7th AIAA Aviation Technology, Integration & Operations (ATIO) Conference
2nd Centre of Excellence for Aircraft Technologies (CEIAT)
International Conference on Innovation & Integration in Aerospace Sciences
18-20 September 2007, Belfast, Northern Ireland, UK
DOI:10.2514/6.2007-7738

AIAA 2007-7738

A Traffic Surveillance Function and Conflict Detection Method for Runway Manoeuvres

Andrew Sammut[*] and Brian Zammit[*]
Faculty of Engineering, University of Malta, Msida, MSD06, Malta

David Zammit-Mangion[†]
School of Engineering, Cranfield University, Bedfordshire, MK43 0AL, England
Faculty of Engineering, University of Malta, Msida, MSD06, Malta

Runway conflicts continue to occur regularly in commercial aviation. There have been several initiatives worldwide to implement new systems capable of detecting such conflicts. For example, ground based systems have been implemented in several airports improving the air traffic controller situational awareness. Unfortunately, however, these systems fail to provide support in situations where the time to conflict is very short. This paper discusses the implementation of an algorithm for traffic surveillance and runway conflict detection on board the aircraft to provide the necessary information directly to the crew. The rules on which the detection algorithm is based and the alert suppression logic to reduce the number of nuisance alerts are discussed. A description of the multi-threaded software architecture is also included.

Nomenclature and Abbreviations

ADS-B	=	Automatic Dependant Surveillance - Broadcast
ARINC	=	Aeronautical Radio Inc.
ATC	=	Air traffic control
ATCO	=	Air traffic control officer
EASA	=	European Aviation Safety Agency
ECEF	=	Earth-centred, Earth-fixed
FMS	=	Flight management system
HMI	=	Human-machine interface
MASPS	=	Minimum Aviation System Performance Standards
NTSB	=	National Transportation Safety Board
RTCA	=	RTCA Inc.

I. Introduction

Runway conflicts constitute a major hazard in commercial aviation. Statistics indicate that over 1300 conflicts have occurred in the United States alone over the period 2001-2004[1]. This constitutes an occurrence of approximately once a day. Considering the potential implications of a runway conflict, this results in an unacceptable risk of accident and it is not surprising that the NTSB has listed runway incursions on its most wanted list in 2006[2].

[*] Research Engineer, Department of Electronic Systems Engineering, Student Member, AIAA.
[†] Lecturer, Department of Aerospace Engineering, Cranfield University and Lecturer, Department of Electronic Systems Engineering, University of Malta, Senior Member, AIAA.

The risk of runway collision is traditionally mitigated by rigorous procedure, which is normally adhered to in operation through good airmanship and ATC practices. However, procedures are prone to failure. This is particularly so in unfavourable operating conditions such as lack of familiarity of airfield, weather, operational pressures and low visibility. The investigation of several runway conflicts indeed indicates that at least one of the said conditions is usually a contributing factor in runway conflicts and collisions. Consequently, current barriers established through procedures are considered inadequate and electronic means are required to provide assistance in further mitigating the risk of collision.

One strategy that has been adopted has been the equipping of ATC with ground traffic movement surveillance tools that identify the location of ground movements and attract the controller's attention in the event of a conflict. However, the NTSB considers that such systems have been ineffective in resolving some of the recent incidents in the United States because the mechanism depends on the ATCO relaying advice to the appropriate traffic in time. This process, however, introduces a significant delay in mitigating action being taken. Such delays can be critical in situations such as where the conflict alert is generated only a few seconds before a potential collision. Accordingly, the NTSB advises that the risk of runway collisions should be mitigated through the direct warning to pilots.

The authors are in agreement with the point of view of the NTSB. Direct warnings to pilots are normally provided through airborne solutions. Accordingly, this application requires an airborne solution that provides for the surveillance of traffic in the vicinity and on the runway, conflict detection logic that determines whether a conflict (either procedural or physical) exists and an alerting function that is driven by the conflict detection logic. A major concern in alerting systems, however, is the generation of false alerts and missed alerts. Missed alerts reduce the effectiveness of the system, thereby, in this context, reducing the level of improved safety levels during runway manoeuvres. False alerts, in comparison, reduce the confidence in the safety-net alerting system. In the application, false alerts may have a much more serious effect than missed alerts, for these may lead the crew to execute a manoeuvre that is hazardous in an attempt to avoid a conflict that does not exist. For example, a false alert close to the decision speed V_1 may result in a low speed overrun and the development of an accident when the take-off could have been continued safely with the aircraft becoming airborne and continuing along its intended route.

In such circumstances, a false alarm can be considered as a system failure. System failures are addressed by regulation. Specifically, EASA CS-25 regulation 1309(b)[3] stipulates that: "*The aeroplane systems and associated components, considered separately and in relation to other systems, must be designed so that -*

(1) Any catastrophic failure condition (i) is extremely improbable; and (ii) does not result from a single failure; and
(2) Any hazardous failure condition is extremely remote; and
(3) Any major failure condition is remote.

Catastrophic failure conditions are those that lead to accidents that generally result in hull loss and multiple fatalities, whilst hazardous conditions lead to incidents in which there is a large reduction in functional capability or safety margins and may lead to serious injury. Major failure conditions are less severe. Regulation 1309 effectively defines the maximum (design target) failure rate allowed of particular failure conditions. For example, 'extremely improbable' means a probability of occurrence of less than once in 10^9 operational hours under the AC1309 definition.

In the context of false runway conflict alerting, a runway excursion could be catastrophic. The determination of the risk associated with this eventuality does not only involve the consideration of the false alert rate of the sytem, but it also involves a number of other considerations such as the probability of the alert occurring in the vicinity of the decision speed, the runway length and the crew response to the alert. Nevertheless, in such circumstances, the need for high system was considered as a fundamental requirement at all design stages. This has led the authors to focus their design philosophy around the problem of false alerts and this has led to the identification of design and implementation strategies to support the effective mitigation the risk of their occurrence.

II. Functional overview

A. Surveillance function

The traffic surveillance function essentially involves the reception of ADS-B messages transmitted by aircraft within the general area of the ownship, the tracking of these aircraft and the identification of whether any of these are within a 3 dimensional protected zone. The protected zone is a zone that completely surrounds a runway (or, under certain circumstances, part of it) and is exclusively assigned to an aircraft, in this case the ownship, during a take-off and landing manoeuvre. The protected zone is assigned to the ownship when it crosses the hold short bars leading to the active runway or it approaches it to within about 1.5nm on arrival.

B. Conflict detection

Although a procedural conflict is caused whenever a traffic movement enters a protected zone assigned to the ownship, this does not always result in a physical conflict, that is, a risk of collision. For example, the entry of an aircraft on a runway behind the ownship manoeuvring in the take-off roll constitutes a runway incursion from a procedural perspective, but poses an extremely low risk of collision. In such circumstances, it is considered inappropriate to alert the ownship of the runway incursion, as this would unnecessarily distract the crew at a critical moment of the manoeuvre, thus inadvertently increasing the risk of accident. This and other such considerations have resulted in the identification of the need to develop a robust conflict detection algorithm based on a set of rules that are functions of the kinematic states of the movements involved as well as the dynamics of the scenario. Overall, it is the quality of the rule-based logic that determines how nuisance alerts are suppressed or avoided. Consequently, the design of the conflict detection algorithm is a critical consideration in the whole design of the solution proposed by the authors.

C. Manipulation of navigational data

ADS-B message data from each traffic movement is transformed into a set of state vectors that are referenced to a set of pre-defined three-dimensional Cartesian coordinates. The geodetic GPS positions are translated into points (displacement vectors) on a Cartesian coordinate system that is referenced to the selected runway threshold. This involves a first translation into ECEF coordinates and a second translation into the runway-referenced coordinate system. A 2-dimensional, flat-earth model space is used for conflict detection, with the target's height defining whether the target lies within the protected zone or above it. This approach is possible primarily because separations are small with respect to the radius of the earth (of the order of a few nautical miles or less, rendering the assumption of a flat earth sufficiently accurate) and because vertical separations do not affect the logic that defines a runway conflict (except for movements that manoeuvre above the protected zone). The use of Cartesian coordinates proves ideal for the application and, in particular, for the definition of a conflict.

D. Initialisation

As the work carried out by the authors focuses on providing protection during runway manoeuvres (take-off and landing), the runway surveillance function is initiated when the ownship is manoeuvring on the aerodrome surface prior to departure and on approach in arrival. During initialisation, the aerodrome database is queried to retrieve runway positions (latitudes and longitudes), bearing and elevation for each runway on the airfield.

III. The Multithreaded Algorithm Approach

The unpredictable nature of traffic movements and the wide variation of traffic densities, which are a function of time of day and year as well as runway location, have a significant impact on the software design and indeed even on the design methodology. In essence, traffic surveillance is an event-driven function. The event is the detection of a traffic movement within the capture area of the surveillance function. As a result, an object-oriented approach was considered to be the best suited for the application, where a thread is instantiated for each movement that is detected by the surveillance function. The thread assigned to each movement determines whether a conflict exists and generates an alert in the event.

A procedural conflict fundamentally exists whenever a movement penetrates a protected zone that is reserved for another aircraft. Consequently, the conflict detection algorithm within the thread requires two data sets, namely those pertaining to the movement of the ownship and traffic to which the thread is assigned. The ownship data can be derived directly from the FMS (FMS navigational information), which contains position and velocity data. On aircraft installed with an ARINC429 bus, navigation information is available at a deterministic minimum rate of 5Hz when using a BNR encoding (two's complement binary encoded)[4]. However, surrounding traffic information, which, in the current implementation, is obtained from ADS-B, is available at a slower update rate and is asynchronous with the ownship data update rates. ADS-B provides GPS-derived positional information, is limited in resolution and accuracy and suffers from low and inconsistent update rates. The requirements of the ADS-B infrastructure, according to RTCA MASPS[5], define a nominal update period of less than 1.5s at the 95th percentile. For this reason, correlating these two data sets in an algorithm requires a software design capable of handling mixed and inconsistent update rates. This is another significant reason for the adoption of the multi-threaded approach. The approach further affords the flexibility of better organizing the algorithm flow by providing the facility of multiprocessing. By sharing the processor time over several code segments within the algorithm, operations that require long processing time, or use external resources when receiving or writing data would not generate a delay in the computation of the whole algorithm. Figure 1 presents a simplified representation of the threading model adopted for conflict detection. On initialization, two threads are dispatched with the function of receiving the traffic and ownship data without being bound to a fixed data rate. A third thread is then dispatched, termed the Input Handler, with the role of monitoring the traffic data and initializing a new thread for each traffic object with its respective data and a copy of the most recent ownship data. In this way, each traffic object can compute the conflict detection algorithm autonomously and set the corresponding alert state. The Output Handler thread is responsible for collecting the alert state from each traffic object at a nominal rate of 10Hz. To cater for an improbable event of having multiple concurrent runway incursions, a prioritization scheme has been included. Multiple alerts are prioritized by considering the time to collision to each target and the most critical conflict is output to the HMI module.

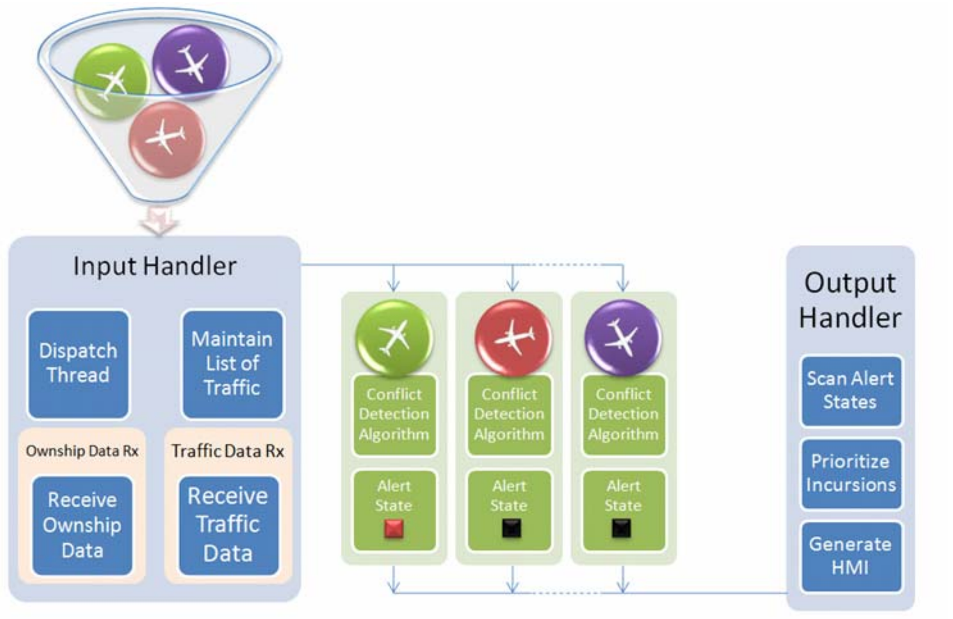

Figure 1 - The multi-threaded approach adopted in the design

IV. The conflict detection algorithm

A. Rule-based design, definition of processes and internal states

The conflict detection algorithm is based on rules that define either a procedural or a physical conflict. An analysis of the nature of runway physical and procedural conflicts suggests that conflict alerting on the ownship is a function of five broad criteria, namely:

(1) Whether a movement has entered the protected zone assigned to the ownship
(2) Whether the two parties associated in the conflict are approaching each other
(3) Whether the separation is acceptable (in some circumstances an acceptable separation is allowed)
(4) Target intent (phase of flight and expected or planned route)
(5) Ownship intent

In the present design, these criteria are defined as states, with the first three being binary in nature.

The five sub-functions that process the five criteria are, in essence, event driven. The event, in this context, is the arrival of new data pertaining to the sub-function. These events are asynchronous in nature and this favours the implementation of the sub-functions in terms of processes. Accordingly, the current implementation of the conflict detection algorithm includes five independent processes as shown in Figure 2, with each process being responsible for monitoring specific parameters that define the value of each state. This implementation is adopted from the concept described in AIAA paper 2006-6272[6] by the same authors.

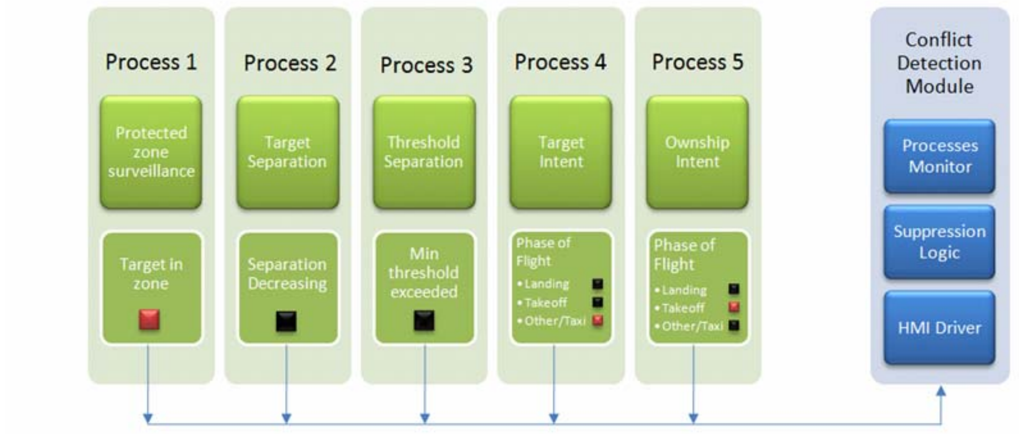

Figure 2 - Runway Conflict Detection.

The thread assigned to a specific aircraft monitors the aircraft movement and determines whether it has penetrated the protected zone assigned to the ownship by the input handler. Once a target is identified in this protected zone, all five conflict detection processes are instantiated. These processes run in parallel. Process 1, providing Protected Zone Surveillance, tracks the target (aircraft to which the process is assigned) within the selected protected zone until it vacates the protected zone. The output is set and will remain set (latched) until the target exits the active protected zone. Process 2 (Target Separation), is responsible of monitoring the separation between the ownship and target in the protected zone. The output of this process is set when the separation is detected to be decreasing. Process 3 (Threshold Separation) provides a trigger if the separation between the two parties falls below a predefined threshold. This process is useful in environment such as low-visibility backtracking, in order to provide a safety envelope around the ownship and in order to convey minimum separation information to the crew. Process 4 (Target Intent) predicts the target's phase of flight and manoeuvre intent within the protected zone. This is based on aircraft velocity, position and heading. A phase of flight is determined and provided as an output. Process 5 (Ownship Intent) monitors the ownship phase of flight and provides this as the output of the process.

The outputs of these processes are provided to the conflict detection module which, based on a set of predefined rules, verifies whether the kinematics of the scenario warrant the generation of an alert. A set of suppression rules, in line with operational procedure and regulation, are integrated in this module. These are designed to suppress the generation of an alert if such alert could result in an unnecessary go-around or aborted take-off manoeuvre and it is the rule-set within this module that fundamentally defines the quality of the algorithm in terms of false alerts and missed alerts.

B. Phase of flight detection

The detection of the phase of flight is based on the definition of three flight phases referred to as states, defined as Take-off, Landing and Other/Taxi. Initially, when no state information is available, the aircraft is assigned the Other/Taxi state by default. The aircraft will remain in this state until the conditions set in the definition of either the Take-off or Landing phase are satisfied.

For an aircraft state to switch to Take-off from the Taxi state, three conditions need to be satisfied. First, the weight-on-wheels flag is monitored to identifying whether the aircraft is manoeuvring on the runway surface. The take-off thrust setting is then checked, giving an indication of the pilot`s intention. .If the state criteria are met, the aircraft is in the protected zone and its heading coincides with that of the runway in the protected zone, the aircraft is then considered to be in the Take-off mode. In a similar way, the switch from a Taxi/Other state to a Landing state involves the satisfaction of a number of other criteria. The aircraft must be airborne and descending with a ground speed greater than a predefined minimum threshold and the aircraft heading must be in the direction of the runway heading.

Once the aircraft is assigned a Take-off or Landing state, changes of states can only occur if the predefined state end conditions are met. Indeed, for the Take-off state, the end conditions occur when the manoeuvre is aborted or after the successful completion of the take-off run. In the aborted take-off case, the retardation of the thrust levers, reduced ground speed and weight-on-wheels flag are monitored. A successful take-off terminates at the 35ft altitude. For the Landing state, the end conditions are met by the successful completion of the landing manoeuvre or on initiation of a go-around manoeuvre. Thrust lever positions, ground speed and weight-on-wheels flag are used in this respect. Figure 3 presents the state diagram associated with the described process of detecting of the phase of flight.

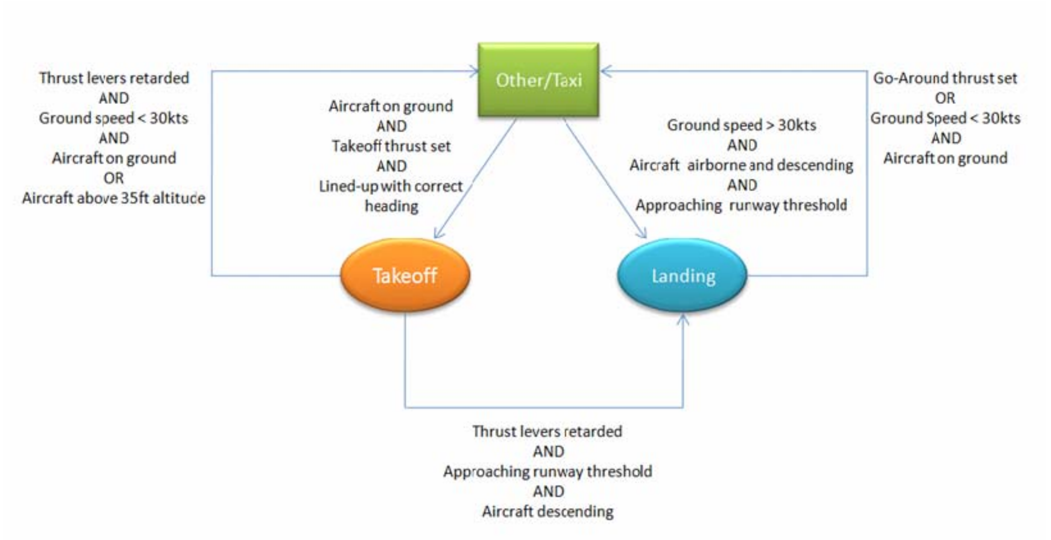
Other/Taxi
Thrust levers retarded AND Ground speed < 30kts AND Aircraft on ground OR Aircraft above 35ft altitude
Aircraft on ground AND Takeoff thrust set AND Lined-up with correct heading
Ground speed > 30kts AND Aircraft airborne and descending AND Approaching runway threshold
Go-Around thrust set OR Ground Speed < 30kts AND Aircraft on ground
Takeoff
Landing
Thrust levers retarded AND Approaching runway threshold AND Aircraft descending

Figure 3 – The state diagram of the phase of flight monitor.

C. Alert suppression

Alert suppression within the conflict detection module is a critical development in the present design, as this module significantly improves the false alert rate. Conflict detection based on simple logic associated with the 5 states that are the outputs of the 5 processes would theoretically provide an alert at the right time and in the right context. In other words, the alert would always be correct if the logic of the five states is correct. However, practical considerations and complexity of scenario in practice may significantly alter conditions. A typical case is when two aircraft are landing in close succession. The second aircraft (assumed to be the ownship) may be given a late landing clearance or even a land-after instruction. If the exact intent of the aircraft ahead (target aircraft) can be determined and the exact nature of the clearance given by ATC is known, then the automation of clearly defined rules would easily define whether the second aircraft is too close or not. This criterion would then be used to generate the alert. However, in current technology, these two sets of information cannot be assured to be transmitted to the ownship within the very short time constraints associated with the manoeuvre. This has led the authors to the concept of alert suppression, which is a rule-based function that determines whether an alert based only on the logic states generated by the five processes is to be allowed or suppressed. In other words, the alert flag is first set and then the alert is either suppressed or allowed, according to the suppression logic rules. In this scenario example, the target's velocity, position and heading are used to determine whether it is vacating the runway, in which case, the alert may be suppressed.

This implementation effectively results in either the alert being delayed or else suppressed completely in the event the conflict detection logic output is reset before the alert suppression is released. As alert suppression may result in the delay in the generation of an alert, this may influence the missed alert rate. In the above scenario, for example, the extended suppression of an alert may result in the alert being eventually triggered too late to avoid an accident. Consequently, alert suppression logic is essentially overridden by physical conflict consideration to ensure that an alert is generated in a timely manner. In this case, a potential procedural conflict would have been tolerated until the risk of collision outweighs the benefits of suppression.

In this way, alert suppression contributes to the reduction of the risk of the alert distracting or confusing the crew and therefore increases the probability of the alert correctly leading the crew to take action that is appropriate to the correct mitigation of the conflict.

D. Functional state diagrams

The functional state diagrams for the conflict detection and alerting functions during the take-off and landing manoeuvres are presented in Figures 4 and 5. Due to the fact that runway conflicts do not occur regularly, crew are usually inexperienced in dealing with such situations and they may expect to be exposed to such scenarios only during simulator training sessions. The effects of such implications are further aggravated by the limited time available to react to such situations, which may be as low as a few seconds. Consequently, it is possible that simply informing the crew of a separation loss could prove inadequate in providing a sufficient level of conflict awareness to the crew. As a result, provision is made in the design to support different alert messages during different segments of the manoeuvre. Currently, the take-off is segmented into the acceleration and deceleration phases, whilst the landing manoeuvre is segmented in the airborne and ground (deceleration) phases.

Figure 4 illustrates the states the function may enter on detecting a runway incursion during take-off. When a conflict is detected by the Conflict Detection module, the function enters the Conflict state. The suppression logic is triggered by the conflict state and this will determine whether suppression of the alert is appropriate. If the alert is not suppressed, the alert is generated and will continue until the ownship is clear of the conflict or the crew abort the take-off. The initiation of the rejection is detected through the monitoring of thrust lever position, brake application and deceleration. Once the run is aborted, the function enters the deceleration state, which can support the generation of a different alert if this is found to be appropriate. The deceleration state terminates when the ownship speed is reduced to taxi speed, thus allowing the function to terminate and the phase of flight monitor enters the Taxi/Other state.

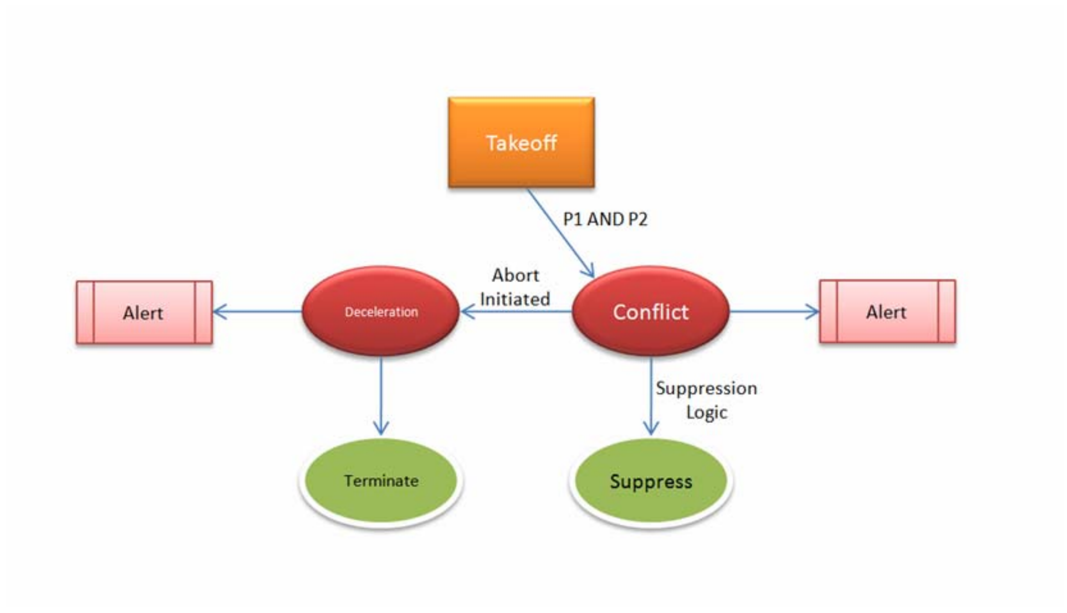

Figure 4 - State diagram for take-off incursion alerting.

Similarly during landing, the system will enter a conflict state when a conflict is identified by the Conflict Detection Module (for example, when a target is detected within the protected zone and the relative separation is decreasing, defined by the outputs of Processes 1 and 2). A runway incursion during landing can occur either when the ownship is still airborne during final approach or when the ownship is in the ground run after touchdown. Consequently, the function will enter one of two states, depending on the phase of the manoeuvre. Suppression of the alert is only possible when the ownship is still airborne, whilst the function can terminate in either phase, either when the conflict is cleared or when the manoeuvre is terminated. On such an event, the flight monitor enters the Taxi/Other state.

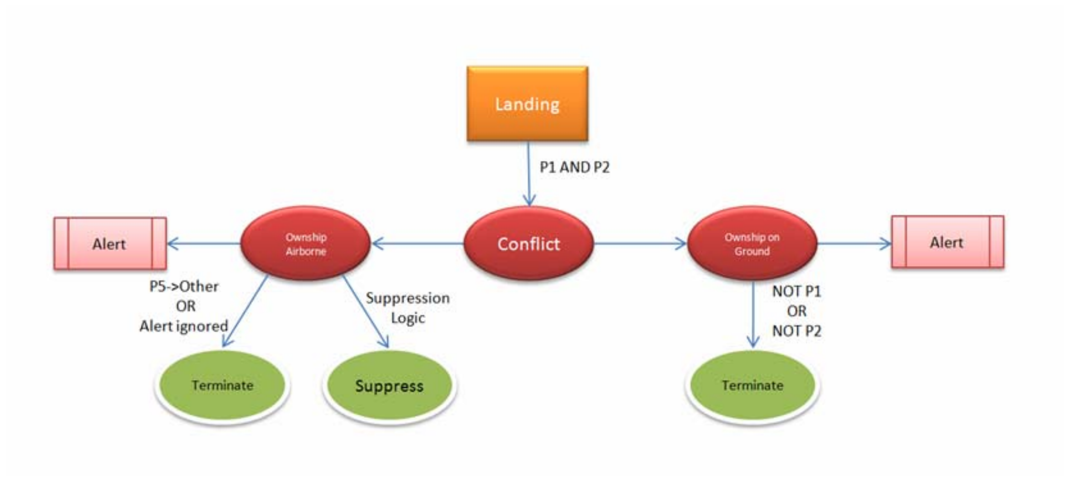

Figure 5 - State diagram for landing incursion alerting.

A conflict state during runway backtracking or line-up prior to take-off can also be triggered if an aircraft inside the protected zone approaches the ownship. In this case, the flight monitor will be in the Taxi/Other state (Figure 6). Considering the case where the ownship is stationary, lined up for take-off, once a conflict is detected, the function enters the Ownship Stationary state and will generate one class of alert that may be different from that when the ownship is moving (which is nominally associated with a taxi manoeuvre). The function will terminate when the conflict is resolved.

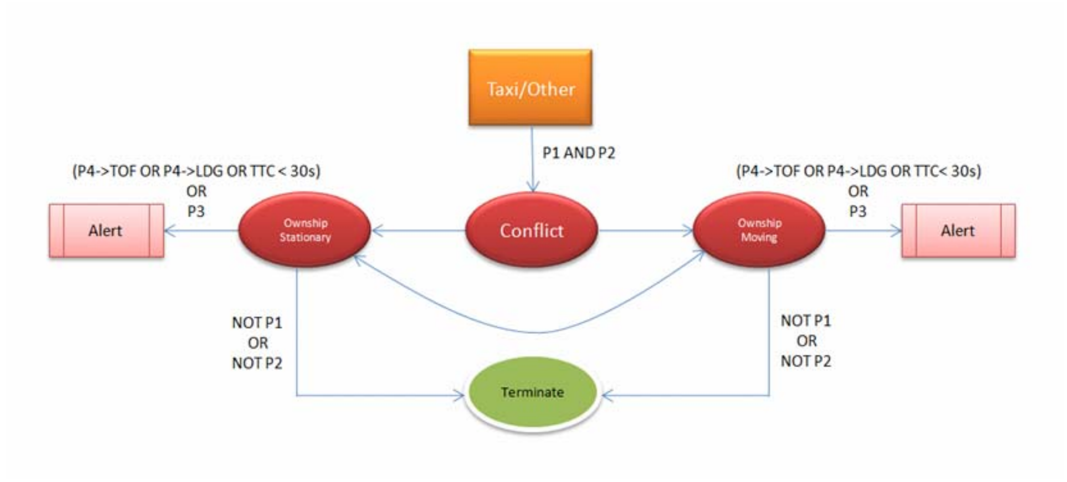

Figure 6 - State diagram for backtracking/line-up incursion alerting.

V. Testing

The described traffic surveillance function and conflict detection method has been coded and tested for functionality in a bench simulation environment as well as Cranfield University's Large Flight Simulator, an immersive, six degree of freedom simulator representative of a large transport aircraft. Bench testing involved the verification of the code segments and as well as the capability of the implementation in handling multiple targets simultaneously. To this effect, up to 50 traffic movements have been simulated, under which conditions the implementation has been found to function correctly and complete all calculations within the 10Hz update rate at which the function is nominally running. Higher numbers of traffic movements were not considered, since it is highly unlikely that such numbers would be approaching a protected zone at any one time. Traffic filters can effectively eliminate any movements that are distant from the protected zone from the input handler, thus ensuring the input handler will never need to handle an unnecessarily large number of movements simultaneously.

Testing on Cranfield's simulator involved the setting up of a number of scenarios, flying these scenarios and determining whether the alerts were correctly generated (at the right moment) or correctly suppressed. The scenarios considered are listed in Table 1. On testing, the implementation behaved in the manner expected and generated alerts at moments that are acceptable on the flight deck and allow sufficient time for manoeuvre to avoid the conflict.

Ownship State	Incursion	Alert generated
Lined up (stationary)	Target enters ahead	No
Lined up (stationary)	Target approaches from behind	Yes
Take-off	Target lined up ahead	On thrust setting
Take-off	Target enter protected zone ahead of ownship, at various ownship speeds below V_1	As soon as target enters protected zone
Take-off	Target enters protected zone behind ownship	No
Landing	Target lined up on runway	Ownship approx 1nm from touchdown (depending on approach speed)
Landing	Aircraft aborts take-off, far down the runway	Ownship approaching threshold
Landing	Aircraft landing ahead fails to vacate runway	Ownship approaching threshold
Landing	Target enters runway	As soon as target enters protected zone
Landed	Target enters runway	As soon as target enters protected zone

Table 1 – Simulator test scenarios.

VI. Conclusion

In continuation on the concepts presented in AIAA paper 2006-6272, the work presented in this paper has focused on the implementation aspects of monitoring an active runway, detecting a runway collision hazard and generating an alert. An object oriented approach was adopted, making use of threads assigned to each traffic movement and processes to determine states on which conflict logic (and suppression rules) are applied. Vector representation of conflict dynamics was used in the implementation of the traffic surveillance and conflict detection functionality. This approach and implementation were tested and found to be appropriate approaches to the subject task.

References

[1] FAA Runway Safety Report, Federal Aviation Administration, Washington, DC, August 2005.
[2] NTSB Most Wanted Transportation Safety Improvements – 2007. National Transporation Safety Board, November 2006.
[3] Certification Specifications for Large Aeroplanes – CS25, European Aviation Safety Agency, 2003.
[4] ARINC, *Mark 33 Digital Information Transfer System (DITS), Part 1, Functional description, electrical Interface, label assignments and Word formats*, ARINC 429 part 1-17, 17th May 2004
[5] RTCA, *ADS-B Minimum Aviation System Performance Standards*, DO242A, 2000.
[6] Zammit-Mangion, D., Sammut, A. and Zammit, B, “A Traffic Movement Monitoring Computer for Pre-emptive Runway Conflict Detection”, AIAA paper 2006-6272, American Institute of Aeronautics and Astronautics, August 2006.