SmartATMS : A SIMULATOR FOR AIR TRAFFIC MANAGEMENT SYSTEMS

Tak-Kuen John Koo
Yi Ma
George J. Pappas
Claire Tomlin

Robotics and Intelligent Machines Laboratory
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, CA 94720
koo,mayi,gpappas,clairet@eecs.berkeley.edu

ABSTRACT

Air Traffic Management Systems (ATMS) of the future will feature Free Flight, in which aircraft choose their own routes, altitude, and speed, and automated conflict resolution methods in which aircraft will coordinate to resolve conflicts. The resulting distributed control architecture is a *hybrid system*, with mixed discrete event and continuous time dynamics. SmartATMS is an object oriented modeling and simulation facility which accounts for these hybrid issues and will serve as a uniform modeling framework for the design and evaluation of various ATMS concepts.

1 INTRODUCTION

Air transportation systems are faced with soaring demands for air travel. The current Air Traffic Management System (ATMS) will not be able to efficiently handle this increase because of inefficient airspace utilization, increased Air Traffic Control (ATC) workload, and obsolete technology. In view of the above problems and in an effort to meet the challenges of the next century, the aviation community is working towards an innovative concept called *Free Flight* (RTCA 1995). Free Flight allows pilots to choose their own routes, altitude and speed and essentially gives each aircraft the freedom to self-optimize.

The economic benefits of Free Flight are immediate. Free Flight is potentially feasible because of enabling technologies such as Global Positioning Systems (GPS), datalink communications, Automatic Dependence Surveillance-Broadcast (ADS-B), Traffic Alert and Collision Avoidance Systems (TCAS) and powerful on-board computation. The above technological advances will also enable air traffic controllers to accommodate future air traffic growth by restructuring ATMS towards a more decentralized architecture. In future ATMS, aircraft which are in conflict will coordinate among each other and possibly with ATC in order to predict and resolve conflicts. This distributed control architecture is modeled using *hybrid systems* in which discrete event systems model the coordination among aircraft and differential equations model the aircraft dynamics and control. In addition, current aircraft operate in various *flight modes*, in which each flight mode may correspond to different objectives (such as regulating altitude, airspeed) or different phases of operation (such as takeoff, cruise or landing). Flight modes are another source of hybridness for aircraft. *The modeling and simulation of hybrid systems in the ATMS domain is an issue which requires immediate attention.*

The existing ATMS modeling tools and simulators have functionality which spans the modeling of runway and airport capacity and operations, through airspace operations and conflict resolution, to human factors and man-machine integration. A detailed study of 27 models is presented in (Odoni 1996), which categorizes the functionality of each model and assesses its strengths and weaknesses. However, the above simulators do not properly address the hybrid issues of the problem.

In addition, Free Flight has triggered an abundance of design concepts and there is a need to evaluate the proposed designs, concepts and methodologies. The comparative evaluation of many different designs requires a uniform and formal modeling framework for design specifications and performance metrics.

In order to address the above issues, we have begun the development of SmartATMS, a modeling and simulation facility for Air Traffic Management Systems in the context of hybrid systems. SmartATMS is vital in accurately simulating various conflict resolution methods in Free Flight which is a crucial piece of future ATMS. In addition, SmartATMS will be object oriented which will allow the uniform modeling of current and various proposed methodologies and will serve as our main tool for design and evaluation of future ATMS concepts.

There are essentially two simulation facilities for hybrid systems: SHIFT (Deshpande et al. 1996), developed by the PATH project at U.C. Berkeley, and OMOLA/OMSIM (Andersson 1994,1995) developed at Lund University, Sweden. SHIFT, in addition to simulating hybrid systems, allows the modeling of layered control architectures, coordination of distributed control agents and object oriented simulation. SmartATMS is being developed using SHIFT and thus inherits all of its advantages. The object oriented structure of SHIFT allows SmartATMS users to create their own future ATMS by composing objects from available libraries, and to compare it with the current system or other future alternatives. Similar research efforts have been done in the Automated Highway Systems domain (Deshpande et al. 1995) .

The organization of this paper is as follows. In Section 2, we briefly describe our proposed architecture of a future ATMS, in Section 3 we describe the use of simulation in the design and evaluation of future ATMS systems, and in Section 4 we briefly describe SmartPlanes which is an animation program for SmartATMS. Section 5 discusses issues for further research.

2 NEXT GENERATION ATMS

In this section we briefly describe a decentralized architecture for Air Traffic Management Systems (ATMS) which has been proposed in (Sastry et al. 1995),(Tomlin 1997). Each airport resides within a Terminal Radar Approach and Control (TRACON) region whose size is typically 50 nautical miles in radius and 11,000 feet in altitude. Due to the heavy congestion within TRACONs, aircraft will travel along predetermined, optimal approach and departure routes. Entry and exit to the TRACON will be

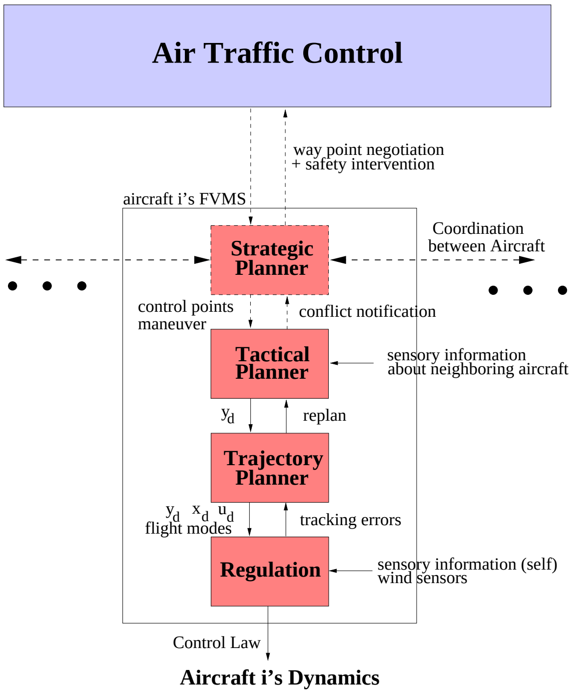

Figure 1: Proposed FVMS Architecture

done at metering gates. From the originating TRACON to the destination TRACON, aircraft will be allowed to enjoy Free Flight and will fly optimal trajectories which may also avoid hazardous weather regions. Outside TRACONs, aircraft are under the supervision of Centers and there are no predetermined routes. Conflicting aircraft will coordinate among each other in order to resolve the conflict.

The smart aircraft of the future are modeled by the hierarchical architecture presented in Figure 1. The levels of architecture below ATC reside on the individual aircraft and comprise what is known as the aircraft's *Flight Vehicle Management System*, or FVMS. The FVMS consists of four layers, the strategic, tactical, and trajectory planners, and the regulation layer. Each layer of this architecture is described in the following sections. We begin with a discussion of the airspace structure.

Airspace Structure: Current nominal trajectories through the airspace are defined in terms of waypoints, which are fixed points in the airspace defined by VOR (Visual Omni Range) points on the ground. The VOR waypoints, which determine the rigid jetways in the sky, are a necessary navigation tool for aircraft which are not equipped with the more sophisticated GPS.

Air Traffic Control: Air Traffic Control (ATC) stands for the human controllers in the TRACON or Center airspace. ATC has more control over aircraft in the TRACON than in Center airspace. In TRACONs, ATC passes a sequence of waypoints to the strategic planner on board the aircraft which will sequence an approaching aircraft from the entry gate of the TRACON to the landing approach. These waypoints are a discretization of trajectories, which have been calculated off-line for different combinations of aircraft kinematics, wind magnitude and direction, and runway configurations. The Center-TRACON Automation System (CTAS), developed by NASA (Erzberger 1992) , is a tool which provides such a calculation process. Once these waypoints have been negotiated they are passed to the strategic planner, and all of the planning and control tasks are taken over by the FVMS on board the individual aircraft. In Center airspace, the FVMS is allowed to choose its own self-optimal trajectory and the Center controllers are focused on maximizing airspace utilization and throughput.

Strategic Planner: The Strategic planner lives on board the FVMS of each aircraft. The main objectives of the strategic planner are to design a coarse, self-optimal trajectory and to resolve conflicts between aircraft in Center Airspace. The trajectory may be optimized with respect to fuel, flight time, and exploitation of favorable winds. The trajectory is then stored in the form of a sequence of four dimensional control points, c_k. In case of a potential conflict, the strategic planners of all aircraft involved in the potential conflict determine a sequence of maneuvers which will result in conflict-free trajectories, either using communication with each other through satellite datalink, or by calculating safe trajectories assuming the worst possible actions of the other aircraft. Each strategic planner then commands its own tactical planner to follow these maneuvers.

Tactical Planner: The tactical planner refines the strategic plan by interpolating the control points with a smooth output trajectory, denoted by y_d in Figure 1. The tactical planner uses a kinematic model of the aircraft for all trajectory calculations. The output trajectories of the kinematic model are then passed to the Trajectory Planner as desired output profiles.

Trajectory Planner: The trajectory planner uses a detailed dynamic model of the aircraft, sensory input about the wind's magnitude and direction, and the tactical plan consisting of an output trajectory, to design a full state and nominal input trajectory for the aircraft, and the sequence of *flight modes* necessary to execute the dynamic plan. These flight modes represent different modes of operation of the aircraft and they correspond to controlling different variables in the aircraft dynamics. The resulting trajectory, denoted y_d, x_d, and u_d in Figure 1, is given to the regulation layer which directly controls the aircraft.

Regulation Layer: Once a feasible dynamic trajectory has been determined, the regulation layer is asked to track it. In the presence of large external disturbances (such as wind shear or malfunctions), however, tracking can severely deteriorate. The regulation layer has access to sensory information about the actual state of the aircraft dynamics, and can calculate tracking errors. These errors are passed back to the trajectory planner, to facilitate replanning if necessary.

3 SmartATMS: A SIMULATION TOOL FOR ATMS

Large scale systems, such as ATMS, require the use of simulation during the design and evaluation stage since most analytic methods are intractable if not impossible to solve. SmartATMS, a modeling and simulation facility for ATMS, models ATMS as a dynamic network of interacting hybrid systems. In the following sections we describe the benchmarks which will be used for performance evaluation, the components which comprise the ATMS system, the language used to develop SmartATMS, and the problems involved in simulating hybrid systems.

3.1 Performance Benchmarks

In order to have a performance comparison between the current and future ATMS, any simulation tool should be able to model proposed system architectures as well as facilitate the modeling of the existing architecture.

An ATMS is designed for deployment in an environment specified in terms of performance benchmark scenarios, which are given in terms of:

Airspace Configuration The Airspace Configuration defines the airspace structure including Center airspace, TRACON airspace, airports, and VOR waypoints.

Travel Demand The Travel Demand specifies the flight trip differentiated by aircraft type, takeoff time, landing time, route, flight time, and origin and destination.

Weather Condition The Weather Condition specifies the weather scenarios that can affect the capabilities of the system.

Abnormal Events The Abnormal Events specify instances of faults and malfunctions, collisions and other rare events.

The performance of an ATMS design under the specified benchmark scenarios is judged by a set of performance metrics. Different categories of performance metrics are required to compare different designs, and those are:

Safety - Probability of collisions and guarantees of safe operation under fault free conditions

Efficiency - Fuel consumption, flight times and delays, operating costs

Comfort - Maximum translational and rotational acceleration of an aircraft

Capacity - Maximum number of aircraft takeoffs and landings that can be supported by an airfield

Environmental Impact - Air and noise pollution

Costs - Implementation and infrastructure costs

Human Factors - Effects of human-in-the-loop

For performance evaluation, the simulation tool should facilitate the access of internal states and conditions of all the components in the system for generating different performance metrics.

3.2 Functional Decomposition of ATMS

In this section, we present a system description framework for modeling the ATMS domain. The architecture of the proposed ATMS system can be decomposed into five types, which are Airspace, Aircraft, Air Traffic Control, Weather and Performance Monitors:

Airspace Airspace models the structure of airspace, and also divides the sky into different regions. Organized by geographical area, Airspace consists of three different categories, which are Center Airspace, TRACON Airspace, and SUA (Special Use Airspace). Inside TRACON Airspace, there are VOR points and Airports with Runways. Center airspace may also consist of VOR points (in current ATMS) or may be completely unstructured (in Free Flight). Within Airspace resides Geography which provides the geographic features such as surface models close to airports. Airports, which reside within TRACONs, may be modeled at various levels of detail, including surface movement, boarding gates etc.

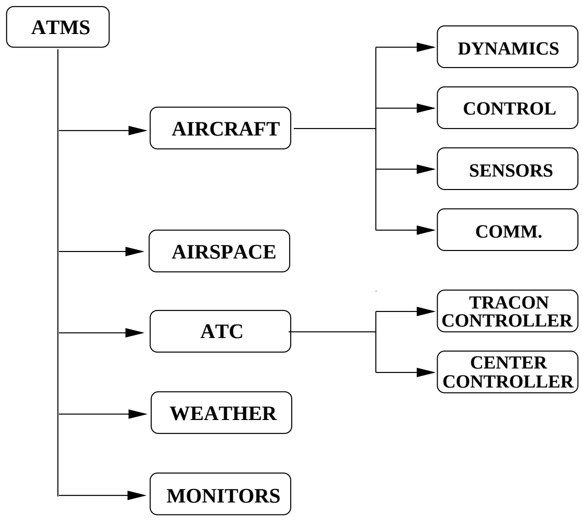

Figure 2: Hierarchical Functional ATMS Layout

Aircraft Aircraft models the behavior of each aircraft. Each aircraft consists of four types, which are Dynamics, Controller, Sensor, and Communication. Dynamics describes the dynamic behavior of an aircraft according to the effects from the controller and the environment. Controller models the FVMS but may also model human pilot commands. Sensor provides navigation, surveillance and other state information which is crucial for control purposes. Communication is for the information exchange between ATC and each aircraft, and among aircraft.

Air Traffic Control Air Traffic Control models the human controllers at TRACONs or Centers and/or the procedures they follow in order to maintain safe separation between aircraft while guiding them to their destinations. ATC is supported by Centers and TRACONs. For both Centers and TRACONs, there are controllers for operation, sensors and communication facilities for information exchange between aircraft, and other Centers and TRACONs.

Weather Weather provides information about current wind profiles, visibility conditions, current or developing storms and possible icy runway conditions.

Performance Monitor Monitor gives the performance measure of the system in six different categories, which are Safety, Efficiency, Comfort, Capacity, Environmental Impact, Cost and Human Factors. Monitors have full access to system states and state trajectories for generating performance metrics.

Figure 2 displays a partial functional hierarchical layout of an Air Traffic Management System. Each component of the architecture will be modeled at various level of abstraction. This will allow the users to create ATMS at different levels of complexity depending on the scenario of interest and the concept that needs to be tested and evaluated. For example, one user could compose a macro-model using many simple aircraft models inside a TRACON in order to simulate traffic close to an airport and evaluate capacity. Another user could compose a micro-model of two very detailed aircraft models in order to evaluate the conflict resolution protocol.

3.3 SHIFT: The Language of SmartATMS

SmartATMS models ATMS as a dynamic network of interacting hybrid systems. A hybrid system is a hierarchical and parallel combination of multiple hybrid automata, which can be created, interconnected and destroyed as the system evolves. Hybrid automata consist of standard finite state machines in which continuous dynamics reside within each discrete state. These automata collaborate or interfere with each other through their input/output interfaces or the synchronization rules.

SHIFT is an object oriented programming language for describing dynamic networks of hybrid automata, and supports the simulation of dynamically reconfigurable hybrid systems. The concept of hybrid automata can be naturally mapped to a SHIFT structure called *type*. Each hybrid automaton is an instantiation, or so called *component* in SHIFT, of a specific *type*. The *type* is a special class which characterizes the structures and properties of hybrid automata: *discrete states*, *flows*, *transitions*, and *synchronization rules*, etc. SHIFT provides methods for input/output connections among different components as well as synchronization rules for actions or transitions in different components. In the *setup* phase, each component can setup its input/output connections with other components by using the *connect* command. Each component can also *export* its transition between discrete states as *events* so as to synchronize with transitions in other components.

3.4 SHIFT Implementation of SmartATMS

In SHIFT, the object oriented approach is used to construct a logical model of the physical components and their control agents. The objects in the logical model have semantic content corresponding to their characteristics, inter-relationships, constraints, and behaviors. Each object can be created, interconnected and destroyed as the system evolves. The object oriented approach simplifies system implementation, and provides user with flexibility in creating his/her own future ATMS, by composing objects from component libraries, and performing comparative analysis on the current system or other future alternatives.

For illustration on the architecture of SmartATMS, we provide an example of the construction of a simplified aircraft model. The aircraft model includes four basic components: *Dynamics, Control, Sensors, Communication*. *Dynamics* consists of sets of differential equations which model the aircraft dynamics. *Control* is a hybrid automaton which provides the aircraft with appropriate control inputs according to different flight modes. *Sensors* provides the aircraft with navigation information, such as its position, speed, attitude, etc. *Communication* models the communication among other aircrafts and Air Traffic Control. The aircraft and each of its subsystems can then be described as a *type* in SHIFT. The aircraft is described as *type Aircraft*. Based on the simple principle, one can create an ATMS by constructing *type Airspace*, *ATC*, *Weather* and *Monitors*. The following is the content of *Aircraft*.

```
type Aircraft{

state  //Subsystems and continuous states of
       //an aircraft object;
  Aircraft_Dynamics      dynamics;
  Aircraft_Control       control;
  Aircraft_Sensors       sensors;
  Aircraft_Communication communication;

discrete  //Discrete states of an aircraft object;
  taking-off, cruising, landing;

setup  //Create subsystems for an aircraft object;
 do
 {
  dynamics     := create(Aircraft_Dynamics, ...);
  control      := create(Aircraft_Control, ...);
  sensors      := create(Aircraft_Sensors, ...);
  communication:= create(Aircraft_Communication, ...);
 };

connect  //Setup Input/Output connections;
 {
  dynamics(inputs_control)
                   <- control(outputs_dynamics);
  control(inputs_sensors)
                   <- sensors(outputs_control);
  control(inputs_communication)
                   <- communication(outputs_control);
  sensors(inputs_dynamics)
                   <- dynamics(outputs_sensors);
  ... ...
 };
}
```

```
type Aircraft_Model{

  input  //Control inputs for the aircraft;
      array(continuous number) inputs_control;
  output  //Outputs of the dynamics;
      array(continuous number) outputs_sensors;
  state  //Aircraft states;
      continuous number x, y, z, theta, psi, phi;
  discrete  //Flight modes:
      takeoff, cruise, landing;
  flow  //Aircraft dynamic equations;
      default { ... };
}

type Aircraft_Control{

  input  //Input information from other subsystems;
      array(continuous number) inputs_sensors;
      array(continuous number) inputs_communication;
  output //Controls for the aircraft;
      array(continuous number) outputs_dynamics;
  state
      array(continuous number) controls;
  discrete
      takeoff, cruise, landing;
  ... ...
}

type Aircraft_Sensors{

  input
      array{continuous number} inputs_dynamics;
  output
      array{continuous number} outputs_control;
  export  \\Exported events;
      alert;
  ... ...
}

type Aircraft_Communication{

  input
      array{continuous number} inputs_communication;
  output
      array{continuous number} outputs_communication;
  export  \\Exported synchronizing events;
      success_intruder,
      failure_intruder,
  ... ...
}
```

3.5 Simulation of Hybrid Systems

Hybrid systems, including ATMS models, possess various properties which pose significant difficulties in their simulation. Some undesirable characteristics of hybrid systems which require immediate attention include:

Noncontinuous dependence on initial conditions Hybrid systems, due to the existence of switching decision logic are extremely sensitive to initial conditions. Very small changes in initial conditions may result in completely different system trajectories. This inherent nonrobustness of hybrid systems is amplified during simulation when the continuous piece of the system trajectory is numerically approximated. This could potentially result in inconsistent system evolutions where the the actual system may evolve in one direction but the simulation evolves in another.

Event Detection A typical hybrid system evolves in time under a given differential equation until some synchronization or failure event is triggered, at which point the system switches to another mode and possibly another dynamic model. The proper detection of events which cause the transition is very crucial since many integration schemes could potentially skip the instance at which the event is occurred. Reducing the time step of the integration may be safe for event detection but may considerably slow down the simulation given the large scale of the problem.

Multiple time scales Large scale systems are usually composed of various subsystems which operate at different time scales. This presents problems in the integration of these systems and further complicates event detection schemes.

Current hybrid system simulators do not adequately address these issues. Research along these directions is needed to ensure that simulators of hybrid systems are producing system evolutions which are consistent with the evolutions of the actual system.

4 SmartPlanes : A VISUALIZATION TOOL FOR SmartATMS

The complexity of large scale projects, such as the proposed Air Traffic Management System, renders simulation a valuable tool both in the design of various control laws and coordination protocols as well as in the evaluation of overall system performance. A good simulation package may also be used as a debugging tool in the design process. The complexity of the system also emphasizes the need for efficient computational schemes, such as parallel computation algorithms. Visualization techniques, such as animation, can be used to present the simulation results in a manner which is much easier to analyze by the designer. In this direction, we have started the development of SmartPlanes, a simulation and visualization facility for ATMS on an Silicon Graphics platform. At the current stage SmartPlanes, which is shown in Figure 3 is a visualization tool which allows the

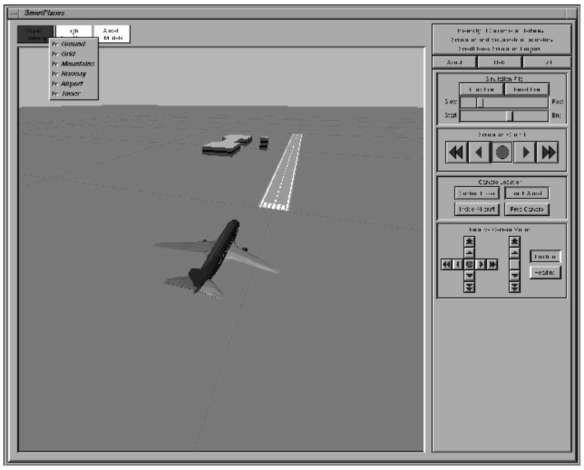

Figure 3: SmartPlanes Visualization

user to view the trajectory of a aircraft from various perspectives. For example, the user has a choice to view the aircraft from the control tower, from a fixed location, or to have the pilot's perspective from the cockpit. In future versions, multiple aircraft will be shown as well as a local radar and furthermore the user will have the ability to configure his/her own airport so as to meet the needs of different cities, e.g., Denver Airport, JFK International Airport, etc.

5 CONCLUSIONS

In this paper, we have introduced SmartATMS, a unified modeling and simulation framework for Air Traffic Management Systems. SmartATMS pays particular attention to the hybrid nature of ATMS and will be used for ATMS concept design and performance evaluation under various benchmark scenarios. Its object oriented nature allows users to easily create their own version of ATMS and compare it with current or alternative future concepts.

We are currently implementing our proposed ATMS architecture, described in Section 2, with emphasis on conflict resolution schemes for Free Flight. The more general issue of accurate and consistent simulation of hybrid systems is also under investigation.

ACKNOWLEDGEMENTS

This work was supported by the Army Research Office under grants DAAH 04-95-1-0588 and DAAH 04-96-1-0341 and by NASA under grant NAG 2-1039. The authors would like to thank Prof. S. Sastry for his invaluable advise and encouragement, Dr. A. Gollu for stimulating discussion, and the SHIFT development team of Californai PATH, the University of California at Berkeley, for their constant support.

REFERENCES

M. Andersson. *Object-Oriented Modeling and Simulation of Hybrid Systems*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden, 1994.

M. Andersson. Omola and Omsim tutorial and user's manual. Technical Report ISRN LUTFD2/TFRT-7504-SE, Lund Institute of Technology, Lund, Sweden, 1995.

A. Deshpande, D. Godbole, A. Gollu, and P. Varaiya. Design and evaluation tools for automated highway systems. In R. Alur, T.A. Henzinger, and E.D. Sontag, editors, *Hybrid Systems III*, Lecture Notes in Computer Science 1066. Springer Verlag, New York, 1995.

Akash Deshpande, Aleks Gollu, and Luigi Semenzato. The SHIFT Programming Language and Run-time System for Dynamic Networks of Hybrid Automata. Technical report, University of California at Berkeley, 1996.

Akash Deshpande, Aleks Gollu, and Luigi Semenzato. SHIFT reference manual. Technical Report PRCS project SHIC version 2.17, University of California at Berkeley, 1996.

Heinz Erzberger. CTAS : Computer intelligence for air traffic control in the terminal area. Technical report, NASA TM-103959, Ames Research Center, Moffett Field, CA, July 1992.

RTCA Inc. Final Report of RTCA Task Force 3: Free Flight Implementation. Technical report, Washington, DC, October 1995.

Amedeo Odoni. Existing and required modeling capabilities for evaluating atm systems and concepts. Technical report, DRAFT Report for NASA AATT program, 1996.

S. Sastry, G. Meyer, C. Tomlin, J. Lygeros, D. Godbole, and G. Pappas. Hybrid control in air traffic management systems. In *Proceedings of the 1995 IEEE Conference in Decision and Control*, pages 1478–1483, New Orleans, LA, December 1995.

C. Tomlin, G. Pappas, J. Lygeros, D. Godbole, and S. Sastry. A next generation architecture for air traffic management systems. Technical Report UCB/ERL M97/7, University of California, Berkeley, 1997.

AUTHOR BIOGRAPHIES

TAK-KUEN JOHN KOO received the B.Eng. degree in 1992 in Electronic Engineering and the M.Phil. in 1994 in Information Engineering both from the Chinese University of Hong Kong. From 1994 to 1995, he was a graduate student in Signal and Image Processing Institute at the University of Southern California. He is currently a Ph.D. student in Electrical Engineering and Computer Sciences at the University of California at Berkeley. His research interests include nonlinear control theory, hybrid systems, inertial navigation systems with applications to unmanned aerial vehicles.

YI MA received the B.Eng. degree in Automatic Control and the B.Sc. degree in Mathematics both from Tsinghua University, China, in 1995. He obtained a M.Sc. in Electrical Engineering and Computer Sciences in 1997 at the University of California at Berkeley, where he is now pursuing his Ph.D. degree in computer vision and robotics.

GEORGE J. PAPPAS received the B.Sc. degree in 1991 and the M.Sc. degree in 1992 both in Computer and Systems Engineering from Renseelaer Polytechnic Institute, Troy, NY. He is currently a Ph.D student in Electrical Engineering and Computer Science at the University of California at Berkeley. His research interests include nonlinear control systems, geometric control theory, hybrid systems and large scale systems with applications to air traffic management systems.

CLAIRE TOMLIN received the B.A.Sc. degree in Electrical Engineering from the University of Waterloo in 1992, the M.Sc. degree in Control Systems from Imperial College, London, in 1993, and is currently completing her Ph.D. degree in Electrical Engineering at the University of California, Berkeley. Her research interests are in hybrid and nonlinear control systems with application to air traffic management. She works jointly with NASA Ames, in Moffett Field, CA. She is a student member of IEEE.