operations : (1) selection of a set of physical objects then (2) verification of the corresponding atemporal constraints until all combinations of physical objects have been tested. Once a set of physical objects satisfies all atemporal constraints, the primitive state is said to be recognised. In order to facilitate primitive event recognition, event templates are generated for each primitive event the last component of which corresponds to this recognised primitive state. The event template contains the list of phys-

ical objects involved in the primitive state. These physical objects partially instantiate the event template.

To recognise a primitive event given the event template partially instantiated, the recognition algorithm consists in selecting (if needed) a set of physical objects matching the remaining physical object variables of the event model then looking backward in the past if a previously recognised primitive state matches the first component of the event model. If these two recognised components verify the event model constraints, the primitive event is said to be recognised. In order to facilitate composite event recognition, after each primitive event recognition, event templates are generated for all composite events the last component of which corresponds to this recognised primitive event.

The recognition of composed states and events usually implies a search in a large space composed of all the possible combinations of components and physical objects. To avoid this combinatorial explosion, all composed states and events are decomposed into states and events composed at the most of two components through a stage of compilation in a preprocessing phase. Then the recognition of composed states and events is performed similarly to the recognition of primitive events. The video event recognition algorithm is detailed in [2].

4 Video Event Recognition for Apron Monitoring

The challenge for apron monitoring was to adapt this algorithm (from earlier work on metro station monitoring) to a new type of environment and new types of physical objects. On the apron, there are not only people but also many vehicles of different types which are involved in all the handling operations. Thus the video event description language has been extended with new types of physical objects and the video event recognition module has been modified to adapt it to this new application domain.

In this domain, video event recognition is used to perform a high level interpretation of activities around parked aircrafts on an apron. The activities to recognise are mainly the apron handling operations (e.g. “catering”, “refueling”, “baggage loading”) processed by the handling companies. The automatic recognition of handling operations can represent an efficient tool for the handling companies who works on apron areas.

The automatic recognition of handling operations is also a real challenge for cognitive vision research because it addresses the recognition of complex activities involving many physical objects of different types (people, aircrafts, cars, trucks, jet bridges...) over a large space observed by a camera network (i.e. eight cameras for one apron) over an extended period of time (e.g. aircraft servicing operation is

approx. one hour).



Figure 6: Detection of the Stop of the G.P.U. Vehicle in its area

4.1 Predefined video events

At present, a first set of 5 primitive states, 4 composite states and 3 primitive events have been defined, needed for the recognition of handling operations. There are 3 states: “Inside_Zone”, “Outside_Zone”, “Stays_Inside_Zone” and 3 events: “Enters_Zone”, “Leaves_Zone”, and “Changes_Zone” concerning the localisation of a mobile object relative to a zone of interest.

There are 6 specific states describing the dynamics of a vehicle: “Vehicle_Stopped”, “Vehicle_Stopped_Inside_Zone”, “Vehicle_Arrived_In_Zone”, “Vehicle_In_Motion”, “Vehicle_Moves_At_Walking_Pace”, and “Vehicle_Exceeds_Zone_Speed”.

For video event recognition, the initial focus is on handling operations involving only one vehicle and/or a person. The main test was performed for the “Aircraft Arrival Preparation” event. This operation involves a vehicle (a Ground Power Unit, named G.P.U.) and its driver (named Handler) relative to four zones. The system has to recognise that the G.P.U. vehicle arrives and stops in the “G.P.U. Access Area” and then the driver gets out from the vehicle and deposits the chocks and the stud at the location where the plane will stop (illustrated in Figure 6 and described in Figure 7).

The following 4 composite states and 4 composite events have been defined to model the G.P.U. operation:

- composite state “Gpu_Arrived_In_ERA” (Event 1)
- composite event “Gpu_Enters_Gpu_Access_Area” (Event 2)
- composite state “Gpu_Exceeds_ERA_Speed” (Event 3)
- composite state “Gpu_Stopped_In_Gpu_Access_Area” (Event 4)

- composite state “Handler_Gets_Out_Gpu” (Event 5)
- composite event “Handler_Deposits_Chocks_Or_Stud” (Event 6)
- composite event “Handler_From_Gpu_Deposits_Chocks_Or_Stud” (Event 7)
- composite event “Aircraft_Arrival_Preparation” (Event 8) : the full operation involving the G.P.U.

The operation “Aircraft Arrival Preparation” (as shown in Figure 7) is recognised when the 5 video events involving the G.P.U. vehicle have been recognised and the constraints verified.

```
CompositeEvent(Aircraft_Arrival_Preparation,
PhysicalObjects( (p1 : Person), (v1 : Vehicle), (z1 : Zone),
(z2 : Zone), (z3 : Zone), (z4 : Zone))
Components( (c1 : CompositeState Gpu_Arrived_In_ERA(v1, z1))
(c2 : CompositeEvent Gpu_Enters_Gpu_Access_Area(v1, z2))
(c3 : CompositeState Gpu_Stopped_In_Gpu_Access_Area(v1, z2))
(c4 : CompositeState Handler_Gets_Out_Gpu(p1, v1, z2, z3))
(c5 : CompositeEvent Handler_From_Gpu_Deposits_
Chocks_Or_Stud(p1, v1, z2, z3, z4))
Constraints( (v1->Type = "GPU")
(z1->Name = "ERA")
(z2->Name = "GPU_Access")
(z3->Name = "GPU_Door")
(z4->Name = "Arrival_Preparation")
(c1 before c2)
(c2 before c3)
(c3 before c4)
(c4 before c5)
(c4 during c3)
(c5 during c3)))
```

Figure 7: The model of the composite event “Aircraft_Arrival_Preparation” contains 6 physical objects, 5 components and 11 constraints

The video event corresponding to the “Refueling Operation” has also been modeled. The system has to recognise that the Tanker arrives and stops in the “Refueling Area” and then the driver gets out from the vehicle and refuels the aircraft. The video event recognition module has been tested on a first stage of the “Refueling Operation” corresponding to the part when the Tanker is getting ready to refuel the aircraft (as described in Figure 8).

To model the Tanker arrival, 3 composite states and 2 composite events are defined:

- composite state “Tanker_Arrived_In_ERA” (Event 9)
- composite event “Tanker_Enters_Refueling_Area” (Event 10)
- composite state “Tanker_Exceeds_ERA_Speed” (Event 11)
- composite state “Tanker_Stopped_In_Refueling_Area” (Event 12)
- composite event “Tanker_Arrival” (Event 13) : the arrival of the Tanker before performing the refueling operation

```
CompositeEvent(Tanker_Arrival,
PhysicalObjects( (v1 : Vehicle), (z1 : Zone), (z2 : Zone))
Components( (c1 : CompositeState Tanker_Arrived_In_ERA(v1, z1))
(c2 : CompositeEvent Tanker_Enters_Refueling_Area(v1, z2))
(c3 : CompositeState Tanker_Stopped_In_Refueling_Area(v1, z2))
Constraints( (v1->Type = "Tanker")
(z1->Name = "ERA")
(z2->Name = "Refuelling_Area")
(c1 before c2)
(c2 before c3)))
```

Figure 8: The model of the composite event “Tanker_Arrival” contains 3 physical objects, 3 components and 5 constraints

The “Tanker_Arrival” event (shown in Figure 8) is recognised when the 3 events involving the Tanker vehicle have been recognised and the constraints verified.

The plan is to start with basic operations and to progressively move to more complex situations involving more people and vehicles.

4.2 Results

This section deals with the evaluation of the Scene Understanding module. This was first evaluated using sequences for which the Scene Tracking module gives good results.

The video events involving a G.P.U. have been tested on a dataset of 4 scenes corresponding to 8 video sequences (containing from 1899 to 3774 frames and including one night sequence) showing the “Aircraft Arrival Preparation” on the same apron and one scene showing the “Tanker Arrival”. Eight cameras observe the same scene with different fields of view. The video event recognition module has been tested on the two best points of view from where the G.P.U. can be observed and on the only point of view from where the Tanker can be observed.

The evaluation is at present mainly qualitative and performed manually with no ground truth. The aim is to get an idea of the performance of the Scene Understanding module and to anticipate potential problems in event detection for apron monitoring. All video events are recognised correctly (45 TPs) with very few false alarms (3 FPs) and no miss detection (0 FNs). These results are very encouraging but one has to keep in mind that situations where the Scene Tracking module miss detects or over detects mobile objects are not addressed.

Events 5, 6, 7 and 8 are only detectable on one of the two tested cameras because the zone of interest is just observable by one camera. The FPs of event 5 are due to a too vague modeling of event 5. This event is detected when the handler (driver of the G.P.U. vehicle) exits from the vehicle in a predefined zone near the door of the vehicle (called “Gpu_Door”). This event is detected incorrectly when a person walks in this zone and is analysed by the system as exiting the G.P.U. vehicle.

Event	Sequences	TP	FP	FN
G.P.U.				
Event 1	4 scenes * 2 cam.	8	0	0
Event 2	4 scenes * 2 cam.	8	0	0
Event 3	4 scenes. * 2 cam.	8	0	0
Event 4	4 scenes. * 2 cam.	8	0	0
Event 5	2 scenes * 1 cam.	2	3	0
Event 6	2 scenes * 1 cam.	2	0	0
Event 7	2 scenes * 1 cam.	2	0	0
Event 8	2 scenes * 1 cam	2	0	0
Tanker				
Event 9	1 scene * 1 cam.	1	0	0
Event 10	1 scene * 1 cam.	1	0	0
Event 11	1 scene * 1 cam.	1	0	0
Event 12	1 scene * 1 cam.	1	0	0
Event 13	1 scene * 1 cam.	1	0	0

Table 1: TP = “Event exists in the real world and is recognised”, FP = “Event does not exist in the real world and is recognised (over detection)”, FN = “Event exists in the real world and is not recognised (miss detection)”

4.3 Discussion and Future Work

The preliminary results are encouraging for both the Scene Tracking and Understanding modules. The performance of single-view object tracking provides adequate results; however, tracking is sensitive to significant dynamic and static object occlusion within the scene. Future work will address data fusion (single- and multi-view), effective shadow detection and suppression, and explicit occlusion analysis. The Scene Understanding results prove that the proposed approach can be adapted to apron monitoring. The main difficulty in using the video event recognition module for apron monitoring is first to model the handling operations using expert knowledge (25 video events) and second to add new possibilities in the video event description language, such as declaring new types of physical objects. Future work will consider the recognition of more complex operations involving more people and vehicles. Currently, video event recognition has only been evaluated using inputs from the Scene Tracking module where all physical objects are completely tracked. The next step is to work on uncertainty to enable recognition of events even when the Scene Tracking module loses physical objects or gives unreliable output.

Acknowledgement

This work was supported by the European Union, grant AVITRACK (AST3-CT-3002-502818)¹.

¹However, this paper does not necessarily represent the opinion of the European Community, and the European Community is not responsible for any use which may be made of its contents.

References

- [1] F. Brémond, N. Maillot, M. Thonnat and V. VU. “Ontologies for Video Events.” *Research report number 51895*, INRIA Sophia-Antipolis, Nov 2003.
- [2] V. VU, F. Brémond, M. Thonnat, “Automatic Video Interpretation: A Novel Algorithm for Temporal Event Recognition.” *IJCAI’03*, Acapulco, Mexico, 9-15 Aug 2003.
- [3] Y. Bar-Shalom and X.R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, YBS, 1995.
- [4] J. Shi and C. Tomasi, “Good Features to Track.” *In Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp 593–600, 1994.
- [5] P. Blauensteiner and M. Kampel, “Visual Surveillance of an Airports Apron - An Overview of the AVITRACK Project.” *In Digital Imaging in Media and Education, Annual Workshop of AAPR*, 2004.
- [6] S. Jabri, Z. Duric, H. Wechsler and A. Rosenfeld, “Detection and Location of People in Video Images Using Adaptive Fusion of Color and Edge Information.” *In Proc. International Conference on Pattern Recognition*, pp 4627–4631, 2000.
- [7] A. Elgammal, D. Harwood and L. Davis, “Non-parametric model for background subtraction.” *In IEEE ICCV’99 FRAME-RATE WORKSHOP*, 1999.
- [8] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: Principles and practice of background maintenance.” *In Proc. International Conference on Computer Vision*, pp 255–261, 1999.
- [9] C. Stauffer and W.E.L. Grimson, “Adaptive Background Mixture Models for Real-Time Tracking.” *In Proc. International Conference on Pattern Recognition*, pp 246–252, 1999.
- [10] 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*, 1999.
- [11] T. Ellis, J. Black, M. Xu and D. Makris, “A Distributed multiple camera surveillance system.” *In Ambient Intelligence*, P.Remagnino, G.L.Foresti and T.Ellis, Editors. Kluwer Academic Publishers, 2004.
- [12] G. Bradski, “Computer Vision Face Tracking For Use in a Perceptual User Interface.” *Intel Technology Journal*, Q2, 1998.
- [13] E.C. Pece, “From Cluster Tracking to People Counting.” *In IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, pp 9–17, 2002.
- [14] T. Xiang and S. Gong, “On the Structure of Dynamic Bayesian Networks for Complex Scene Modelling.” *In Proc. Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pp 17–22, Nice, France, October, 2003.
- [15] Sullivan, G. D. “Visual Interpretation of known objects in constrained scenes” *Phil. Trans. R. Soc. Lon.* , B, 337, pp 361–370, 1992.