Safety Verification of Fault Tolerant Goal-based Control Programs with Estimation Uncertainty

Julia M. B. Braman and Richard M. Murray

Abstract— Fault tolerance and safety verification of control systems that have state variable estimation uncertainty are essential for the success of autonomous robotic systems. A software control architecture called Mission Data System, developed at the Jet Propulsion Laboratory, uses goal networks as the control program for autonomous systems. Certain types of goal networks can be converted into linear hybrid systems and verified for safety using existing symbolic model checking software. A process for calculating the probability of failure of certain classes of verifiable goal networks due to state estimation uncertainty is presented. A verifiable example task is presented and the failure probability of the control program based on estimation uncertainty is found.

I. Introduction

Autonomous robotic missions by nature have complex control systems. The necessary fault detection, isolation and recovery software for these systems is often cumbersome and added on as failure cases are encountered in simulation. One way to systematically incorporate fault tolerance in autonomous robotic control systems is to create a flexible control system that can reconfigure itself in the presence of faults. However, if the control system cannot be verified for safety, even in the presence of state variable estimation uncertainty, the added complexity of the reconfigurability of a system could reduce the system's effective fault tolerance.

A great deal of work to date has focused on detecting and recovering from sensor failures in the control of autonomous systems [1]. One particularly useful way to model a fault tolerant control system is as a hybrid system. The control of hybrid systems has been well studied [2]. When the continuous dynamics of these systems are sufficiently simple, it is possible to verify that the execution of the hybrid control system will not fall into an unsafe regime [3]. There are several software packages available that can be used for this analysis, including HyTech [4], UPPAAL [5], and PHAVer [6], all of which are symbolic model checkers. PHAVer, the symbolic model checker used in this paper, is able to exactly verify linear hybrid systems with piecewise constant bounds on continuous state derivatives and is able to handle arbitrarily large numbers due to the use of the Parma Polyhedra Library. Safety verification for fault tolerant hybrid control systems ensures that the occurrence of certain faults will not cause the system to reach an unsafe state.

However, these verification software packages cannot verify linear hybrid systems that have uncertainty in the estimated state variables involved in transition condition

J. Braman and R. Murray are with the Dept. of Mech. Eng., California Institute of Technology, Pasadena, CA 91125, USA `braman@caltech.edu`

logic. For autonomous systems, none of the state variables used in the control system are known perfectly, and these uncertainties can affect the safety of the system. Stochastic hybrid systems include uncertainty in the transitions of the hybrid automaton as probabilistic transition conditions. Much work has been done on the verification of stochastic hybrid systems. Prajna et al [7] use barrier certificates to bound the upper limit of the probability of failure of the stochastic hybrid system; Kwiatkowska et al [8] discuss a probabilistic symbolic model checking software called PRISM; and Amin et al [9] describe stochastic reachability and maximal probabilistic safe set computations for discrete time stochastic hybrid systems. However, purely probabilistic transition conditions do not model estimation uncertainty well; deterministic transitions with probabilistic components may be a better model.

In this paper, Mission Data System (MDS), developed at the Jet Propulsion Laboratory, is the goal-based control architecture. The structure of this paper is as follows. Section II summarizes important concepts of MDS that pertain to this work and describes the goal network conversion and verification procedure. Section III describes the major contribution of this work, the process for calculating the probability of entering the unsafe set due to estimation uncertainty for certain classes of verifiable goal networks. Section IV describes an example verification and failure probability calculation. Finally, Section V concludes the paper and discusses future directions of research.

II. Background Information

A. State Analysis and Mission Data System

State Analysis is a systems engineering methodology that focuses on a state-based approach to the design of a system [10]. Models of state effects in the system to be controlled are used for such things as the estimation of state variables, control of the system, planning, and goal scheduling. State variables are representations of states or properties of the system that are controlled or that affect a controlled state. Examples of state variables could include the position of a robot, the temperature of the environment, the health of a sensor, or the position of a switch.

Goals and goal elaborations are created based on the models. Goals are specific statements of intent used to control a system by constraining a state variable in time. Goals are elaborated from a parent goal based on the intent and type of goal, the state models, and several intuitive rules, described in [10]. A core concept of State Analysis is that the language used to design the control system should be nearly

the same as the language used to implement the control system. Therefore, the software architecture, MDS, is closely related to State Analysis.

Goal networks replace command sequences as the control input to the system. A goal network consists of a set of goals with their associated starting and ending time points and temporal constraints. A goal may cause other constraints to be elaborated on the same state variable and/or on other causally related state variables. The goals in the goal network and their elaborations are scheduled by the scheduler software component so that there are no conflicts in time, goal order or intent. Each scheduled goal is then achieved by the estimator or controller of the state variable constrained.

Elaboration allows MDS to handle tasks more flexibly than control architectures based on command sequences. One example is fault tolerance. Re-elaboration of failed goals is an option if there are physical redundancies in the system, many ways to accomplish the same task, or degraded modes of operation that are acceptable for a task. The elaboration class for a goal can include several pre-defined tactics. These tactics are simply different ways to accomplish the intent of the goal, and tactics may be logically chosen by the elaborator based on programmer-defined conditions. This capability allows for many common types and combinations of faults to be accommodated automatically by the control system [11].

B. Goal Network Conversion and Verification Procedure

Hybrid system analysis tools can be used to verify the safe behavior of a hybrid system; therefore, a procedure to convert goal networks into hybrid systems is an important tool for goal network verification. A process for converting certain types of goal networks is described in [12]. These goal networks can have several state variables and several layers of goal elaborations, however time points must be well-ordered, which means that the time points fire in the order they are listed in the elaboration.

Each state variable in the goal network is labeled as either controllable, uncontrollable, or dependent. A controllable state variable (CSV) is directly associated with a command class. An uncontrollable state variable (USV) is not associated with a command class in any way. A dependent state variable (DSV) has model dependencies on at least one controllable state variable. Different hybrid automata are created from goals on and states of these different types of state variables.

An outline of the conversion procedure for the goals on CSVs and DSVs is as follows:

1) Create elaboration and transition logic tables for each goal that elaborates any constraints on CSVs and for each CSV and continuous DSV, respectively. List transition conditions between states for each discrete DSV.
2) Place goals between consecutive time points into groups.
3) In each group, create locations (modes) by combining branch goals (goals on CSVs that are not ancestors of other goals on CSVs in the group) with all parent and sibling goals (goals in the same tactic or other root goals) that constrain CSVs. Label each location with the dynamical update equations for all CSVs and continuous DSVs constrained in the location. Create Success and Safing locations.
4) Create transitions between locations and groups using the elaboration and transition logic tables found above. Elaboration logic controls transitions into groups and failure transitions between locations in a group, and transition logic controls the transitions out of a group to the next group or to the Success location.
5) Add exit and failure transitions based on time to locations in groups that have time constraints. Add entry actions that reset the time variable to zero when transitions from the group connector into locations in these groups are taken.
6) Remove unnecessary locations, groups, and transitions.

For each each USV, a separate hybrid automaton is created by making locations from the discrete states or discrete sets of states of the variable. The transition conditions are stochastic rates or are based on the state models. For safety verification, the hybrid automata are converted into PHAVer code and the appropriate transitions are synchronized between the automata. The unsafe (or incorrect) set is determined and conditions that would cause the hybrid automata to enter the unsafe set are searched for using the verification software. If no such conditions are found, the goal network is said to be verified.

III. Failure Probability Derivation

An important assumption for the verification of the control programs described in the previous section is that the values of the state variables are known exactly. This is unrealistic in autonomous systems, where the estimation of the states of the system are bound to have some uncertainty. In this section, a method for calculating the probability of failure of a verifiable goal network is described for a subset of the goal networks allowed in the previous section.

In addition to a goal network being verifiable without estimation uncertainty, some restrictions must be placed on the state variables that can be uncertain. Any uncontrollable state variable may have bounded estimation uncertainty. All uncontrollable state variables take values that fall into discrete sets. In this work, each combination of discrete values that the uncertain state variables can take must be unambiguously associated with only one location in each group of the hybrid automaton. Stated another way, the location in a group that the automaton transitions into after the initial transition into the group must only depend on the values that the uncertain state variables take. Finally, it is assumed that the probability of transitions of the uncertain state variable from one value to another (or the same) value only depends on the previous value of that state variable, and therefore can be modeled as a stationary Markov process. The probability that the estimated value of an uncertain state

variable is derived from the estimation uncertainty, and only depends on the actual value of the state variable.

Several pieces of information that characterize the uncertain state variables are assumed to be known. First, the number of state variables that are uncertain is n and the number of discrete states or sets of states that each uncertain state variable can take is m_i, where $i = 1, ..., n$. Let j_i represent the jth possible value of the ith uncertain state variable, where $j_i = 1, ..., m_i, \forall i = 1, ..., n$. Let v_{ei} represent the estimated value of the ith uncertain state variable and let v_{ai} represent the actual value of each uncertain state variable. The Markov transition probabilities for the actual values of the uncertain state variables are also given, and the stationary probabilities represent the probability that each state variable's actual value is each of its possible values. Finally, the estimation uncertainty of each state variable is the probability distribution of the estimated value of the state variable being $j_{i,e}$ given that the actual value of the state variable is $j_{i,a}$.

A. Problem Set-up

Let S be the set of all possible combinations of actual and estimated values that each uncertain state variable can take; in Fig. 1, S is the overall closed set. S has $\prod_{i=1}^{n} m_i^2$ elements, and each element takes the form $v_{a1} v_{e1} v_{a2} v_{e2} ... v_{an} v_{en}$. For each group g_k, where $k = 1, ..., N$ and N is the number of groups in the automaton, there are sets $\Omega_k \subset S$, where the elements of Ω_k cause the automaton to enter the unsafe set from g_k and are called "unsafe" elements. Since the goal network is verifiable, entrance into this set is always due to estimation uncertainty, and the probability of entering this set is the failure probability. Entrance into the unsafe set for each group is dictated by the actual values of the uncertain state variables, while the transition conditions within the group are dependent on the estimated value of the uncertain state variable. There also are sets $F_k \subset S$ in which each element causes the automaton to leave g_k and enter the Safing location without also entering the unsafe set. A set F_k may be empty. Finally, there are sets

$$\Xi_k \equiv S - (\Omega_k + F_k) \tag{1}$$

to which the remainder of elements in S belong. Elements in Ξ_k, called "non-failing" elements, allow operation to continue in the group or allow a successful transition out of the group to the next one. These sets are represented graphically in Fig. 1.

Some groups may be broken into two or more subgroups. A subgroup of locations is a proper subset of locations in the group that, once entered from the previous group (or initially), is not exited until a transition out of the group is taken (to Safing or to the next group). Subgroups are treated like groups; adjustments to the numbering of the groups are needed to account for them.

For each location $l_h, h = 1, ..., \lambda_k$, where λ_k is the number of locations in each group (or subgroup), there is a set I_{kh} that contains all of the initial conditions that will cause
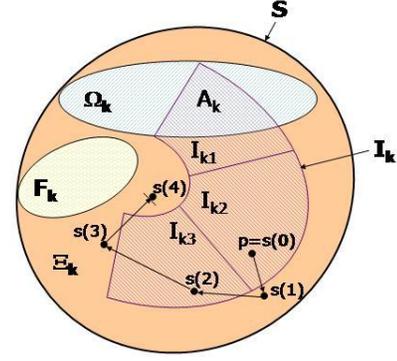
Fig. 1. A representation of sets of uncertain state variable elements. Set S is the sum of the set of unsafe elements, Ω_k, the set of "Safing" elements, F_k, and the set of non-failing elements, Ξ_k. The set of initial conditions, $I_k \subseteq S \backslash F_k$, is the union of the sets of initial conditions for each of the locations in the group, $I_{kh}, h = 1, 2, 3$ in this case. Set $A_k = I_k \cap \Omega_k$ is the set of unsafe initial conditions. A representative path starting at $s(0) = p$ that the automaton takes through the sets in this group, with completion time $c_k = 5$, is shown.

the automaton to enter that location when entering g_k. The intersection of I_{kh} for any two locations in g_k is the empty set and the set of initial conditions of g_k is the union of all the sets of initial conditions for each of its locations,

$$I_k = \bigcup_{h=1}^{\lambda_k} I_{kh}. \tag{2}$$

If g_k is a subgroup, I_k will be a proper subset of set $S \backslash F_k$; however if g_k is a group, I_k will equal $S \backslash F_k$. A representation of set $I_k \subset S \backslash F_k$ with $\lambda_k = 3$ is shown in Fig. 1.

For each group, the initial conditions that would cause immediate failure of the group due to estimation uncertainty can be found in set $A_k = I_k \cap \Omega_k$, which is shown graphically in Fig. 1. The probability of failing due to the initial condition can be found by summing the probability of each element of A_k occurring, $a_k = \sum_{b=1}^{b_k} P(s(0) = x_b)$, where b_k be the number of elements in set A_k and x_b is the bth element in A_k.

$$a_k = \sum_{b=1}^{b_k} \prod_{i=1}^{n} P(v_{ai} = j_{i,ab}) P(v_{ei} = j_{i,eb} | v_{ai} = j_{i,ab}) \tag{3}$$

For the elements in $I_k \backslash A_k$ for each group, the vector W_k contains the probability of starting in each non-failing initial condition. Let p_k be the number of elements in $I_k \backslash A_k$ and $W_k(p)$ be the pth element of W_k, where $p = 1, ..., p_k$ and $W_k(p) = P(s(0) = x_p)$.

$$W_k(p) = \prod_{i=1}^{n} P(v_{ai} = j_{i,ap}) P(v_{ei} = j_{i,ep} | v_{ai} = j_{i,ap}) \tag{4}$$

For each group, there is a $q_k \times q_k$ matrix, Q_k, whose elements are the probability of making a transition from element $q \in \Xi_k$ to element $q' \in \Xi_k$, where $q, q' = 1, ..., q_k$, and $Q_k(q, q') = P(s(r + 1) = x_{q'} | s(r) = x_q)$ and q_k is the

number of elements in set Ξ_k.

$$Q_k(q,q') = \prod_{i=1}^{n}(P(v_{ai} = j_{i,aq'}|v_{ai,p} = j_{i,aq})\times$$
$$P(v_{ei} = j_{i,eq'}|v_{ai} = j_{i,aq'})), \quad (5)$$

where $v_{ai,p}$ is the previous actual value of the ith uncertain state variable.

For each group, there exists a $q_k \times 1$ vector, $W_{u,k}$, whose elements are the sum of probabilities of the transitions from the qth element in Ξ_k to each element y_u in Ω_k, which is the transition from each non-failure state to any failure state, or $W_{u,k}(q) = \sum_{u=1}^{u_k} P(s(r+1) = y_u|s(r) = x_q)$. Let u_k be the number of unsafe elements in Ω_k and $q = 1, ..., q_k$, then

$$W_{u,k}(q) = \sum_{u=1}^{u_k}\prod_{i=1}^{n}(P(v_{ai} = j_{i,au}|v_{ai,p} = j_{i,aq})\times$$
$$P(v_{ei} = j_{i,eu}|v_{ai} = j_{i,au})). \quad (6)$$

If set $I_k\backslash A_k \subset \Xi_k$, then there are two more definitions to make, $W_{uI,k}$ and $Q_{I,k}$, which have probabilities of transitions from elements in $I_k\backslash A_k$ to every element in Ω_k and each element in Ξ_k, respectively. Let $p = 1, ..., p_k$ represent the pth initial condition in $I_k\backslash A_k$,

$$W_{uI,k}(p) = \sum_{u=1}^{u_k}\prod_{i=1}^{n}(P(v_{ai} = j_{i,au}|v_{ai,p} = j_{i,ap})\times$$
$$P(v_{ei} = j_{i,eu}|v_{ai} = j_{i,au})), \quad (7)$$

and

$$Q_{I,k}(p,q) = \prod_{i=1}^{n}(P(v_{ai} = j_{i,aq}|v_{ai,p} = j_{i,ap})\times$$
$$P(v_{ei} = j_{i,eq}|v_{ai} = j_{i,aq})). \quad (8)$$

Each group g_k has a minimum execution time that could be due to the completion of the goal constraints or to a specific time constraint applied to the group. This minimum execution time is called the completion time and is labeled c_k. There are only three possible ways of leaving a group:

1) The completion time is reached, and the execution moves normally into the next group.
2) A transition to the "Safing" set, $s(r) \in F_k$, is taken before the completion time is reached.
3) A transition to the unsafe set, $s(r) \in \Omega_k$, is taken before the completion time is reached.

Only the last condition is considered to be a failure case that would contribute to the failure probability of that group. There is a set of execution paths of a group, U_k, that end with a transition into the unsafe set and each of these paths has some probability of occurring. The failure probability of the group, $W_s(k)$, is the sum of the probabilities of all paths $\sigma \in U_k$.

B. Uniform Completion Case

In the uniform completion case, the minimum execution time of a group is also the maximum execution time; in other words, all locations in the group contribute an equal amount to the completion of the goals or to the time constraint placed on the group. Therefore, a uniform completion time c_k would require exactly c_k execution time steps before the group would be exited normally. An example of this execution path with $c_k = 5$ is shown in Fig. 1. Therefore, there are only c_k paths in U_k, each consisting of a failure after each step up until the completion time.

For $c_k = 1$, the only way to reach the unsafe set is to start in it, the probability of which is represented by a_k. For $c_k = 2$, the probability of starting in the non-failure initial conditions and then making the transition to the unsafe set is added to the probability of starting in the unsafe set, and so on. The equations for the failure probability for different cases of c_k, where $k = 1, ..., N$ are

$$W_s(k) = a_k, \quad c_k = 1, \quad (9)$$
$$W_s(k) = a_k + W_k \cdot W_{uI,k}, \quad c_k = 2, \quad (10)$$

and for $c_k \in [3, \infty)$,

$$W_s(k) = a_k + W_k \cdot \left[W_{uI,k} + \sum_{x=0}^{c_k-3} Q_{I,k}Q_k^x W_{u,k}\right]. \quad (11)$$

If $Q_{I,k} = Q_k$ and $W_{uI,k} = W_{u,k}$, the $c_k \in [2, \infty)$ equation is

$$W_s(k) = a_k + W_k \cdot \left[\sum_{x=0}^{c_k-2} Q_k^x W_{u,k}\right]. \quad (12)$$

For the special case when $c_k = \infty$, the equation for the failure probability is

$$W_s(k) = a_k + W_k \cdot \left[W_{uI,k} + Q_{I,k}(I - Q_k)^{-1} W_{u,k}\right], \quad (13)$$

or, if $W_{uI,k} = W_{u,k}$ and $Q_{I,k} = Q_k$,

$$W_s(k) = a_k + W_k \cdot (I - Q_k)^{-1} W_{u,k}, \quad (14)$$

given that

$$(I - Q_k)^{-1} = \sum_{x=0}^{\infty} Q_k^x. \quad (15)$$

C. Non-Uniform Completion Case

In the non-uniform completion case, the minimum execution time of a group is not the same as the maximum execution time. Sets of locations contribute a different amount to the completion of goals or time constraints on the group. An example of this is a group that cannot be transitioned out of normally until the robot covers some distance, however, different locations in this group constrain the maximum speed of the robot to different values. The set of locations with the maximum speed limit have a contribution to the goal completion of 1; that is, this set of locations dictates the minimum execution time of the group. All other sets of locations with lower speed limit constraints have contribution values that are less than one. Paths in this group that exit the group normally due to goal completion have location
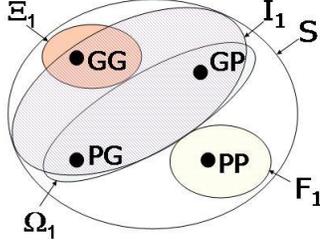
Fig. 2. A representation of the sets of uncertain state variable elements for the single uncertain state variable example.

contribution value sums that are equal to or exceed the completion time.

While all normal completion paths in the uniform completion case had the same path length and the same contribution value pattern (a sum of all ones), in the non-uniform completion case, the normal completion paths have different lengths and location contribution value patterns. Therefore, the number of ways to reach group completion, and likewise the unsafe set, is much greater in this case than in the uniform completion case. The failure probability, $W_s(k)$, for this problem is calculated in the same way as in the uniform completion case, by summing the path probabilities of each path $\sigma \in U_k$, however the determination of these paths is a more difficult problem. This problem is solvable and is addressed in [13].

IV. EXAMPLES

A. Single Uncertain State Variable Example

Consider a verifiable goal network that has one uncertain state variable that can take on two discrete values, GOOD and POOR (G and P, respectively). The set S consists of four elements that all have the form $v_{a1}v_{e1}$, $S = \{GG, GP, PG, PP\}$; one element is in the "Safing" set $F_1 = \{PP\}$, two are in the unsafe set $\Omega_1 = \{GP, PG\}$, and one is in the non-failing completion set $\Xi_1 = \{GG\}$. As shown in Fig. 2, the set of initial conditions is $I_1 = S \backslash F_1 = \{GG, GP, PG\}$.

The set of unsafe initial conditions is the intersection of the set of initial conditions and the unsafe set, $A_1 = I_1 \cap \Omega_1 = \{GP, PG\}$. Therefore, the calculation of the probability of failure due to starting in the unsafe set is

$$a_1 = P(s(0) = GP) + P(s(0) = PG). \quad (16)$$

The initial condition that is non-failing is $I_1 \backslash A_1 = \{GG\}$, so the vector of the probabilities of starting in a non-failing state is a scalar in this case, and is

$$W_1 = P(s(0) = GG). \quad (17)$$

Likewise, since there is only one element in Ξ_1, the vector of transition probabilities from non-failing elements in Ξ_1 to any of the unsafe elements in Ω_1 is also a scalar in this case, and is

$$W_{u,1} = P(s(r+1) = GP | s(r) = GG) + $$
$$P(s(r+1) = PG | s(r) = GG). \quad (18)$$
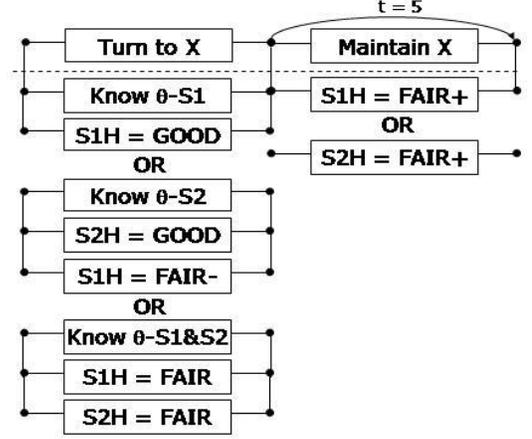
Fig. 3. Goal network for precision orientation device example; S1 = sensor 1, S2 = sensor 2, S1H = sensor 1 health, S2H = sensor 2 health. The dashed line indicates that the goals below it are elaborated from the goal above the line. The curved arrow indicates a time constraint.

The transition matrix between non-failing elements in Ξ_1 is again a scalar in this case, and is

$$Q_1 = P(s(r+1) = GG | s(r) = GG). \quad (19)$$

Since there is only one element in Ξ_1, this is a uniform completion case. Once given the value of c_1, an equation can be chosen (either (9) or (12)) and used to find the probability of reaching the unsafe set due to estimation uncertainty in this group.

B. Precision Orientation Device Example

An example for the uniform completion case involves two sensors that are used to estimate the orientation for a precision orientation device. The first sensor is more accurate, reliable, and has less estimation uncertainty than the second sensor; therefore, when turning, the first sensor is used if its health is estimated to be 'GOOD.' If the first sensor's health is estimated to be 'FAIR' or lower and the second sensor's health is estimated to be 'GOOD,' the second sensor is used to estimate the orientation. If both sensors' healths are estimated to be 'FAIR,' both are used to estimate the orientation. Otherwise, the goal network safes, stopping the turning mechanism.

It takes five time steps in any combination of the turning goal's tactics, shown in Fig. 3, to complete the turn. The goal network then transitions into a maintenance goal on the orientation that is also active for five time steps before moving onto the next goal.

The breakdown of state variable types is one controllable state variable, the orientation state variable; two uncontrollable state variables, the sensor health state variables; and no dependent state variables. The hybrid automaton created from the goals on the orientation state variable is shown in Fig. 4 and the automata created from Sensor 1 Health and Sensor 2 Health state variables are shown in Fig. 5. The unsafe set for this system is the union of several states, listed as follows:
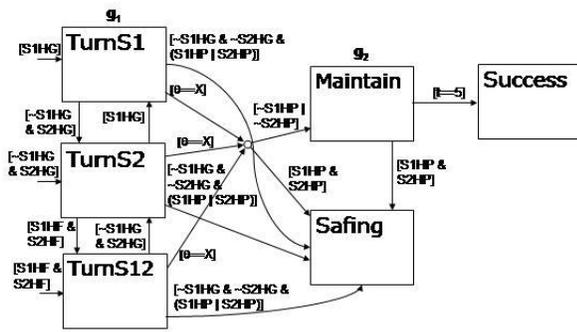
Fig. 4. Hybrid automaton for the CSV, orientation; S1HG = sensor 1 health is GOOD, S1HF = sensor 1 health is FAIR, S1HP = sensor 1 health is POOR, and likewise for sensor 2.

Fig. 5. Hybrid automaton for Sensor 1 Health (Sensor 2 Health) state variable.

1) Orientation in TurnS1 and $S_1 > \alpha$
2) Orientation in TurnS2 and $S_2 > \gamma$
3) Orientation in TurnS12 and $S_1 > \alpha + \beta$ or $S_1 < \alpha$ or $S_2 > \gamma + \delta$ or $S_2 < \gamma$
4) Orientation in Maintain and $S_1 > \alpha + \beta$ and $S_2 > \gamma + \delta$.

The hybrid system was successfully verified for safety over all values of α, β, γ, and δ.

The failure probabilities for each group were calculated for several values of estimation uncertainty and one choice of stationary Markov transition probabilities, and the results can be found in Fig. 6.

V. Conclusions and Future Work

This paper derives the failure probability of certain verifiable goal networks due to state variable estimation uncertainty. An example goal network was verified and its failure probability given the estimation uncertainty was calculated. The calculation of the failure probability for the different groups of a goal network can be used as a verification of the
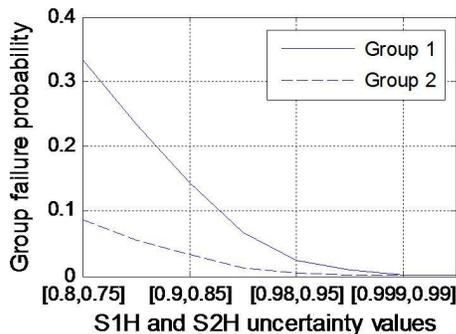
Fig. 6. Calculated group failure probabilities vs. sensor health uncertainty values

goal network in the presence of estimation uncertainty as well as a design tool to drive the design of goal networks or the choice of sensors and estimators to reduce the probability of failure.

Future work includes extending the calculation of a failure probability to include other types of uncertainty. Uncertainty in group transitions, controllable state variables, completion times, and the given probability distributions are all examples of possible types of uncertainty to include. The verification of goal networks in the presence of different forms of uncertainty, including estimation uncertainty, is an important problem, and this approach seems promising, at least for certain types of goal networks.

VI. Acknowledgements

References

[1] Z.-H. Duan, Z.-X. Cai, and J.-X. Yu, "Fault diagnosis and fault tolerant control for wheeled mobile robots under unknown environments: A survey," *IEEE Int'l Conference on Robotics and Automation*, pp. 3428–3433, 2005.

[2] G. Labinaz, M. M. Bayoumi, and K. Rudie, "A survey of modeling and control of hybrid systems," *Annual Reviews of Control*, 1997.

[3] R. Alur, T. Henzinger, and P.-H. Ho, "Automatic symbolic verification of embedded systems," *IEEE Transactions on Software Engineering*, vol. 22, no. 3, pp. 181–201, 1996.

[4] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi, "HyTech: A model checker for hybrid systems," *International Journal on Software Tools for Technology Transfer*, 1997.

[5] K. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *International Journal on Software Tools for Technology Transfer*, vol. 1, no. 1-2, pp. 134–152, 1997.

[6] G. Frehse, "PHAVer: Algorithmic verification of hybrid systems past HyTech," *International Conference on Hybrid Systems: Computation and Control*, 2005.

[7] S. Prajna, A. Jadbabaie, and G. J. Pappas, "Stochastic safety verification using barrier certificates," *IEEE Conference on Decision and Control*, 2004.

[8] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic symbolic model checking with PRISM: a hybrid approach," *Int J Software Tools Technology Transfer*, vol. 6, pp. 128–142, 2004.

[9] S. Amin, A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Reachability analysis for controlled discrete time stochastic systems," *International Conference on Hybrid Systems: Computation and Control*, pp. 49–63, 2006.

[10] M. Ingham, R. Rasmussen, M. Bennett, and A. Moncada, "Engineering complex embedded systems with State Analysis and the Mission Data System," *AIAA Journal of Aerospace Computing, Information and Communication*, vol. 2, pp. 507–536, December 2005.

[11] R. D. Rasmussen, "Goal-based fault tolerance for space systems using the Mission Data System," *IEEE Aerospace Conference Proceedings*, vol. 5, pp. 2401–2410, March 2001.

[12] J. M. Braman, R. M. Murray, and D. A. Wagner, "Safety verification of a fault tolerant reconfigurable autonomous goal-based robotic control system," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.

[13] J. M. Braman and R. M. Murray, "Failure probability of verifiable goal-based control programs due to state estimation uncertainty." Submitted, IEEE Conference on Decision and Control, 2008.