REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 28-03-2007	2. REPORT TYPE Final Performance Report	3. DATES COVERED *(From - To)* May 2006 - September 2006

4. TITLE AND SUBTITLE System Level Architecture of Cooperative Satellite Control Using Hierarchical Hybrid Systems and Automata Theory	5a. CONTRACT NUMBER
	5b. GRANT NUMBER FA9550-06-1-0293
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Won, Chang-Hee	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Temple University 1801 N. Broad Street, Philadelphia, PA 19122-6003	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) USAF, AFRL, AF Office of Scientific Research, 875 N. Randolph St. Room 3112, Arlington, VA 22203 OFC of Naval RSCH, Chicago Regional Office, 230 South Dearborn, Room 380, Chicago, IL 60605-1595 Dr Scott Wells	10. SPONSOR/MONITOR'S ACRONYM(S) AFOSR/ONRRO
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited

AFRL-SR-AR-TR-07-0127

13. SUPPLEMENTARY NOTES

14. ABSTRACT

We investigated a hierarchical hybrid system architecture to implement an autonomous multiple satellite control scheme for the low earth orbit remote sensing satellites. We networked multiple satellites and ground control station into a hierarchical hybrid system and applied hybrid automata theory onto the different tiers of the system. Compared with the traditional satellite control concept, the hierarchical hybrid system is more autonomous, effective, scalable, and robust for the space remote sensing applications. In this paper, we developed the mathematical model for the multiple satellite control model and simulated the model with Matlab, Simulink and Stateflow. The simulation results show that the hierarchical hybrid system theory is feasible for the autonomous control of multiple satellite.

15. SUBJECT TERMS
Hierarchical hybrid system, autonomous satellite control, orbit and attitude control

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE		30	Chang-Hee Won
					19b. TELEPHONE NUMBER *(Include area code)* (215) 204-6158

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Final Performance Report

System Level Architecture of Cooperative Satellite Control Using Hierarchical Hybrid Systems and Automata Theory

Grant Award Number FA9550-06-1-0293

Submitted to Air Force Research Laboratory

Chang-Hee Won
Department of Electrical and Computer Engineering
Temple University

Abstract

We investigated a hierarchical hybrid system architecture to implement an autonomous multiple satellite control scheme for the low earth orbit remote sensing satellites. We networked multiple satellites and ground control station into a hierarchical hybrid system and applied hybrid automata theory onto the different tiers of the system. Compared with the traditional satellite control concept, the hierarchical hybrid system is more autonomous, effective, scalable, and robust for the space remote sensing applications. In this report, we developed the mathematical model for the multiple satellite control model and simulated the model with Matlab, Simulink and Stateflow. The simulation results show that the hierarchical hybrid system theory is feasible for the autonomous control of multiple satellite.

1. Introduction

Using multiple small and low-cost satellite to perform a mission instead of a large satellite has a number of advantages. Small satellites are more cost-effective to build and launch—they can piggyback on large payload launch. Adding more small satellites to the system is easier from development time and economic perspective. The system would be more robust, because even if a small satellite fails, the mission would still go on with rest of the satellites. All of these properties are predicated, however, on the fact that autonomous control of multiple satellites is possible. Controlling a large number of satellites from a ground control station would require many resources. Moreover, autonomous multiple satellite system would react in a more time-effective manner. Therefore, we study the control paradigm necessary to autonomous control multiple satellites.

Although the developed method is not necessarily restricted the remote sensing application, in this report, we will concentrate on the remote sensing applications with low-earth-orbit (LEO) satellites. The remote sensing applications include environmental monitoring, natural disaster detecting, and battlefield surveillance. Therefore, how to acquire information from the target area more timely, more accurately, more effectively with less cost are important issues when we design a LEO satellite remote sensing system.

In the traditional LEO satellite based remote sensing applications, satellites are controlled by the ground control stations. The satellites receive the mission commands from ground control station when they are visible from the ground control station, they execute the mission. After performing the mission, the data acquired by the satellites will be sent to the ground control station in the subsequent pass. The traditional control of the satellites is implemented via communication links between ground control station and the satellites. There are several communication architectures, which are described in [1], dealing with the ground control station and satellites such as Store and Forward architecture, Geostationary Crosslink Communication system, and Low-Altitude Multiple Satellites with Crosslinks. All of these traditional space system architectures have some disadvantages when they are applied to the low-cost space applications. These systems are designed as a communication link rather than a control and communication system architecture. It is important to include communication architecture, however, that is not the only consideration when designing a multiple satellite autonomous control system. The control architecture is also important. In fact, both communication and control architecture have to be considered to design an effective multi-satellite control system.

In this report, we propose a hierarchical hybrid system architecture by connecting the LEO satellites into a network to meet the requirements of more efficient and autonomous real time remote sensing missions. Unlike the traditional system architectures mentioned above, the new system has a hierarchical architecture for the networked satellites. Hierarchical system had been used in the space applications [2], the multiple intelligent robotics control [3], and unmanned vehicles control [4, 5]. It is also a typical architecture of sensor networks [6]. Hybrid automata are used in cooperative control of a group of aerial or ground objects [7, 8] and autonomous vehicles formation maneuvers [9]. We combine the hybrid control theory into the hierarchical system because that there are not only discrete control dynamics involved in the satellite system, but also the continuous dynamics. This hierarchical hybrid architecture has several advantages in performance, autonomy, cost, scalability, and efficiency compared with the tradition system architectures.

1.1 System Architecture

Our hierarchical architecture in the satellite sensor network consists of three tiers, the top tier is the Ground Control Station, the middle tier component is called the Mother-ship, and the bottom tier is called the Agent. Figure 1 shows this hierarchical architecture.

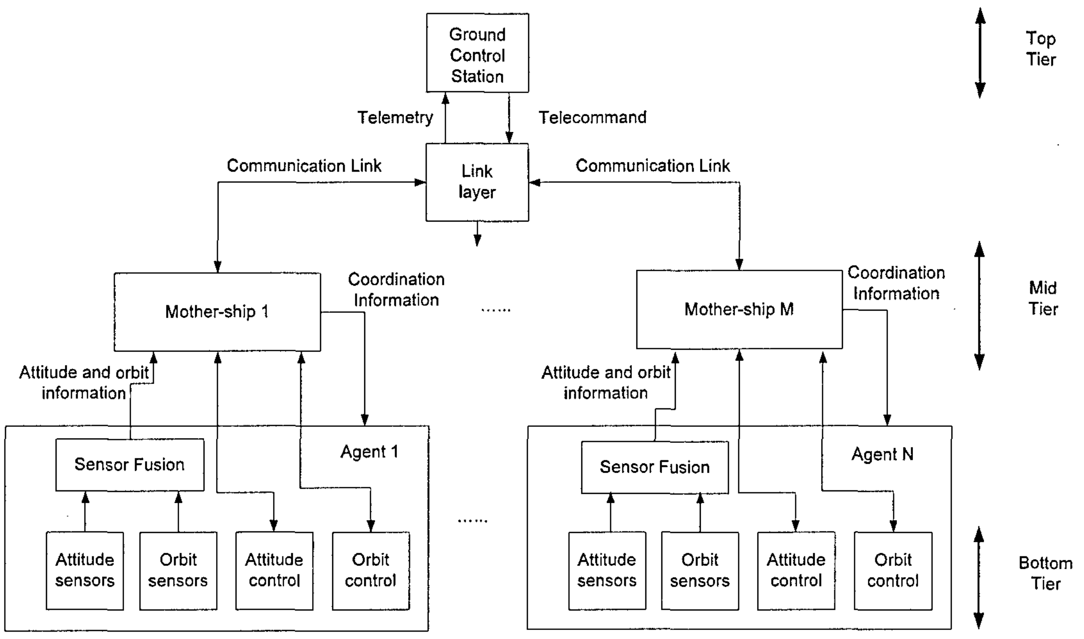

Figure 1. Hierarchical Cooperative Satellite Control System Architecture

The main goal for the Ground Control Station is to generate the mission command, monitor the status of the whole satellite network, and gather data which is obtained from the networked satellites. The Ground Control Station also performs the ranging and tracking tasks for the satellites passing through its communication radius to ensure the control accuracy of Mother-ships and Agents.

A Mother-ship is a satellite which connects the Ground Control Station and Agents. One Mother-ship can control a number of Agents in a cluster, within its communication radius. A Mother-ship receives mission command from the Ground Control Station, then analyzes the command, calculates the orbit parameters for each Agent, then generates the task commands to coordinate each Agent. Then the Mother-ship sends these coordination information to Agents at the scheduled time to control the orbit and attitude of the Agents to perform the tasks. The Mother-ship then monitors the feedback from Agents to determine whether the tasks are accomplished and whether it is necessary to reschedule the tasks for the Agents if some of the tasks are failed. Typically, a Mother-ship and the Agents within a cluster are in the same orbit in such a way that the Mother-ship can maintain a fixed relative position to its Agents. A Mother-ship can also communicate with the other Mother-ships to relay the mission command from the Ground Control Station and perform the coordinated control on Agents. When all Agents, which are controlled by a certain Mother-ship, finish the tasks, the Mother-ship will feedback the mission status to the Ground Control Station. This alerts the operator whether the mission is accomplished, partial accomplished, or failed. Then the sensor data will be collected by the Mother-ship and routed to the Ground Control Station directly or by multiple-hop routing.

An Agent is a satellite with sensors and actuators which performs the required operations for the mission specified by the Ground Control Station. The Agents are grouped into different clusters, within each cluster there is one Mother-ship acting as the cluster head which can control all Agents in this group. The Agents perform the orbit and attitude maneuver according to the task commands from Mother-ship. After performing the designated task operation, the Agent will send a feedback message and the sensed data to the Mother-ship. In case a Mother-ship malfunctions, one of the Agents can be assigned as the new Mother-ship to take over the control and coordination of the cluster. The metrics for selecting a Mother-ship from Agents depend on the physical characteristics of the Agents and system requirements.

Both the Mother-ship and Agent are LEO satellites. The difference between a Mother-ship and an Agent is that the Mother-ship has more computation and communication capacity. In addition, the Mother-ship can obtain more accurate position and orientation information than the Agents. However, if a Mother-ship fails and an Agent takes over the control of the cluster, then there is no performance difference between the new Mother-ship and other Agents.

Figure 2 shows one possible configuration of the satellite sensor network where there are one Ground Control Station and six clusters of satellites. Each cluster consists of one Mother-ship as cluster head and two Agents. We note that different clusters of satellites may operate in the different LEO orbits whereas satellites within the same cluster are in the same orbit.

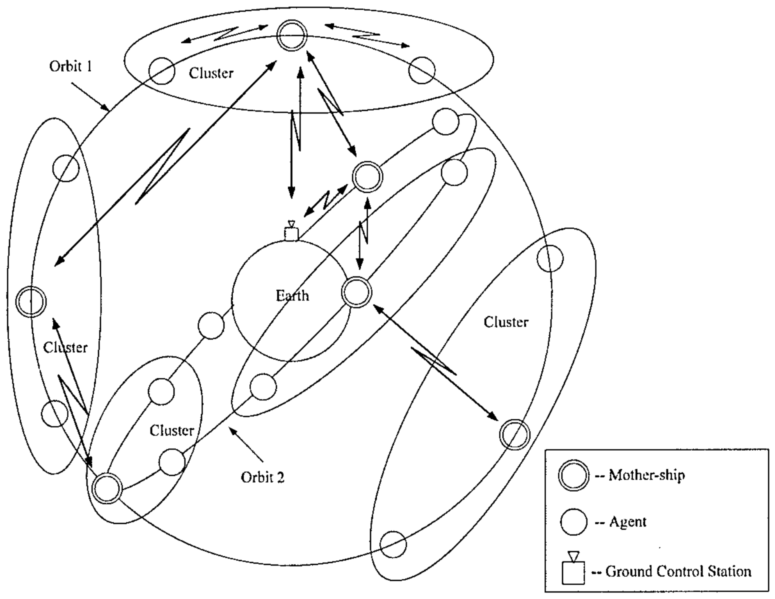

Figure 2. Satellite network system configuration

1.2 Communications Architecture

In a multiple satellite control system, communications architecture is closely related to the control because satellites are not always accessible from the Ground Control Station. Therefore, the control commands and coordination information are disseminated only by the intercommunication among the satellites. Therefore, it is necessary to design an effective architecture to transmit the control command and relay the data in the system.

Due to the difference among the functional performances of the components in different tiers, communications methods used in satellite network vary in different tiers.

In the top tier, the Ground Control Station sends the mission commands to all reachable Mother-ships. The mission commands include scheduling missions and rescheduling missions.

In the middle tier, a Mother-ship which receives the mission command from the Ground Control Station will broadcast the command to other reachable Mother-ships. The sensed data collected by the Mother-ships will also be relayed to other reachable Mother-ships until the data reach the Ground Control Station. The Mother-ship sends task commands to different Agents by the unicasting method.

In the bottom tier, all Agents within the cluster can only communicate with their cluster head, the Mother-ship. The Agents receive the task command from the Mother-ship and send the sensed data back to the Mother-ship. There is an exception to this rule; if Mother-ship malfunctions and it is unable to perform the mission, then Agents in the cluster will communicate with each other to elect a new cluster head. The Agent with the closest performance specification in terms of processing speed, orbit determination accuracy, and attitude determination accuracy will replace as the Mother-ship. If all the Agents have similar performance specifications, the Agent which is closest in the orbit will replace the failed Mother-ship.

2. *Hybrid Automata Theory*

The mathematical model that we will use for the multiple satellite control architecture is hybrid automata. In a hybrid system, both discrete and continuous dynamics exist, and this can be modeled using hybrid automata. Hybrid automata are represented using the graph theory as a state transition diagram. The transitions between each state in the diagram are the discrete transitions, while the continuous dynamics are included in each state. Certain conditions that can be viewed as discrete events constitute the transitions, causing the system to move from one state to another. In hybrid automata theory, there are several important definitions and properties which we will review in this report. We will state the definitions and theorems as given in [11] for the sake of completeness. For more details and proofs see [11, 12, 13, 14, 15].

- *Hybrid Automaton:*

A hybrid automaton H can be described mathematically as,

$H = (Q, X, f, Init, D, E, G, R)$

where

Q	Finite set of discrete variables representing the discrete dynamics of H ;
X	Finite set of continuous variables;
$f : Q \times X \to TX$	Vector field, defining the continuous flow in each discrete node;
$Init \subseteq Q \times X$	Set of initial states;
$D : Q \to P(X)$	Domain;
$E \subseteq Q \times Q$	Set of edges;
$G : E \to P(X)$	Guard condition;

$R : E \times X \to P(X)$ Reset map.

The set Q is simply the set of discrete states where the system is allowed to exist. The set X will represent all of the continuous variables that are possible in each of the states of Q. The vector field, f, usually consists of time dependent functions such as differential equations that describe how each of the variables in X changes over time. TX denotes the tangent bundle of X. The initial states are simply the initial values of the continuous variables and the domain, D, contains the range where each continuous variable is allowed to exist. The domain is also called the invariant set. $P(X)$ denotes the set of all subsets of X. The set of edges, E, describe what transitions are allowed to occur between states. The guard conditions, G, describe the events that must occur for a transition to take place. Finally, the reset map, R, is the set of conditions that cause the system to enter its initial state.

- *Hybrid Time Trajectory*

A hybrid time trajectory is a finite or infinite sequence of intervals $\tau = \{I_i\}_{i=0}^{N}$, such that

1) $I_i = [\tau_i, \tau'_i]$, for all $i < N$;

2) if $N < \infty$, then either $I_N = [\tau_N, \tau'_N]$ or $I_N = [\tau_N, \tau'_N)$; and

3) $\tau_i \le \tau'_i = \tau_{i+1}$ for all i.

In other words, a hybrid time trajectory is a sequence of intervals of the real line, whose end points overlap. The interpretation is that the end points of the intervals are the times at which discrete transitions take place [16]. We will use hybrid time trajectory to analyze time horizon of executions defined below.

- *Execution*

Like a solution to a continuous system, an execution of a hybrid automata H is a collection $\chi = (\tau, q, x)$, where τ is a hybrid time trajectory, $q : \langle\tau\rangle \to Q$ is a map, and $x = \{x^i : i \in \langle\tau\rangle\}$ is a collection of differentiable maps $x^i : I_i \to X$, such that

1) $(q^0, x^0) \in Init$;

2) for all $t \in [\tau_i, \tau'_i), \dot{x}^i(t) = f(q(i), x^i(t))$ and $x^i(t) \in D(q(i))$;

3) for all $i \in \langle\tau\rangle \setminus \{N\}$, $e = (q(i), q(i+1)) \in E$, $x^i(\tau'_i) \in G(e)$, and $x^{i+1}(\tau_{i+1}) \in R(e, x^i(\tau'_i))$.

Here, $\langle \tau \rangle$ is the set {0, 1, 2,..., N} if N is finite and {0, 1, 2,...} if $N = \infty$. In this report, (q^0, x^0) denotes the initial state. we use $\mathcal{E}_H(q^0, x^0)$ to denote the set of executions with the initial condition $(q^0, x^0) \in Init$, $\mathcal{E}_H^M$ to denote the maximum executions, $\mathcal{E}_H^*$ to denote all the finite executions and $\mathcal{E}_H^\infty$ to denote the set of all infinite executions.

- *Reachability*

Reachability is a concept of hybrid systems which specifies that the certain state $\hat{q} \in Q$ can be reached in the hybrid automaton H through a finite number of executions. The reachability of a hybrid automaton can be expressed in the following equation:

$$Reach_H = \{(\hat{q}, \hat{x}) \in Q \times X : \exists(\{[\tau_i, \tau'_i]\}_{i=0}^N, q, x) \in \mathcal{E}_H^*, (q(N), x^N(\tau'_N)) = (\hat{q}, \hat{x})\}.$$

The set for which the continuous evaluation is impossible can be expressed as

$$Out_H = \{(q, x) \in Q \times X : \forall \varepsilon > 0, \exists t \in [0, \varepsilon), \psi(q, x, t) \notin D(q)\},$$

where $\psi(q, x, t)$ is a solution of $\dot{x} = f(q(i), x(t))$ for $x(0) = x^0$.

- *Non-Blocking and Deterministic*

The non-blocking and deterministic properties are two important criteria for analyzing the existence and uniqueness of the executions of hybrid automata. The non-blocking property implies that executions exist for all initial states and this is related to the controllability and closure. The deterministic property implies that the infinite executions are unique [12].

A hybrid automaton is non-blocking if $\mathcal{E}_H^\infty(q^0, x^0)$ is non-empty for all $(q^0, x^0) \in Init$. The following lemma can be used to determine the non-blocking property of a hybrid automaton.

Lemma I: A hybrid automaton H is non-blocking if for all $(q, x) \in Reach_H \cap Out_H$, there exists $(q, q') \in E$ so that $x \in G(q, q')$.

A hybrid automaton, H, is called deterministic if $\mathcal{E}_H^M(q^0, x^0)$ contains at most one element for all $(q^0, x^0) \in Init$. Also, there is a lemma to examine the deterministic property of H.

Lemma II: A hybrid automaton H is deterministic if and only if for all $(q,x) \in Reach_H$,

1) if $x \in G(q,q')$ for some $(q,q') \in E$, then $(q,x) \in Out_H$;
2) if $(q,q') \in E$ and $(q,q'') \in E$ with $q' \neq q''$, then $x \notin G(q,q') \cap G(q,q'')$;
3) if $(q,q') \in E$ and $x \in G(q,q')$, then $R(q,q'x)$ cotains at most one element.

- *Existence and uniqueness:*

Now, we present a theorem to determine the existence and uniqueness of the hybrid automaton, *H.*

Theorem I: If a hybrid automaton, *H,* satisfies Lemma I and Lemma II above, then H accepts a unique infinite execution for all $(q^0,x^0) \in Init_H$.

In the following sections, we will utilize these definitions and theorems to characterize a hybrid system.

3. *Hierarchical Modeling of Multiple Satellite Control:*

3.1 Ground Control Station (GCS) Function Definitions and Model:

1) Generate and send commands to the Mother-ships. For example, Imaging Command, would includes the coordinates of the area to be imaged and/or operation time period.
2) If the status feedback from a Mother-ship indicates that the command is un-executable, then GCS will re-generate a command to the Mother-ships.
3) Track the orbital status of all Mother-ships and Agents which are visible to the GCS.

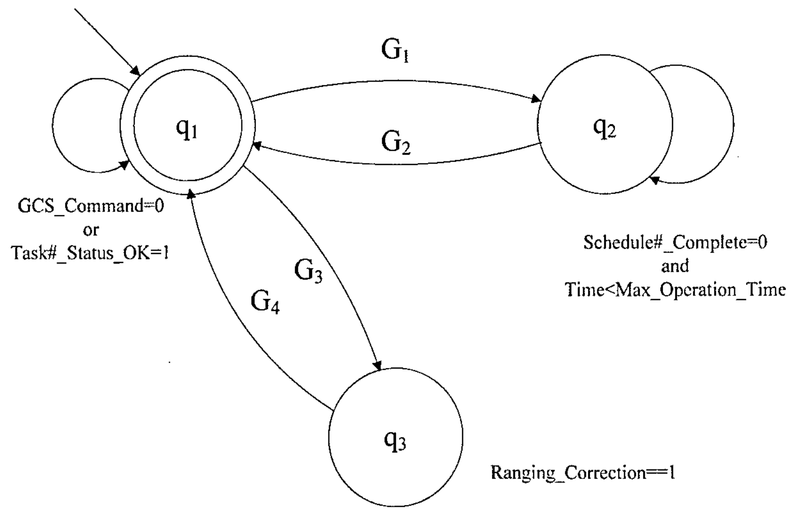

Figure 3. State Transition Diagram for Ground Control Station

For the Ground Control Station automaton, there are three discrete states q_1, q_2, and q_3, therefore $Q = \{q_1 \quad q_2 \quad q_3\}$. Where

- q_1 : Operation Node. In this node, GCS generates mission commands according to the requirement or the operator's input, then sends the command to the visible Mother-ships at the specific time. After sending the mission command to the Mother-ships, the system will jump from q_1 to q_2. In q_2, the system will wait for the Mother-ship to schedule the mission. If the mission has been successfully scheduled, then the node will switch back from q_2 to q_1. In q_1, the node will check the Task#_Status_OK tag to determine whether the mission is accomplished or failed. The "#" will be the mission number. If Task#_Status_OK = 1, it means that the mission was accomplished so that the system will stay in node q_1 waiting for the new mission command. The Task#_Status_OK = 0, means the mission number, #, failed during the last operation. The operator may send another command at this point. In q_1, GCS does ranging to the visible MSs and Agents to correct error in the navigation information. If GCS monitors the navigation error of the satellites, it will jump to q_3 to control the satellite to make the orbit maneuver.

- q_2: Command Schedule Status Check Node. In this node, the system checks the mission schedule status from the Mother-ships. If at least one Mother-ship sends back scheduling success message then the Schedule#_Complete tag is equal to 1, the system will jump back to q_1 for the new command. If more than one Mother-ships send back the command executable messages, then the Agent with the earliest possible execution time will execute the mission, and the system will jump back to q_1 with the tag Schedule#_Complete = 1, which means that the command has been successfully scheduled. The "#" is going to represent the command number so that there can be more than one command.
- q_3: Ranging Correction Node. In this node, GCS detects the altitude error of the MS or Agent, it sends the orbit command to the Mother-ships to control the Agents. After the orbit maneuver, the system will jump back to the previous node where it transits from.

The finite set defined for the Ground Control Station contains only one continuous variable $x_1 = t$, which represents the system time.

$$X = \{x_1\}.$$

The Domains of $Q = \{q_1 \quad q_2 \quad q_3\}$ are given as follows,

$$D(q_1) = \left\{ \begin{array}{l} x \in \mathbb{R}^1 : [(x_1 < Command\_Time)\text{and}(GCS\_Command = 0)] \\ \qquad\qquad \text{or } (Task\#\_Status\_OK = 1) \end{array} \right\},$$

$$D(q_2) = \left\{ x \in \mathbb{R}^1 : (x_1 \le Max\_Operation\_Time)\text{and}(Schedule\#\_Complete = 0) \right\},$$

$$D(q_3) = \left\{ x \in \mathbb{R}^1 : (Ranging\_Status\_OK \ne 1) \right\}.$$

The discrete transitions of the system are:

$$E = \{[(q_1, q_2) \quad (q_2, q_1) \quad (q_1, q_3) \quad (q_3, q_1) \quad]\}.$$

The guard conditions are given as:

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \end{bmatrix} = \left\{ \begin{array}{l} (q_1, q_2) \Rightarrow \left\{ \begin{array}{c} x \in \mathbb{R}^1 : [(x_1 \geq Command\_Time) \text{ and } (GCS\_Command = 1)] \\ \text{or } (Task\#\_Status\_OK = 0) \end{array} \right\} \\ (q_2, q_1) \Rightarrow \{ x \in \mathbb{R}^1 : (x_1 > Max\_Operation\_Time) \text{ or } (Schedule\#\_Complete = 1) \} \\ (q_1, q_3) \Rightarrow \{ x \in \mathbb{R}^1 : (Ranged\_Altitude \neq Reported\_Altitude) \} \\ (q_3, q_1) \Rightarrow \left\{ \begin{array}{l} x \in \mathbb{R}^1 : (Ranging\_Status\_OK = 1) \\ \text{and } [(x_1 < Command\_Time) \text{and} (GCS\_Command = 0)] \\ \quad \text{or } (Task\#\_Status\_OK = 1)] \end{array} \right\} \end{array} \right\}.$$

3.2 Mother-ship (MS) Function Definitions and Model:

1) Generate and send the task commands to control Agents in a group to implement orbit and attitude change.
2) Receive the mission command from GCS, parse the command for the task parameters calculation including orbit parameters and attitude for each Agent.
3) Communicate with other MSs to distribute the command from GCS.
4) Scheduling the task for each Agent.
5) In case that more than one Agents can execute the assigned task, MS will use the Agent which can execute at the earliest time. If all the Agents in control cannot schedule the task, MS will send message to other MSs so that other MSs can resume the task, at the same time MS will send a mission status message back to the GCS.
6) MS itself can perform the orbit maneuver and attitude change and execute the mission like an Agent.

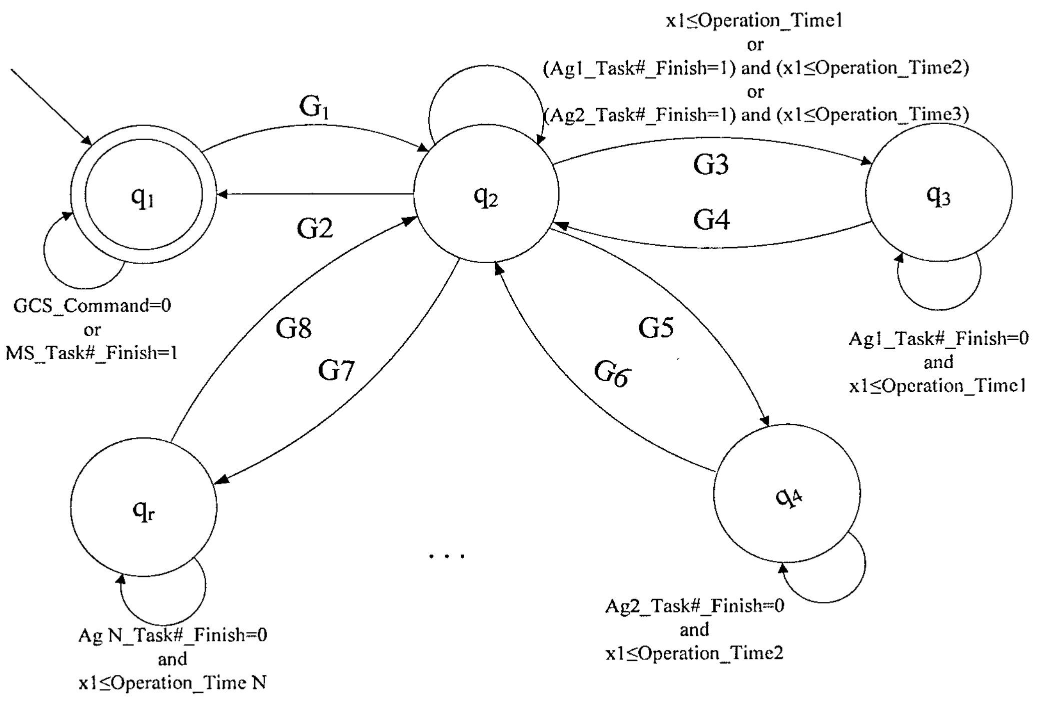

Figure 4. State Transition Diagram for the Mother-ships

Figure 4 shows the state transition diagram for the Mother-ship. The number of state transition automata nodes equals the number of agents plus two ($r = N + 2$). $Q = \{q_1 \quad q_2 \quad q_3 \quad q_4 \cdots q_r\}$, where

- q_1: Standby Node. In this node, MS will monitor the command from GCS and from Agents, if GCS_Command=1, the system will go to q_2, the task scheduling node. The system will jump from q_2 to q_1 when the system receives the status feedback messages from all Agents and the MS_Task#_Finish tag is marked to one. "#" stands for the number of commands issued by the MS. If MS receives the message from an Agent which does not belongs to any other MS, then MS will refresh its Agent table and add that Agent into its cluster. Then the number of Agent will be $N = N + 1$, which in turn changes $r = r + 1$, and a new node q_r will be added into the state transition model. Thus the system becomes scalable.

- q_2: MS Task Scheduling Node. In this node, the system will analyze the GCS command type, then calculate the orbit and attitude parameters for each Agent, it also calculates the task start and end time for each Agent. At this point, the MS send the status report to the GCS notifying it whether it is possible or not to execute the command. The MS also sends the command to the Agents. After sending the command to the Agents, it will monitor the mission status feedbacks from the Agents, if one or more Agents send back the mission failure messages to MS, the system will re-calculate the mission and re-schedule the task for the other Agents. When the system receive the status feedback messages from all Agents, then the system will send the mission status message to GCS to indicate success or failure. Then the system goes back to q_1.

The node numbers for the Agents are denoted $i = 3,\ldots,r$. This number depends on the number of agents, N.

- q_i: Agent $(i-2)$ Maneuver Node: From q_2 to q_i, the system sends the orbit and attitude command to Agent $(i-2)$ at the calculated task start time. Then the system will standby for the mission status feedback from Agent $(i-2)$. After the system receives the status information or if the time period exceeds the specification, the system will jump from q_i to q_2 with a tag Ag$(i-2)$_Task#_Finish to notify the status of the task for Agent $(i-2)$.

Note that q_2 to q_3, q_4 ... and q_r does not occur sequentially. The order depends on the command type from GCS and the calculations performed in q_2.

The finite set defined for the Mother-ship also contains only one continuous variable $x_1 = t$, which represents the system time.

$X = \{x_1\}$.

The Domains of $Q = \{q_1 \quad q_2 \quad q_3 \quad q_4 \quad \cdots \quad q_r\}$ are given as follows,

$D(q_1) = \{x \in \mathbb{R}^1 : (GCS\_Command = 0) \text{ or } (MS\_Task\#\_Finish = 1)\}$,

$$D(q_2) = \left\{ \begin{array}{ll} x \in \mathbb{R}^1 : & (x_1 < Ag1\_Command\_Time) \text{ or } \\ & [(Ag1\_Task\#\_Finish = 1) \text{ and } (x_1 < Ag2\_Command\_Time)] \\ & \text{or } [(x_1 > Ag1\_Operation\_Time) \text{ and } (x_1 < Ag2\_Command\_Time)] \\ & \text{or } [(Ag2\_Task\#\_Finish = 1) \text{ and } (x_1 < Ag3\_Command\_Time)] \\ & \text{or } [(x_1 > Ag2\_Operation\_Time) \text{ and } (x_1 < Ag3\_Command\_Time)] \end{array} \right\},$$

$$D(q_3) = \left\{ \begin{array}{ll} x \in \mathbb{R}^1 : & [(x_1 \geq Ag1\_Command\_Time) \text{ and } (x_1 \leq Ag1\_Operation\_Time)] \\ & \text{and } (Ag1\_Task\#\_Finish = 0) \end{array} \right\},$$

$$D(q_4) = \left\{ \begin{array}{ll} x \in \mathbb{R}^1 : & [(x_1 \geq Ag2\_Command\_Time) \text{ and } (x_1 \leq Ag2\_Operation\_Time)] \\ & \text{and } (Ag2\_Task\#\_Finish = 0) \end{array} \right\},$$

......

$$D(q_r) = \left\{ \begin{array}{ll} x \in \mathbb{R}^1 : & [(x_1 \geq AgN\_Command\_Time) \text{ and } (x_1 \leq AgN\_Operation\_Time)] \\ & \text{and } (AgN\_Task\#\_Finish = 0) \end{array} \right\}.$$

The discrete transitions of the system are:

$$E = \{[(q_1, q_2) \quad (q_2, q_1) \quad (q_2, q_3) \quad (q_3, q_2) \quad (q_2, q_4) \quad (q_4, q_2) \quad \cdots \quad (q_2, q_r) \quad (q_r, q_2)]\}.$$

The guard conditions are given as

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \\ \vdots \\ G_7 \\ G_8 \end{bmatrix} = \left\{ \begin{array}{l} (q_1, q_2) \Rightarrow \{ x \in \mathbb{R}^1 : (GCS\_Command = 1) \} \\ (q_2, q_1) \Rightarrow \left\{ \begin{array}{l} x \in \mathbb{R}^1 : \left(x_1 > \max \left\{ \begin{array}{l} Ag1\_Operation\_Time, \\ Ag2\_Operation\_Time, \\ Ag3\_Operation\_Time) \end{array} \right\} \right) \\ \qquad \text{or} \left[\begin{array}{l} (Ag1\_Task\#\_Finish = 1) \text{ and} \\ (Ag2\_Task\#\_Finish = 1) \text{ and} \\ (Ag3\_Task\#\_Finish = 1) \end{array} \right] \end{array} \right\} \\ (q_2, q_3) \Rightarrow \{ x \in \mathbb{R}^1 : (x_1 \geq Ag1\_Command\_Time) \} \\ (q_3, q_2) \Rightarrow \left\{ \begin{array}{l} x \in \mathbb{R}^1 : (x_1 > Ag1\_Operation\_Time) \\ \qquad \text{or } (Ag1\_Task\#\_Finish = 1) \end{array} \right\} \\ (q_2, q_4) \Rightarrow \{ x \in \mathbb{R}^1 : (x_1 \geq Ag2\_Command\_Time) \} \\ (q_4, q_2) \Rightarrow \left\{ \begin{array}{l} x \in \mathbb{R}^1 : (x_1 > Ag2\_Operation\_Time) \\ \qquad \text{or } (Ag2\_Task\#\_Finish = 1) \end{array} \right\} \\ \quad \vdots \\ (q_2, q_r) \Rightarrow \{ x \in \mathbb{R}^1 : (x_1 \geq AgN\_Command\_Time) \} \\ (q_r, q_2) \Rightarrow \left\{ \begin{array}{l} x \in \mathbb{R}^1 : (x_1 > AgN\_Operation\_Time) \\ \qquad \text{or } (AgN\_Task\#\_Finish = 1) \end{array} \right\} \end{array} \right\} .$$

3.3 Agent Function Definition and Model:

The Agent functions are as follows.

1) Receive orbit and attitude commands from MS.
2) Perform the orbit maneuver and attitude change for the scheduled tasks.
3) Send the task status feedback to MS to notify whether the task is successful.
4) Switch to MS in the case that the MS malfunctions, this operation is done autonomously by the Agents in that group.

The satellite attitude and orbital perturbations are modeled as follows. As for the attitude model, we use the deterministic version of the model developed in [17]. The satellite attitude model for the Euler angle, angular velocity, and reaction wheel speed vector is given as,

$$\underline{\dot{\Theta}} = \frac{1}{\cos\psi}\begin{bmatrix} \cos\psi & -\cos\phi\sin\psi & \sin\phi\sin\psi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\cos\psi & \cos\phi\cos\psi \end{bmatrix}\underline{\omega} + \begin{bmatrix} 0 \\ n \\ 0 \end{bmatrix},$$

$$I_g\underline{\dot{\omega}} = -\underline{\omega}\times(I_t\underline{\omega} + L^T I_w\underline{\Omega}) - L^T\underline{\tau}_w + \underline{\tau}_{thrust} + 3n^2\underline{c}_3{}^{\times}I_t\underline{c}_3$$

$$\underline{\dot{\Omega}} = I_w^{-1}\underline{\tau}_w - L\underline{\dot{\omega}},$$

where the superscript, T, denotes transpose, I_t is the total moment of inertia for the satellite body, I_w is the moment of inertial matrix for the wheels, $I_g = I_t - L^T I_w L$ is the total moment of inertia minus the moment of inertia of the wheels, L is the wheel orientation matrix, $\underline{\Theta} = [\phi, \theta, \psi]^T$ is the roll, pitch, yaw Euler angles, $\underline{\omega}$ is the angular velocity vector in body fixed coordinate system, $\underline{\Omega}$ is the wheel speed vector $\underline{\tau}_w$ is the absolute torque due to the reaction wheels, $\underline{\tau}_{thruster}$ is the torque due to the thrusters, n is the orbital rate, $\underline{c}_3^{\times}$ represents the cross product matrix of a vector $\underline{c}_3$.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{BFC} = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi & -\cos\psi\sin\theta \\ -\cos\phi\sin\psi\cos\theta + \sin\phi\sin\theta & \cos\phi\cos\psi & \cos\phi\sin\psi\sin\theta + \sin\phi\cos\theta \\ \sin\phi\sin\psi\cos\theta + \cos\phi\sin\theta & -\sin\phi\cos\psi & -\sin\phi\sin\psi\sin\theta + \cos\phi\cos\theta \end{bmatrix}\begin{bmatrix} x_L \\ y_L \\ z_L \end{bmatrix}_{LVLH}$$
$$\equiv \begin{bmatrix} \underline{c}_1 & \underline{c}_2 & \underline{c}_3 \end{bmatrix}\begin{bmatrix} x_L \\ y_L \\ z_L \end{bmatrix}_{LVLH}$$

where BFC stands for body-fixed-coordinate and $LVLH$ stands for local-vertical-local-horizontal coordinate system. In LVLH model, x is the direction of the velocity vector, z is pointing towards the Earth, and y completes the triad. The above model is linearized and utilized as the attitude dynamics model [16]. We note that we are ignoring the magnetic and the aerodynamic torques.

As for the orbital perturbation model, we incorporate atmospheric drag and Earth oblateness (J2) effects only. We assume other effects are negligible. The following orbit decay rate due to the atmospheric drag is used. This drag force will affect the velocity in $-x$ direction only.

$$F_{drag} = -2\pi(C_d A / m)\rho \frac{r^2}{P}$$

where ρ is the atmospheric density at a particular point in the solar cycle, C_d is the satellite ballistic coefficient, A is the satellite cross-sectional area, m is the satellite mass, r is the Earth radius plus altitude, and P is the orbital period.

Earth oblateness will be the other perturbing effect that will be included in the orbit model. This perturbing force due to J2 on orbit altitude can be represented as,

$$F_x = -\frac{3}{2}\frac{\mu J_2 R_E{}^2}{r^4}\left[\left(\sin^2 i\right)\left(\sin 2u\right)\right]$$
$$F_y = -\frac{3}{2}\frac{\mu J_2 R_E{}^2}{r^4}\left[\sin i \cos i \sin u\right]$$
$$F_z = -\frac{3}{2}\frac{\mu J_2 R_E{}^2}{r^4}\left[1 - \frac{3}{2}\left(\sin^2 i\right)\left(1-\cos 2u\right)\right]$$

where u is the argument of latitude (the sum of true anomaly υ and argument of perigee p), J_2 is taken as 0.00108263, i is the inclination of the orbit, μ is the gravitational parameter of the Earth, R_E is the equatorial radius of the Earth, and r is the Earth radius plus the altitude, h, of the satellite. The total altitude decay rate, F, for the satellite is the sum of the above four equations. To simplify the orbital maneuver, the Hohmann transfer orbital maneuver will be used to control the orbit.

For the modeling of the discrete part of the Agent, we apply the hybrid automation, the set of discrete nodes can be defined into set $Q = \{q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_5\}$, which represents the following Agent satellite nodes:

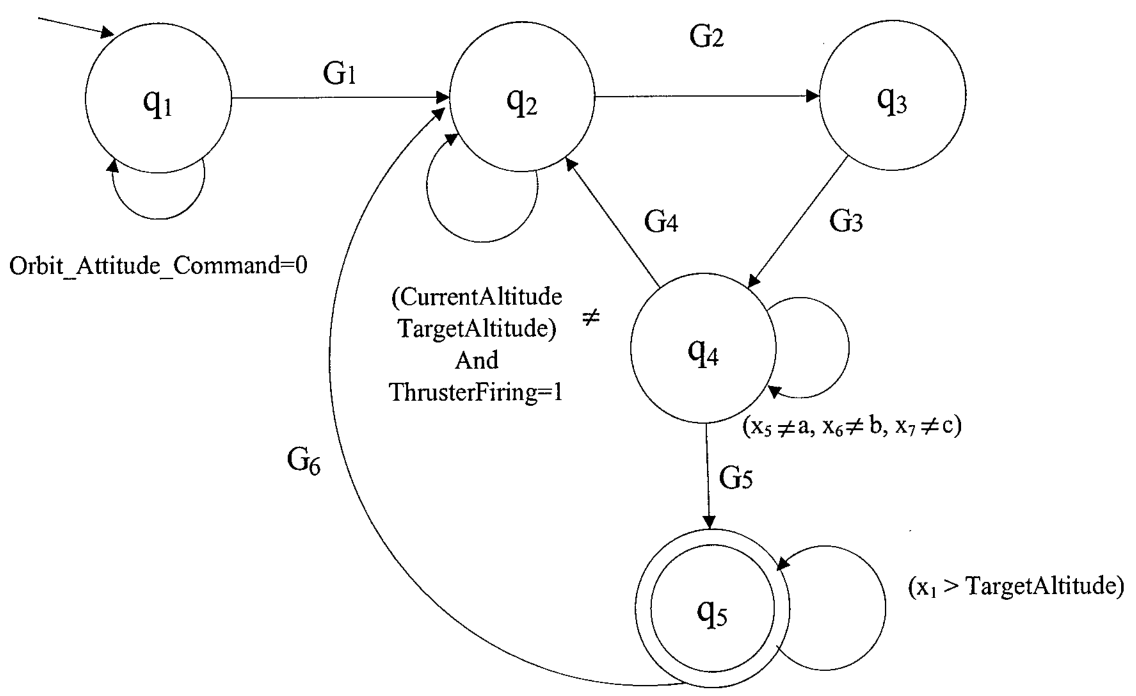

Figure. 5. State transition diagram for the general satellite control system.

- q_1 : Initial Node. In this node, the satellite is at the initial orbit with its initial altitude, initial attitude and initial inclination. The satellite will check the status of *Orbit-Attitude Command* to see if there is a requirement to do the orbit and inclination change. If *Orbit-Attitude Command* is true, the satellite will jump to the q_2 and determine whether an orbit maneuver is necessary or not.
- q_2 : Orbit Maneuver Node. In this node the satellite changes its altitude to the target altitude determined by the input condition. It will check the current altitude with the target altitude, when there is a difference between the current and the target, then it will check the current attitude to see whether the satellite is in the thruster firing mode, if not, it will jump into q_3 to change its attitude to the thruster firing mode to boost the satellite to the target altitude. If the satellite is already in the thruster firing mode, satellite will perform a thruster fire to shoot the satellite to the target orbit. If there is no need to change the altitude (the *Orbit-Attitude Command* only includes an attitude change), the satellite will jump to q_3 to calculate the Euler angles change to reach target attitude directly. This node is triggered by *Orbit-Attitude Command* from q_1 or from q_4, the Attitude Adjustment Node
- q_3 : Attitude Calculation Node. In this node, the satellite calculates and outputs the different Euler angles, which are needed by q_4 to implement the attitude change, according to the input

conditions. For example, if the satellite goes from q_2 to q_3 to implement an target attitude adjustment, q_3 node will calculate the roll, pitch and yaw angles used in q_4 node to adjust the attitude to the target; if the satellite carries out orbit and inclination change due to the *Orbit-Attitude Command*, q_3 node will calculate the corresponding roll, pitch and yaw angles for the thruster firing and inclination change.

- q_4: Attitude Adjustment Node. In this node, the satellite adjusts its attitude to the input Euler angles calculated by Attitude Calculation Node for the satellite orbit and attitude change. When the current attitude reaches to the calculated attitude from q_3, it will then check the current altitude with the target altitude, if the current altitude does not match the target, the satellite will jump to q_2 node again to implement the orbit maneuver otherwise it will jump into q_5 satellite operation node.
- q_5: Satellite Operation Node. In this node the satellite operates at the final orbit with a target attitude of roll, pitch, and yaw angles. The satellite will check the current attitude and the current altitude, if the altitude of satellite drops below the target altitude or the current attitude is different from the target attitude, it will jump to the q_2 node and the orbit maintenance procedure will take in effect to maintain the orbit and the attitude of the satellite.

The finite set of real valued continuous variables for the LQR case is defined as, $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}\}$, where x_1 represents the altitude (h) of the satellite, x_2 represents angular velocity ω_x, x_3 represents the angular velocity $\delta\omega_y$, x_4 represents the angular velocity ω_z, x_5 represents the roll Euler angle ϕ, x_6 is the pitch Euler angle θ, x_7 is the yaw Euler angle ψ, and x_8 to x_{11} represent the angular velocities of the four reaction wheels. The continuous dynamics are specified as

$$f(q_1, x) = \begin{bmatrix} \gamma & \omega_x & \delta\omega_y & \omega_z & \phi & \theta & \psi & \Omega_1 & \Omega_2 & \Omega_3 & \Omega_4 \end{bmatrix}^T,$$

$$f(q_2, x) = \begin{bmatrix} \dfrac{a(1-e^2)}{1+e\cos\nu} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

The deterministic version of the satellite with reaction wheels can be modeled as follows.

$$\frac{d}{dt}\begin{bmatrix}\phi\\ \theta\\ \psi\\ \omega_x\\ \delta\omega_y\\ \omega_z\\ \Omega_1\\ \Omega_2\\ \Omega_3\\ \Omega_4\end{bmatrix}=A\begin{bmatrix}\phi\\ \theta\\ \psi\\ \omega_x\\ \delta\omega_y\\ \omega_z\\ \Omega_1\\ \Omega_2\\ \Omega_3\\ \Omega_4\end{bmatrix}+B\begin{bmatrix}\tau_\omega\\ \tau_{thruster}\end{bmatrix},\ \text{where } A=\begin{bmatrix}0\ 0\ n & 1\ 0\ 0 & 0_{3\times4}\\ 0\ 0\ 0 & 0\ 1\ 0 & \\ 0\ 0\ n & 0\ 0\ 1 & \\ 3I_g^{-1}n^2N_3 & I^{-1}nN_1 & I^{-1}nN_2\\ -L3I_g^{-1}n^2N_3 & -LI^{-1}nN_1 & -LI^{-1}nN_2\end{bmatrix},\quad B=\begin{bmatrix}0_{3\times4} & 0_{3\times3}\\ -I_g^{-1}L' & I^{-1}\\ I_\omega^{-1}+LI_g^{-1}L' & -LI_g^{-1}\end{bmatrix}.$$

We can then calculate the continuous dynamics of q_4 as

$$f(q_4,x)=\begin{bmatrix} h-F\times\Delta t\\ \int\left(\phi\times 9.2334\text{e-}7+\omega_z\times 2.8937\text{e-}4+\Omega_1\times 2.1753\text{e-}8+\Omega_2\times 2.1753\text{e-}8+\Omega_3\times 2.1753\text{e-}8+\Omega_4\times 2.1753\text{e-}8\right)dt\\ \int\left(-\theta\times 2.2257\text{e-}6\right)dt\\ \int\left(-\omega_x\times 8.3747\text{e-}4-\Omega_1\times 3.0565\text{e-}8+\Omega_2\times 3.0565\text{e-}8+\Omega_3\times 3.0565\text{e-}8-\Omega_4\times 3.0565\text{e-}8\right)dt\\ \int\left(\psi\times 1.0636\text{e-}3+\omega_x\right)dt\\ \int\left(\delta\omega_y\right)dt\\ \int\left(-\phi\times 1.0636\text{e-}3+\omega_z\right)dt\\ \int\left(-\phi\times 5.3309\text{e-}7+\theta\times 1.285\text{e-}6+\omega_x\times 4.8352\text{e-}4-\omega_z\times 1.6707\text{e-}4+\Omega_1\times 5.0877\text{e-}9-\Omega_2\times 3.0205\text{e-}8-\Omega_3\times 3.0205\text{e-}8+\Omega_4\times 5.0877\text{e-}9\right)dt\\ \int\left(\phi\times 5.3309\text{e-}7+\theta\times 1.285\text{e-}6+\omega_x\times 4.8352\text{e-}4+\omega_z\times 1.6707\text{e-}4+\Omega_1\times 3.0205\text{e-}8-\Omega_2\times 5.0877\text{e-}9-\Omega_3\times 5.0877\text{e-}9+\Omega_4\times 3.0205\text{e-}8\right)dt\\ \int\left(\phi\times 5.3309\text{e-}7-\theta\times 1.285\text{e-}6+\omega_x\times 4.8352\text{e-}4+\omega_z\times 1.6707\text{e-}4+\Omega_1\times 3.0205\text{e-}8-\Omega_2\times 5.0877\text{e-}9-\Omega_3\times 5.0877\text{e-}9+\Omega_4\times 3.0205\text{e-}8\right)dt\\ \int\left(-\phi\times 5.3309\text{e-}7-\theta\times 1.285\text{e-}6+\omega_x\times 4.8352\text{e-}4-\omega_z\times 1.6707\text{e-}4+\Omega_1\times 5.0877\text{e-}9-\Omega_2\times 3.0205\text{e-}8-\Omega_3\times 3.0205\text{e-}8+\Omega_4\times 5.0877\text{e-}9\right)dt \end{bmatrix},$$

$$f(q_5,x)=\begin{bmatrix}\gamma & \omega_x & \delta\omega_y & \omega_z & \phi & \theta & \psi & \Omega_1 & \Omega_2 & \Omega_3 & \Omega_4\end{bmatrix}^T.$$

There is no continuous dynamics in state q_3 because that q_3 is the static node, in this node the Euler angles are calculated and outputted directly according to the input conditions.

The domains—where the continuous dynamics are valid—are given as

$$D(q_1)=\begin{Bmatrix}x\in\mathbb{R}^{11}:(\text{Orbit\_Attitude\_Command}=0)\\ \text{and}(x_5=\text{TargetRoll},x_6=\text{TargetPitch},x_7=\text{TargetYaw})\end{Bmatrix},$$

$$D(q_2)=\left\{x\in\mathbb{R}^{11}:(x_1\le\text{TargetAltitude})\text{ or }(\text{Orbit\_Attitude\_Command}=1)\right\},$$

Because that q_3 has no continuous dynamics so that the domain of q_3 can be presented as:

$D(q_3)=\left\{x \in \mathbb{R}^{11},\ \text{EulerOutput=0}\right\}$,

$D(q_4)=\left\{x \in \mathbb{R}^{11} : (x_5 \neq \text{CalculatedRoll}, x_6 \neq \text{CalculatedPitch}, x_7 \neq \text{CalculatedYaw})\right\}$,

$D(q_5)=\left\{x \in \mathbb{R}^{11} : (x_1 \geq \text{TargetAltitude})\right\}$.

Initial conditions enforced on the system are:

$$Init=\{q_1\}\times\left\{\begin{array}{l} x \in \mathbb{R}^{11} : (x_1 = \mathit{init\_altitude}) \wedge (x_2 = 0°/\text{s}, x_3 = 0°/\text{s}, x_4 = 0°/\text{s}) \\ \qquad \wedge (x_5 = 0°, x_6 = 0°, x_7 = 0°) \wedge (x_8 = 0°, x_9 = 0°, x_{10} = 0°, x_{11} = 0°) \end{array}\right\}.$$

The discrete transitions in the system are:

$$E=\{[(q_1,q_2)\quad (q_2,q_3)\quad (q_3,q_4)\quad (q_4,q_2)\quad (q_4,q_5)\quad (q_5,q_2)]\}.$$

The guard conditions are defined as

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \\ G_6 \end{bmatrix} = \left\{ \begin{array}{l} (q_1,q_2) \Rightarrow \left\{x \in \mathbb{R}^{11} : (\text{Orbit\_Attitude\_Command} = 1)\right\} \\ (q_2,q_3) \Rightarrow \left\{\begin{array}{l} x \in \mathbb{R}^{11} : (x_1 \geq \text{Target\_Altitude})\mathit{or} \\ (x_5 \neq \text{TargetRoll}, x_6 \neq \text{TargetPitch}, x_7 \neq \text{TargetYaw}) \end{array}\right\} \\ (q_3,q_4) \Rightarrow \{\text{EulerOutput} = 1\} \\ (q_4,q_2) \Rightarrow \left\{\begin{array}{l} x \in \mathbb{R}^{11} : [(x_5 = \text{CalculatedRoll}, x_6 = \text{CalculatedPitch}, x_7 = \text{CalculatedYaw})] \\ \text{and } (x_1 < \text{Target\_Altitude}) \end{array}\right\} \\ (q_4,q_5) \Rightarrow \left\{\begin{array}{l} x \in \mathbb{R}^{11} : [(x_5 = \text{CalculatedRoll}, x_6 = \text{CalculatedPitch}, x_7 = \text{CalculatedYaw}) \\ \text{and } (x_1 \geq \text{Target\_Altitude})] \end{array}\right\} \\ (q_5,q_2) \Rightarrow \left\{\begin{array}{l} x \in \mathbb{R}^{11} : [(x_5 = \text{TargetRoll}, x_6 = \text{TargetPitch}, x_7 = \text{TargetYaw}) \\ \mathit{and}\ (x_1 \leq \text{Target\_Altitude})] \end{array}\right\} \end{array} \right\}$$

The reachability of each node in this case, is represented as

$$
\begin{aligned}
Reach_H = &\{q_1\}\times\{x\in\mathbb{R}^{11}\}\\
&\bigcup\{q_2\}\times\left\{x\in\mathbb{R}^{11}:(\text{Orbit\_Attitude\_Command}=1)\text{ or }(x_1\le\text{TargetAltitude})\right\}\\
&\bigcup\{q_3\}\times\{x\in\mathbb{R}^{11}\}\\
&\bigcup\{q_4\}\times\left\{\begin{array}{l} x\in\mathbb{R}^{11}:\left[\begin{array}{l}(\text{Orbit\_Attitude\_Command}=1)\\ \text{and }(x_5\ne\text{CalculatedRoll},x_6\ne\text{CalculatedPitch},x_7\ne\text{CalculatedYaw})\end{array}\right]\\ \quad\text{or }[(x_1\le\text{TargetAltitude})\wedge(x_5\ne\text{CalculatedRoll},x_6\ne\text{CalculatedPitch},x_7\ne\text{CalculatedYaw})]\end{array}\right\}\\
&\bigcup\{q_5\}\times\left\{\begin{array}{l}x\in\mathbb{R}^{11}:(x_1\ge\text{TargetAltitude})\wedge[(x_5=\text{TargetRoll},x_6=\text{TargetPitch},x_7=\text{TargetYaw})]\\ \quad\text{and (Current\_Inclination=TargetInclination)}\end{array}\right\}.
\end{aligned}
$$

The Out_H is:

$$
\begin{aligned}
Out_H = &\{q_1\}\times\left\{x\in\mathbb{R}^{11}:\text{Orbit\_Attitude\_Command}\ =\ 1\right\}\\
&\bigcup\{q_2\}\times\left\{x\in\mathbb{R}^{11}:(x_1\ge\text{TargetAltitude})\text{ or }(\text{Orbit\_Attitude\_Command}=0)\right\}\\
&\bigcup\{q_3\}\times\left\{x\in\mathbb{R}^{11}:\text{EulerOutput}=1\right\}\\
&\bigcup\{q_4\}\times\left\{x\in\mathbb{R}^{11}:(x_5=\text{CalculatedRoll},x_6=\text{CalculatedPitch},x_7=\text{CalculatedYaw})\right\}\\
&\bigcup\{q_5\}\times\left\{x\in\mathbb{R}^{11}:(x_1\le\text{TargetAltitude})\right\}.
\end{aligned}
$$

The equations show that q_1, q_2, q_3, q_4 and q_5 can all be reached from the initial condition within a finite execution $(\tau, q)\in\mathcal{E}_H^*$ with $\langle\tau\rangle = N<\infty$ and $q(\tau_N{}') = q_i$.

For Non-Blocking property, we examine $(q,x)\in Reach_H \bigcap Out_H$, where

$$
\begin{aligned}
Reach_H \bigcap Out_H = &\{q_1\}\times\{x\in\mathbb{R}^{11}:\text{Orbit\_Attitude\_Command=1}\}\\
&\bigcup\{q_2\}\times\{x\in\mathbb{R}^{11}:(x_1=\text{TargetAltitude})\}\\
&\bigcup\{q_3\}\times\{x\in\mathbb{R}^{11}:\text{EulerOutput=1}\}\\
&\bigcup\{q_4\}\times\{x\in\mathbb{R}^{11}:(\text{Orbit\_Attitude\_Command=1})\text{ or }(x_1\le\text{TargetAltitude})\}\\
&\bigcup\{q_5\}\times\{x\in\mathbb{R}^{11}:(x_1=\text{TargetAltitude})\}.
\end{aligned}
$$

From the above result, we note that for all $(q,x)\in Reach_H \bigcap Out_H$, there exists $(q,q')\in E$ such that $x\in G(q,q')$. This shows that the hybrid automaton is non-blocking.

For the deterministic property, we study the expression of $Reach_H$ again, where we can find that from the initial condition, (q_1,x), the only transition that can occur is (q_1,q_2) when $(q,x)\in Out_H$; for (q_2,x), the transitions that can occur are (q_2,q_3) when $x_1\ge\text{TargetAltitude}$, when $(q,x)\in Out_H$; and for (q_3,x), the transition (q_3,q_4) will occur when EulerOutput equals to 1; for (q_4,x), the possible transitions are

(q_4, q_2) and (q_4, q_5), however, $G(q_4, q_2) \cap G(q_4, q_5) = \phi$, thus for all $(q, x) \in Reach_H$, $x \notin G(q_4, q_2) \cap G(q_4, q_5)$; when the node is in (q_5, x), the only transition that can occur is (q_4, q_5), when $(q, x) \in Out_H$. Therefore, the conditions of Lemma II are satisfied. The result verifies that the hybrid automata of general control case are deterministic. According to Theorem I, we conclude that the hybrid automaton accepts a unique infinite execution.

4. Simulations

In this section, we illustrate this hierarchical hybrid system architecture by simulating the mission of a remote sensing satellite network. In this scenario, there is a Ground Control Station located at latitude 40.0 N and longitude 75.6 W, near the city of Philadelphia. The scenario also includes six LEO satellites which are grouped into two clusters. Each cluster consists of one Mother-ship and two Agents, satellites within a cluster are in the same orbit but with different argument of Perigees. Two orbits, orbit A and orbit B are of the same altitude, which is set as 680km. The inclination of orbit A is 28 degrees and the inclination of orbit B is 58 degrees. The model of the LEO satellite used in this scenario is the Korea Multipurpose Satellite (Kompsat), which is a LEO remote sensing satellite. The attitude parameters associated with this satellite are given in [17].

The mission of this scenario is to control these two clusters of satellites to take images of a certain area. To take the images, the satellite switches its attitude from satellite flying mode to image taking mode. Figure 6 shows the difference between satellite flying mode and image taking mode. In the satellite flying mode, the yaw, pitch and roll angles of the satellite are 0, 0, 0 degrees respectively, in the imaging mode, the attitude of the satellite becomes 0, 90, 0 degrees.

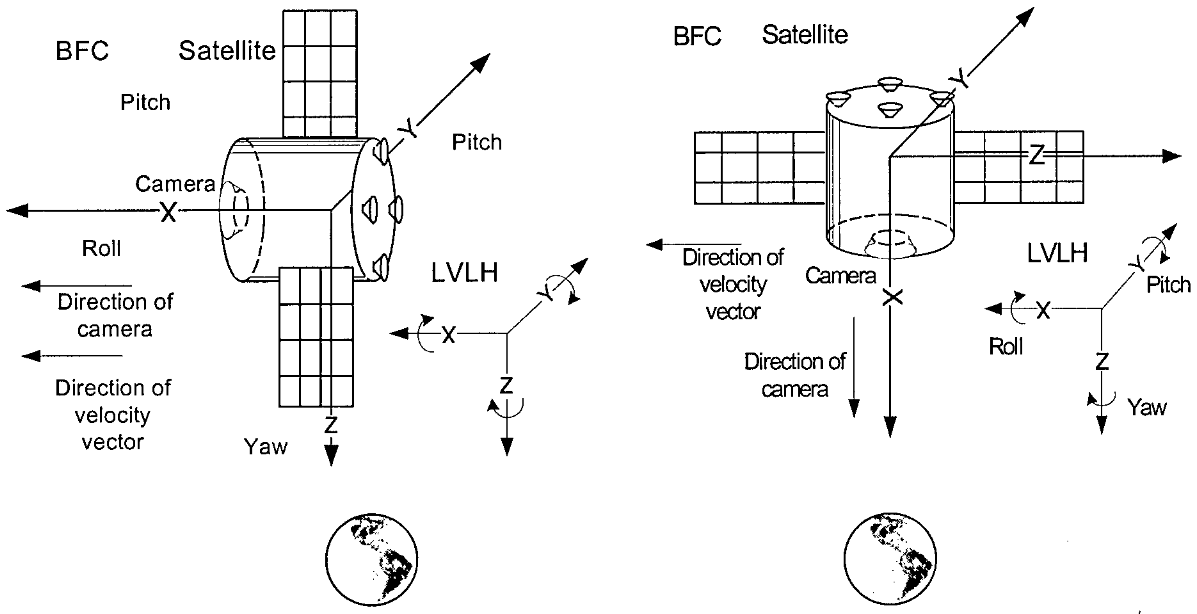

Figure 6. Left: Satellite Flying Mode, Right: Image Taking Mode.

In the first simulation, the mission begins with the command sent by the Ground Control Station. The command is to take images over the area with latitude 22.5 N and longitude 41.8 W. The command is sent to Mother-ship B on orbit B, which is visible from the Ground Control Station. Then the command is relayed to the Mother-ship A via the link between the two Mother-ships. After Mother-ship receives the command, it analyzes the command and decomposes the mission into different tasks for the Agents in the cluster. Then it calculates the orbit parameters and schedules the tasks for each Agent. The Mother-ships generate these coordination commands so as to control the Agents to take images of the target area.

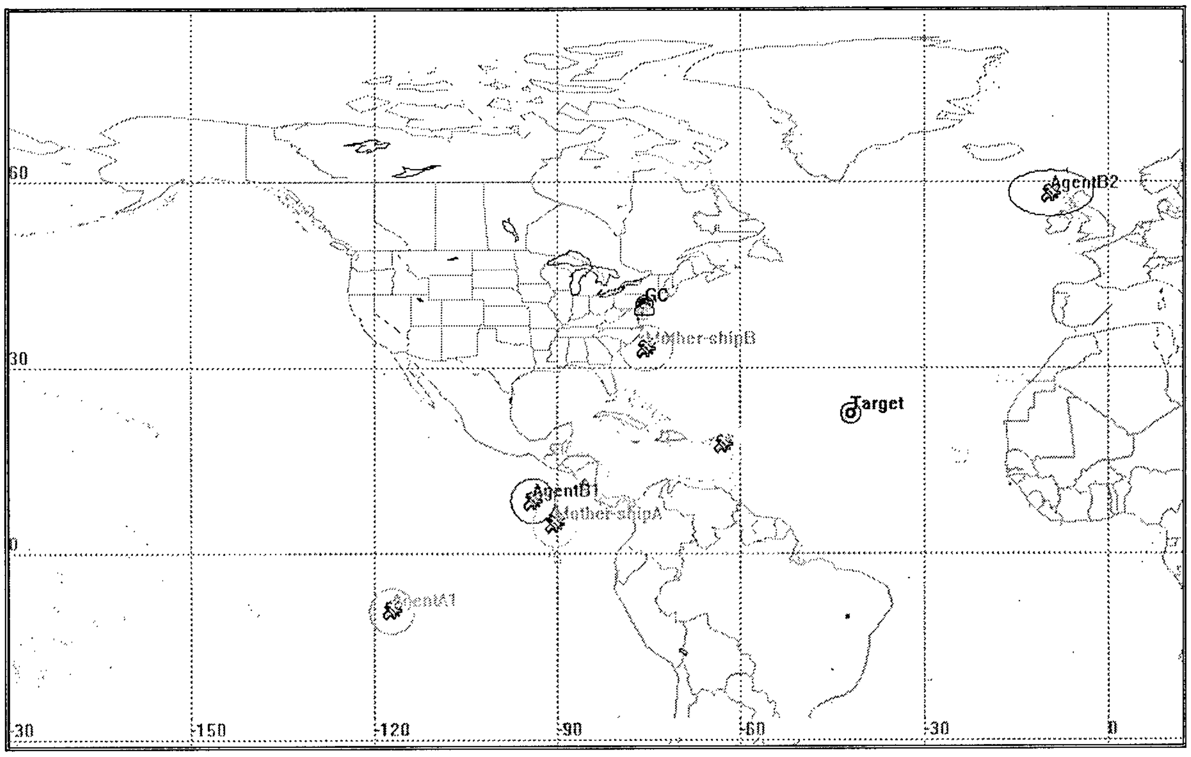

Figure 7. The initial deployment of the satellites network

As we observe in the simulation, satellites in orbit B pass through the target area before satellites in orbit A, and Mother-ship B controls Agent B2 to switch from satellite flying mode to image taking mode at the simulation time 683 seconds which translates into about six hours fifty minutes from the begging of the mission. Agent B2 returns back to thruster firing mode at 783 seconds. Then, Mother-ship B controls Agent B1 to take images from simulation time 871 seconds to 971 seconds. At the second stage, in order to obtain the better image quality, Mother-ship A on orbit A controls Agent A2 and Agent A1 to transfer into the lower orbit with the altitude of 500km before switching to the image mode. After performing the image taking tasks, Agent A1 and Agent A2 will go back to their original orbit with an altitude of 680km. Agent A2 goes into the target area at simulation time 2290 seconds, which means that the Agent A2 goes to the target area after twenty two hours fifty six minutes after the mission beginning. Therefore, Mother-ship A controls Agent A2 to take images from 2290 seconds to 2390 seconds. And Agent A1 takes images from 2487 seconds to 2587 seconds. The upper graph of Figure 8 shows the attitude change of each satellite in the simulation. We notice that Agent B2 first changes its attitude from 0, 0, 0, degrees of yaw, pitch, and roll angles to 0, 90, 0 degrees to switch to the image taking mode. Then, Agent B1, Agent A2, and Agent A1 switch to the image taking mode in file. The lower graph of Figure 8 gives the orbit change of satellites in cluster A. From the graph we can see that Agent A1 and Agent A2 change their orbit before taking images, and go back to the higher orbit when the tasks are finished.

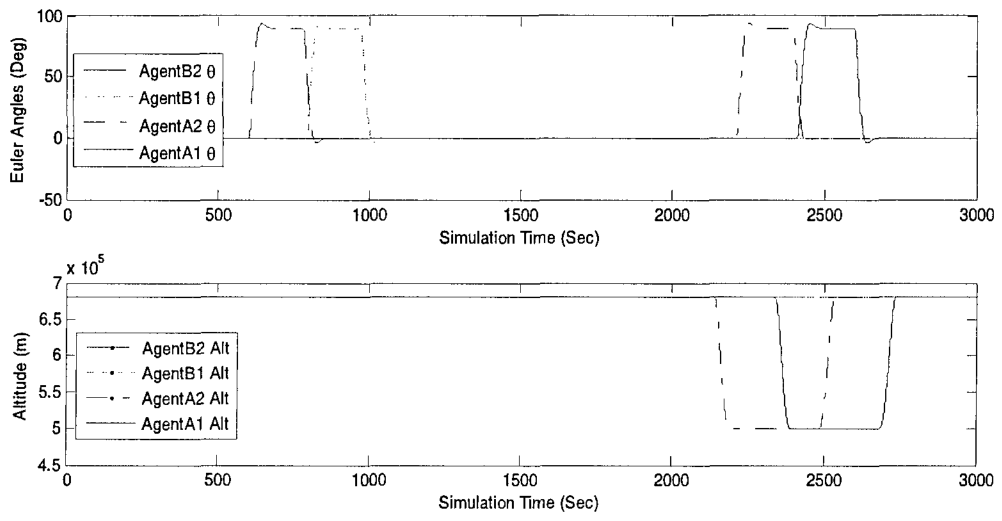

Figure 8. Attitude and orbit change of controlling four satellites in two clusters

From the simulation results, we notice that the Mother-ship acts as the coordinator, which receives the command from the Ground Control Station, and generates the coordination information for the Agents. It takes over some control functions from Ground Control Station and is able to control all Agents in the cluster even if they are invisible from the Ground Control Station. The simulation shows our hierarchical system architecture is effective for the multiple satellite control.

The second simulation is to demonstrate the robustness and intelligence of the system. The command is the same as the previous simulation. However, we assume this time Agent B2 fails in executing the image taking task, then, Mother-ship B will reschedule the tasks to the rest Agents. In cluster B, there is only Agent B1 left, and Agent B1 has been scheduled to take images from simulation time 871 seconds to 971 seconds, therefore, Mother-ship B itself will switch into image taking mode and perform the image taking task when it passes through the target area. Then, we assume the Mother-ship A malfunctions. In this case, Agents in cluster A will elect a new Mother-ship to take over the control over the cluster A. Here we assume Agent A1 becomes the new Mother-ship, it communicates with Mother-ship B to notify this change. Then, it will control Agent A1 and itself to take images for the target area. Figure 9 shows the mode switching of Mother-ship B, Agent B1, Agent A2, and Agent A1.

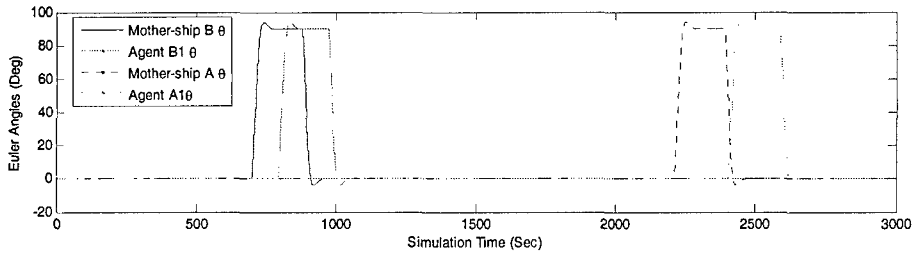

Figure 9. Mission rescheduling and fault handling of the system

The simulations we have done are simple examples to demonstrate that using the hierarchical hybrid architecture to control the operation of multiple satellites is feasible. We can extend the model to the more complex scenarios, where there are more Mother-ships and Agents participating in the mission, and the control of the satellites will include attitude, orbit and inclination maneuver. The functions of Mother-ship and the Agents can also be altered to meet the different mission requirements. These researches are left as the future works.

5. Conclusions

In this report, we introduced a three tier hierarchical hybrid architecture to control the multiple remote sensing satellite system. The three tiers consist of the Ground Control Station, Mother-ship, and Agent. This hierarchical architecture lessens the control and communication burden of the ground control station, and improves the effectiveness and robustness of the system.

Because the control scheme of the system includes both discrete and continuous dynamics, we applied hybrid automata theory to the orbit and attitude control of the agents, and discrete event system theory to control the ground control station and the Mother-ships. We established the mathematical model for each tier to describe this hierarchical hybrid system and investigated the interrelationship among different tiers. We showed that by using hierarchical hybrid architecture, we can implement an autonomous, intelligent control system for the multiple remote sensing satellite system.

We simulated this hierarchical hybrid system with Matlab, Simulink and Stateflow. Through the simulations, we demonstrated that a group of remote sensing satellites can be cooperatively controlled by applying hierarchical hybrid system architecture onto the satellites. The simulation results also illustrate the effectiveness and the robustness of the system.

References

[1] J. R. Wertz, *Space Mission Analysis and Design*, Third Edition, Microcosm Press, 1999.

[2] J. S. Albus, R. Lumia, H. McCain, "Hierarchical Control of Intelligent Machines Applied to Space Station Telerobots," *IEEE Transactions on Aerospace and Electronic Systems, Vol. 24. No. 5,* September, 1988.

[3] T. Fukuda, T. Shibata, "Hierarchical Control System in Intelligent Robotics and Mechatronics," *Proceedings of the IECON' 93, International Conference on Industrial Electronics, Control, and Instrumentation,* 15-19, Nov. 1993.

[4] S. Li, J. D. Boskovic, S. Seereeram, R. Prasanth, J. Amin, R. K. Mehra, R. W. Beard, T. W. McLain "Autonomous Hierarchical Control of Multiple Unmanned Combat Air Vehicles (UCAVs)," *Proceedings of the American Control Conference,* Anchorage, AK, May 8-10, 2002.

[5] R. H. Byrne, "A Practical Implementation of a Hierarchical Control System for Telerobotic Land Vehicles," *IEEE Aerospace and Electronics System Magazine, Vol. 7, Issue 10,* Oct. 1992.

[6] M. Ilyas and I. Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*, CRC Press, pp. 13–3, 2005.

[7] J. Liu, X., Liu, T. K. Koo, B. Sinopoli, S. Sastry, and E.A. Lee, "A Hierarchical Hybrid System Model and Its Simulation," *Proceedings of the 38th Conference on Decision and Control,* Phoenix, Arizona, Dec. 1999, pp3508-3513.

[8] Y. Sun, J. Tan, and N. Xi, "Hybrid System Model for Event-Based Planning and Control of Robot Operations," *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation,* July 16-20, 2003, Kobe, Japan.

[9] S. Zelinski, T. J. Koo, and S. Sastry, "Hybrid System Design for formations of Autonomous Vehicles," *42nd IEEE Conference, Decision and Control,* Maui, Hawaii, USA, Dec. 2003.

[10] C. G. Cassandras and W. Li, "Sensor Networks and Cooperative Control", European Journal of Control, Vol. 11, pp. 436–463, 2005.

[11] J. Lygeros, K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry, "Dynamical Properties of Hybrid Automaton," *IEEE Transactions on Automatic Control*, vol. 48, no 1, Jan. 2003.

[12] T.A. Henzinger. "The theory of hybrid automata," *In Proc. of LICS: Logic in Computer Science*, IEEE Computer Society Press, 1996, pp. 278-292.

[13] J. Lygeros, C. Tomlin and S. Sastry, *Art of Hybrid Systems,* Compendium of Lecture Notes for the Hybrid Systems Class, 2002.

[14] T.A. Henzinger and V. Rusu. "Reachability verification for hybrid automata", *HSCC 98: Hybrid Systems: Computation and Control, Lecture Notes in Computer Science* 1386, Springer-Verlag, 1998, pp 190–204.

[15] J. Lygeros, K. H. Johansson, S. Sastry, and M. Egerstedt, "On the existence of executions of hybrid automata," in *IEEE Conf. Decision Control*, Phoenix, AZ, 1999, pp. 2249–2254.

[16] K. H. Johansson, "Lecture notes of Hybrid syatems," UC Berkeley, 2000, http://www.s3.kth.se/~kallej/eecs291e/lecture02.pdf.

[17] C.-H. Won, "Comparative Study of Various Control Methods for Attitude Control of a LEO Satellite," *Aerospace Science and Technology*, May, 1999.