2011 American Control Conference
on O'Farrell Street, San Francisco, CA, USA
June 29 - July 01, 2011

Collision Avoidance System Optimization with Probabilistic Pilot Response Models

James P. Chryssanthacopoulos and Mykel J. Kochenderfer

Abstract— **All large transport aircraft are required to be equipped with a collision avoidance system that instructs pilots how to maneuver to avoid collision with other aircraft. Uncertainty in the compliance of pilots to advisories makes designing collision avoidance logic challenging. Prior work has investigated formulating the problem as a Markov decision process and solving for the optimal collision avoidance strategy using dynamic programming. The logic was optimized to a pilot response model in which the pilot responds deterministically to all alerts. Deviation from this model during flight can degrade safety. This paper extends the methodology to include probabilistic pilot response models that capture the variability in pilot behavior in order to enhance robustness.**

I. Introduction

The Traffic Alert and Collision Avoidance System (TCAS), carried by all large transport aircraft, has been shown to significantly improve safety in the event of failure in the air traffic control system. TCAS does not control the aircraft directly; it can only issue advisories to pilots on how to maneuver vertically to prevent collision. As recorded radar data have shown, there is significant variability in the delay and strength of the response of pilots to these advisories [1]. In order to ensure safety, the design of the collision avoidance logic should accommodate the variability in response.

The current TCAS logic does not explicitly model variability in pilot response. Instead, it uses a deterministic model to predict the future paths of the aircraft and applies a complex set of heuristics in an attempt to provide robustness to unexpected behavior. This paper pursues a more principled approach that uses a probabilistic model to account for pilot response variability. The approach builds upon prior work on framing the collision avoidance problem as a Markov decision process (MDP) and solving for the optimal collision avoidance strategy using dynamic programming (DP) [2]–[5]. This paper shows that systems that incorporate probabilistic response models are significantly more robust than prior systems that assumed a deterministic response.

The next section provides a brief overview of MDPs. Section III describes the approximation method used in this work. Section IV formulates the collision avoidance problem as an MDP. Section V demonstrates the usage of the logic. Section VI presents simulation results. Section VII concludes and outlines further work.

This work is sponsored by the Federal Aviation Administration under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

The authors are with Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Massachusetts, 02420. Email addresses: chryssanthacopoulos@ll.mit.edu; mykelk@ll.mit.edu

II. Markov Decision Processes

This section reviews finite-horizon Markov decision processes (MDPs) [6]. MDPs model sequential decision-making problems where cost is to be minimized. An MDP is defined by the tuple $(\mathscr{S}, \mathscr{A}, C, T)$. The sets $\mathscr{S}$ and $\mathscr{A}$ are a finite set of states and a finite set of actions, respectively. The cost function $C(s,a)$ is the immediate cost when taking action a in state s, and $C(s)$ is the cost when terminating in state s. Both $C(s,a)$ and $C(s)$ are assumed deterministic. The state-transition function $T(s,a,s')$ is the probability of transitioning from state s to state s' when taking action a.

A deterministic policy $\pi_k(s)$ indicates what action should be taken for each of the states s in $\mathscr{S}$ when there are k steps remaining until termination. An optimal policy π_k^* is typically found by first computing the optimal state-action cost-to-go function J_k^* and then extracting the greedy policy. The optimal state-action cost-to-go function $J_k^*(s,a)$ is the expected cost incurred when starting in state s, k steps from termination, taking action a for one time step, and then continuing with the actions prescribed by π^* for the remainder of time. It obeys the following recursion:

$$J_k^*(s,a) = C(s,a) + \sum_{s' \in \mathscr{S}} T(s,a,s') \min_{a'} J_{k-1}^*(s',a'). \tag{1}$$

An optimal policy with respect to J_k^* satisfies

$$\pi_k^*(s) = \arg\min_a J_k^*(s,a). \tag{2}$$

Several different dynamic programming algorithms exist for iteratively arriving at the optimal cost-to-go function. An algorithm known as value iteration, for example, generates a sequence of state-action cost-to-go functions $J_0^*, J_1^*, \ldots, J_K^*$ by successively applying Equation 1 with $J_0^*(s,a) = C(s)$ for all state-action pairs.

III. Approximation Method

Because the collision avoidance domain contains variables that are naturally continuous, such as altitudes and vertical rates, it is necessary to discretize the state space. Discretizing the state space and directly solving for the optimal policy using value iteration can be infeasible for problems with many state variables. A new approximation method was introduced in [4] to solve a certain class of MDPs where only a subset of the state variables is controllable. It is applicable to TCAS-like systems that only affect the vertical motion of the aircraft.

The approach involves decomposing the full problem into two subproblems, one involving the controlled state space $\mathscr{S}_c$ and the other involving the uncontrolled state space $\mathscr{S}_u$.

Dynamic programming is used to estimate the time τ until the uncontrolled state s_u enters a certain subspace, denoted G, of the uncontrolled state space. The entry time τ is represented as a discrete random variable; the probability that s_u enters G in k steps is denoted $D_k(s_u)$. The probability distribution for τ is computed up to a certain horizon K and the remaining probability mass, denoted $D_{\bar{K}}(s_u)$, represents the probability of the event $\{\tau > K\}$. The entry distributions are computed offline for every uncontrolled state and saved as tables $D_0, D_1, \ldots, D_K$ in memory.

The controlled subproblem is treated as a finite-horizon MDP where τ indicates the number of steps until termination. The solution of the controlled subproblem is a sequence of cost-to-go functions for the controlled state space for every possible value of τ up to the horizon K. The cost-to-go $J_k^*(s_c, a)$ represents the lowest expected cost incurred starting from controlled state s_c and taking action a for one time step assuming s_u enters G in exactly k steps. It is computed recursively using value iteration. The cost-to-go for $\{\tau > K\}$, $J_{\bar{K}}^*(s_c, a)$, is computed by initializing $J_0^*(s_c, a) = 0$ for all state-action pairs and iterating Equation 1 until horizon K.

Let $\pi_k^*(s_c)$ represent the optimal action to take from s_c given $\tau = k$:

$$\pi_k^*(s_c) = \arg\min_a J_k^*(s_c, a). \tag{3}$$

This policy will be examined further in Section IV. The tables $J_0^*, J_1^*, \ldots, J_K^*, J_{\bar{K}}^*$ are computed offline for every controlled state and stored in memory.

The solutions to the subproblems are combined to form an approximation to the joint (controlled and uncontrolled) optimal state-action cost-to-go function:

$$J^*(s,a) = D_{\bar{K}}(s_u) J_{\bar{K}}^*(s_c, a) + \sum_{k=0}^{K} D_k(s_u) J_k^*(s_c, a). \tag{4}$$

Similar to Equation 3, the joint optimal policy is given by

$$\pi^*(s) = \arg\min_a J^*(s,a). \tag{5}$$

Further details regarding the algorithm, including the assumptions under which it is applicable, can be found in [4].

IV. Collision Avoidance Modeling

This section formulates the collision avoidance problem as an MDP. In the collision avoidance problem, one aircraft equipped with a collision avoidance system, called the own aircraft, encounters an unequipped aircraft, called the intruder aircraft. The system on the own aircraft can issue resolution advisories to the pilots advising them to adjust their vertical rate to avoid conflict with the intruder aircraft. Conflict in this paper occurs when the intruder comes within 500 ft horizontally and 100 ft vertically. This has been called a near mid-air collision (NMAC) in prior TCAS studies [7].

The approximation method of Section III can be applied by observing that the horizontal and vertical motions can be decoupled. The set of all states describing vertical motion is the controlled state space, and the set of all states describing horizontal motion is the uncontrolled state space. The subspace G is the set of all states in which the aircraft are in conflict horizontally. The entry time τ becomes the time until horizontal conflict. The cost function is designed so that a high penalty is incurred when both $\tau = 0$ (horizontal conflict) and vertical separation is less than 100 ft (vertical conflict).

A. Resolution Advisories

The system on the own aircraft can issue one of two different initial advisories: climb at least 1500 ft/min or descend at least 1500 ft/min. The pilot response is modeled as a $1/4$ g acceleration to meet the target rate, if necessary. Following the initial advisory, the system can either terminate, reverse, or strengthen the advisory. The pilot responds to a reversal by applying a $1/3$ g acceleration to reach a 1500 ft/min vertical rate in the direction opposite the original advisory. The pilot response to a strengthening, similarly, consists of a $1/3$ g maneuver to achieve a vertical rate of 2500 ft/min in the direction of the previous advisory. After an advisory has been strengthened, it can be weakened to reduce the required vertical rate to 1500 ft/min. It is responded to with $1/3$ g acceleration. The various advisories, as well as the decision to not alert, define the action set $\mathscr{A}$.

B. Aircraft Dynamic Model

The aircraft trajectories are perturbed by zero-mean Gaussian accelerations horizontally and vertically. The dynamic state of the aircraft is captured using seven variables:

- h: altitude of the intruder aircraft relative to own,
- $\dot{h}_0$: vertical rate of the own aircraft,
- $\dot{h}_1$: vertical rate of the intruder aircraft,
- s_{RA}: state of the resolution advisory,
- r: horizontal range to the intruder,
- r_v: relative horizontal speed, and
- θ_v: difference in the direction of the relative horizontal velocity and the bearing of the intruder.

The intruder aircraft experiences random vertical accelerations drawn from a zero-mean Gaussian with $3\,\text{ft/s}^2$ standard deviation. The own aircraft, moreover, undergoes the same process except when a resolution advisory is actively followed, in which case the own aircraft accelerates according to the model of Section IV-A. In the horizontal plane, both aircraft move independently in response to random accelerations selected from a zero-mean Gaussian with $30\,\text{ft/s}^2$ standard deviation.

The variable s_{RA} is a discrete random variable that is used to model the pilot response. The dynamics of s_{RA} are given by a controlled Markov chain, which is discussed further in Section IV-C.

The continuous state variables are discretized according to a standard grid-based scheme that uses cut points along each of the dimensions of the state space. The controlled state space $\mathscr{S}_c$ is comprised of all discrete states of the form $(h, \dot{h}_0, \dot{h}_1, s_{\text{RA}})$. The discrete transition probabilities of the Markov chain over the controlled state space are estimated using the continuous vertical motion model and the Markov chain pilot response model in combination with sigma-point sampling and multilinear interpolation [3]. There are approximately 472,000 discrete controlled states for the linear pilot response model, approximately 1.75 million for the quadratic model. The uncontrolled state space $\mathscr{S}_u$ contains all states of the form (r, r_v, θ_v). The uncontrolled Markov chain model was constructed using the continuous horizontal motion model, sigma-point sampling, and multilinear interpolation. There are approximately 730,000 discrete uncontrolled states.

C. Pilot Response Models

Two models were constructed to capture the dynamics of the s_{RA} variable that encodes pilot response states. Each state in the Markov chain indicates the active advisory and the current response. The state "climb/none," for example, signifies that a climb advisory is currently on display but that the pilot is unresponsive. In the first model, the number of states scales linearly with the number of advisories (13 states in total). The second model scales quadratically with the number of advisories (49 states in total). Figure 1, while not explicitly enumerating all state-action pairs, illustrates some features of the models.

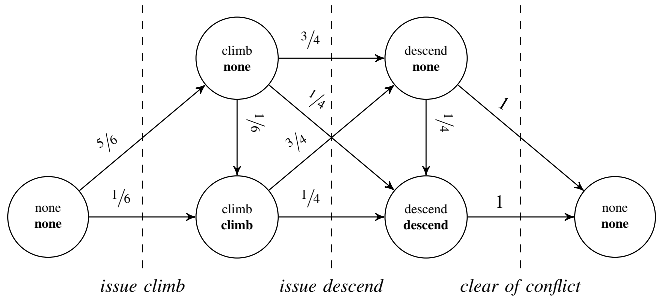
Before an advisory is issued, the Markov chain is in the "none/none" state. Upon the issuance of a climb advisory, for example, the pilot responds immediately with probability $1/6$, transitioning to "climb/climb," and remains unresponsive otherwise, transitioning to "climb/none." Should the climb advisory remain in effect at the next time step, the pilot responds with probability $1/6$ if he has not responded already. For a given advisory, therefore, the response delay follows a geometric distribution; a success probability of $1/6$ was chosen so that, on average, the pilot responds in five seconds.

According to the linear model, if a descend advisory is subsequently issued, the pilot responds with probability $1/4$ and neglects all advisories otherwise, regardless of whether he was responding to the climb advisory. The quadratic model differs in that, if the pilot is responding to the climb advisory, he will continue to respond to the advisory with probability $3/4$. With success probability $1/4$, the pilot will respond to the new advisory (three seconds on average).

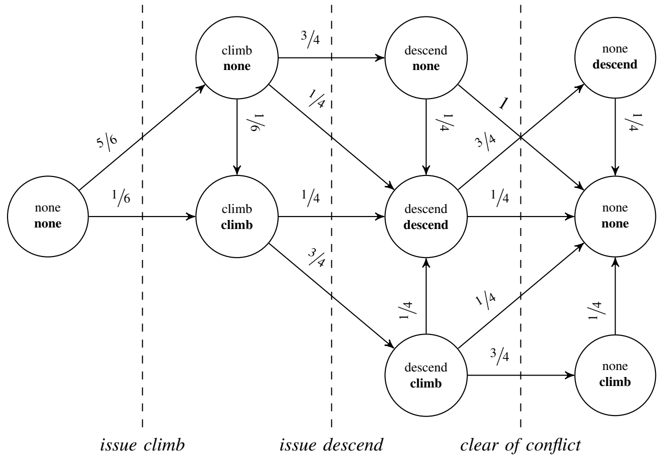
If the advisory is discontinued, the Markov chain transitions to "none/none" with probability one in the linear model. However, in the quadratic model, the pilot retains some memory of the advisory he was previously executing and continues executing it with probability $3/4$ even after the advisory is terminated.

D. Cost Function

Unit cost is accrued when the aircraft come into conflict, i.e., when $|h| < 100\,\text{ft}$ and $\tau = 0$ (s_u enters G). Additionally, to reduce false alerts and course deviation, a small cost of 0.01 is incurred when an alert is initially issued. A cost of 0.009 and of 0.01 are accumulated when the system strengthens and reverses, respectively. A small negative cost of -1×10^{-6} is awarded at every time step in which the system is not alerting to provide some incentive to discontinue alerting after the encounter has been resolved.



(a) Linear model.



(b) Quadratic model.

Fig. 1. Probabilistic pilot response models. The pilot response is in bold, and the active advisory is in regular text. Transition probabilities are indicated along the directed arcs. Self-transitions are omitted. Notice that for several states, transitions out of the state when executing different actions are shown; hence, the transition probabilities need not sum to one.

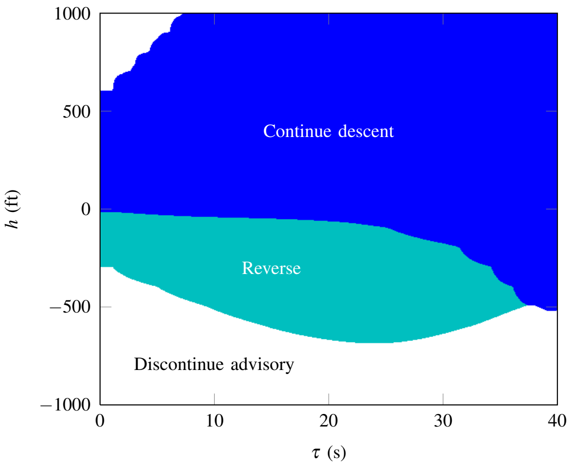
E. Entry Distribution

The entry distribution for all states in the uncontrolled state space was computed using dynamic programming. The subspace G was chosen to be the set of all states for which the horizontal range is less than 1000 ft, twice the size of the horizontal conflict zone that defines an NMAC. Offline computation of the entry time tables for a horizon of $K = 39$ steps required 100 s on a single 3 GHz Intel Xeon core. Storing the tables in memory using a 64-bit floating point representation requires 228 MB.

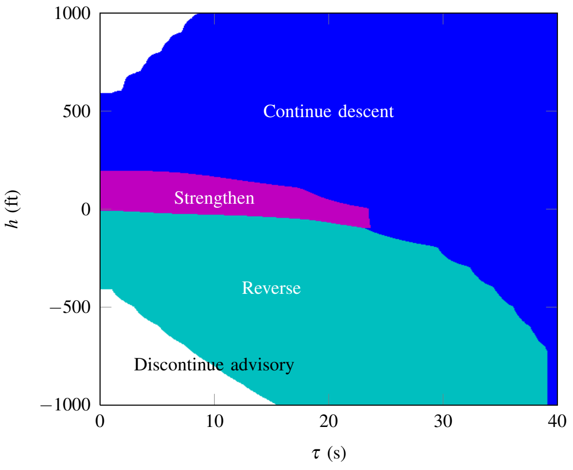
F. Optimal Policy

The cost-to-go tables $J_0^*, J_1^*, \ldots, J_K^*, J_{\bar{K}}^*$ were computed offline in 2.5 min for the linear pilot response model, requiring 151 MB of storage to save only the valid state-action pairs. The quadratic pilot response model required 11.5 min of computation and 560 MB of storage.

While the tables themselves are not immediately informative, it is insightful to inspect the policy $\pi_\tau^*(s_c)$ of Equation 3 for various controlled states and values of τ. Figure 2 depicts the optimal policy as a function of the relative altitude h and entry time τ when both aircraft are flying level, a descend advisory is active, and the pilot is responding to it.



(a) Linear pilot response model.



(b) Quadratic pilot response model.

Fig. 2. Optimal policy slices for $\dot{h}_0 = 0\,\text{ft/min}$, $\dot{h}_1 = 0\,\text{ft/min}$, and $s_{\text{RA}} =$ "descend/descend."

Figure 2(a) shows the optimal policy computed using the linear pilot response model. The blue region indicates the best action is to continue issuing the descend advisory. The descend advisory is typically maintained when the intruder is above the own aircraft (the top half of the plot). However, for large values of τ, the descend advisory is continued even when the own aircraft is above because there is ample time to pass below the intruder. In the teal region, the optimal policy is to reverse the descend to a climb when the intruder is below. This region widens and narrows as τ changes. In certain regions of the state space, depicted in white, the best action is to discontinue the advisory.

Figure 2(b) is the optimal policy computed using the quadratic pilot response model. The policy is similar to that of the first plot. Because in the quadratic model the pilot continues to follow the descend advisory with probability $3/4$ even after it is switched to a climb, the reversal region has expanded to allow additional time for the pilot to reverse and prevent conflict. The reversal region is smaller for the linear model because the pilot will become unresponsive to all advisories with probability $3/4$ when the reversal is issued, which is much safer than continuing the descent.

The purple region in Figure 2(b) marks the places in the state space where the optimal action is to strengthen the descent. This region is absent in the first plot because the response of the pilot to strengthenings, together with the strengthening cost, makes continuing the advisory more advantageous than strengthening. Namely, because the pilot ignores all advisories with probability $3/4$ and responds only with probability $1/4$ after the strengthening is issued, it is unlikely that strengthening can improve safety, especially when conflict is imminent. The probability of responding to the strengthening is also $1/4$ in the quadratic model. However, because the pilot continues his descent with probability $3/4$, the probability of strengthening, though the same, causes strengthening to have a lower expected cost than continuing the descend advisory.

V. Real-time Logic Usage

Once the entry time and cost-to-go tables have been computed offline, they are combined online to choose actions. The optimal action to take from state s in the discrete state space is, as mentioned previously, $\arg\min_a J^*(s,a)$. If the current state $\mathbf{x}$ does not correspond exactly to one of the discrete states, an approximation method such as multilinear interpolation can be used to approximate the state-action cost-to-go function at $\mathbf{x}$. Figure 3 shows an example encounter comparing the behavior of the logic optimized using dynamic programming (DP) with that of the current version of the TCAS logic, Version 7.1. The logic was optimized using the quadratic pilot response model.

The encounter was randomly generated using an encounter model developed using nine months of recorded radar data [8]. This model represents the variables governing the initialization of an encounter as a Bayesian network. The dynamics of the aircraft are represented by a dynamic Bayesian network, which captures the changes in turn rate, vertical rate, and airspeed over time. In this example encounter, although the logic is optimized using a probabilistic pilot response model, the pilot responds to all initial advisories in exactly five seconds and to all subsequent advisories in exactly three seconds.

Determination of the alert to issue at each time step requires knowledge of the state of the resolution advisory, s_{RA}. Because there is uncertainty as to whether the pilot is responding to an advisory, s_{RA} is not completely observable. Instead, a probability distribution over possible values of s_{RA}, called a belief state, must be maintained to summarize the belief regarding the response of the pilot to advisories. As new observations of the aircraft state are made each time step, the belief state $b(s_{\text{RA}})$ is recursively updated using Bayes' rule. The belief state is initialized at $t = 0$ to $b_0(\text{none/none}) = 1$. For all subsequent time steps $t > 0$, the belief state is updated as follows:

$$b_t(s'_{\text{RA}}) \propto p(\dot{h}'_0 \mid \dot{h}_0, s'_{\text{RA}}) \sum_{s_{\text{RA}}} T(s_{\text{RA}}, a, s'_{\text{RA}})\, b_{t-1}(s_{\text{RA}}), \quad (6)$$

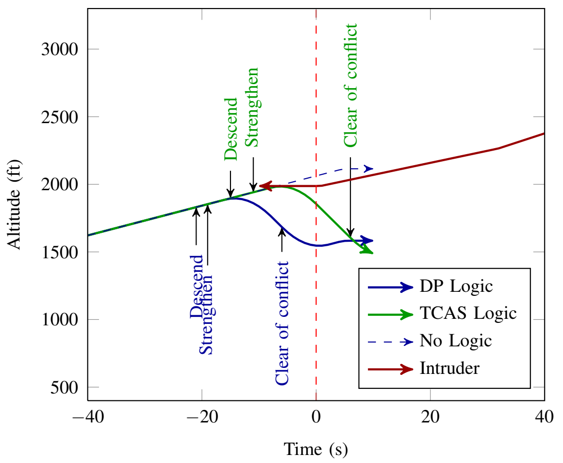


Fig. 3. Example encounter comparing the performance of the DP logic optimized using the quadratic pilot response model with that of TCAS.

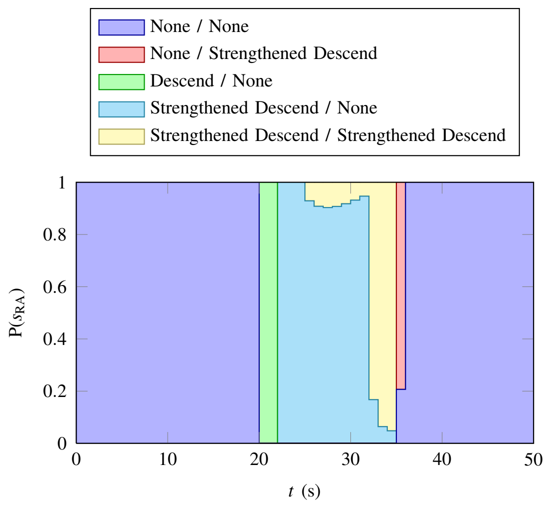


Fig. 4. Advisory response belief state over time for the example encounter.

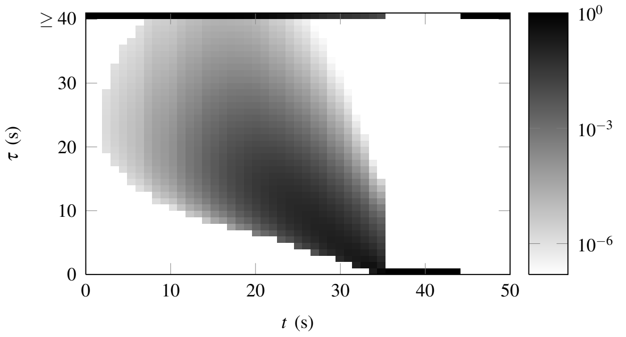


Fig. 5. Entry distribution over time for the example encounter.

where $p(\cdot \mid \dot{h}_0, s'_{RA})$ represents the probability density of the own aircraft vertical rate at time t conditioned on the previously-observed vertical rate, $\dot{h}_0$, and the response state at time t, s'_{RA}. This density is evaluated at the observed vertical rate at the current time, $\dot{h}'_0$. After each belief update, the action to take accounts for the likelihood of different responses using the following approximation [9]:

$$\pi^*(b) = \arg\min_a \sum_s b(s) J^*(s,a). \tag{7}$$

Belief updating using a Bayesian framework is one way of monitoring the progression of the encounter. TCAS, similarly, makes accommodations to handle situations when the own aircraft is moving counter to the current advisory. In such a situation, if certain requirements are met, the advisory will be reversed to prevent potentially dangerous vertical chases. One strength of the current approach, however, is that the introduction of specialized heuristics is unnecessary.

Figure 4 shows the s_{RA} belief state throughout the course of the example encounter. Figure 5 shows the evolution of the entry distribution τ.

Nineteen seconds into the encounter, the DP logic issues a descend to pass below the intruder. The expected cost-to-go for issuing a descend advisory is approximately 0.044, lower than the expected cost-to-go for issuing a climb advisory (0.055) or for not issuing an advisory (0.046). After the descend advisory is issued, as Figure 4 illustrates, the probability distribution over s_{RA} indicates that the own aircraft is not responding to the descend advisory with nearly probability one. With probability on the order of 1×10^{-8}, not shown, the own aircraft is responding to the descend.

Two seconds after the descend advisory is issued, the system strengthens the advisory. At the following time step, the probability distribution over s_{RA} reveals that the own aircraft is, with nearly probability one, not following the strengthened descend advisory. Very small probabilities (on the order of 1×10^{-15} and 1×10^{-13}, respectively) are assigned to the own aircraft following the initial descend advisory and the strengthened descend advisory. As the own aircraft begins to respond to the strengthening three seconds after its issuance, the distribution is updated to reflect this change in pilot behavior. Ten seconds after the pilot begins responding, it is believed that the own aircraft is responding to the strengthening with probability 0.95. The entry distribution falls off quickly as the aircraft come within close proximity laterally.

The DP logic discontinues the advisory at $t = 34$ s, and the belief state over s_{RA} quickly changes in response to the own aircraft leveling off. The aircraft are vertically separated by 441 ft at the point of minimal horizontal separation (368 ft), and an NMAC does not occur.

TCAS also initially issues a descend advisory, six seconds after the DP logic, and then strengthens four seconds later. The alerts come too late, and an NMAC results at $t = 39$ s.

VI. Results

Table I summarizes the results of evaluating the DP logic and TCAS logic on half a million encounters generated by the encounter model. The performance of the DP logic optimized using three different pilot response models—linear, quadratic, and deterministic—was assessed. These three systems, together with TCAS, were tested in three operating environments: one with a deterministic pilot response, one with a linear probabilistic pilot response, and one with a quadratic probabilistic pilot response. Each cell in the table indicates the performance of the systems in a particular operating environment and for a particular metric. The systems appear in the following order within each cell: linear DP, quadratic DP, deterministic DP, and TCAS. The table reveals the following:

1) The DP systems resolve more conflicts than TCAS across all environments while alerting less often. Although the DP logic strengthens more frequently, the rate can be reduced by adjusting costs.
2) The logic computed using the deterministic model has a greater NMAC rate when operating in an environment different from the one for which it was optimized. The logic computed using probabilistic pilot response appears less sensitive to unmodeled behavior. For example, the probability of NMAC is 142% greater when the deterministic logic is used in the linear environment than when it is used in the deterministic environment. The probability of NMAC increases by 61% in the quadratic environment. On the other hand, the probability of NMAC is decreased by 18% and 10% when the linear logic is used in the deterministic and quadratic settings, respectively.
3) The linear and quadratic DP systems perform comparably in almost every category despite the structural differences in the pilot response models used to compute them. Quadratic DP strengthens more often, as might be expected based on Figure 2.

The standard error associated with the estimates of Table I was also calculated and was found to be small in relation to the actual estimates. The standard error, on average, was 10.08% the size of the estimate.

VII. Conclusions and Further Work

This paper has presented a method for creating airborne collision avoidance logic that is robust to variability in pilot behavior. The approach involves explicitly designing probabilistic pilot response models and optimizing the logic offline using dynamic programming. The logic is used online as a decision support tool that human operators can choose to follow or ignore. Estimation of the adherence of the pilot to the system is continuously performed online and fed back into the system so that the best alert is issued.

This paper has shown, using realistic encounter simulations, that the logic optimized using probabilistic pilot response models operates effectively across different pilot responses, achieving a greater safety margin than TCAS while alerting less. Unlike TCAS, it does not rely on specialized heuristics to handle atypical, low-probability events.

Extensions to the current work include the incorporation of realistic sensor noise. The introduction of sensor noise turns the problem into a partially-observable MDP, or POMDP, which is typically difficult to solve without approximation [10]. A natural extension to the current work would be to apply Equation 7 [11]. The belief space would be extended to include all variables with any uncertainty, not just the advisory state.

TABLE I

Metrics for different pilot response models

	P(NMAC)	P(alert)	P(strengthening)	P(reversal)
	$7.19 \cdot 10^{-5}$	$1.29 \cdot 10^{-1}$	$4.32 \cdot 10^{-2}$	$7.23 \cdot 10^{-4}$
	$7.86 \cdot 10^{-5}$	$1.30 \cdot 10^{-1}$	$3.79 \cdot 10^{-2}$	$1.24 \cdot 10^{-3}$
Lin.	$1.40 \cdot 10^{-4}$	$1.15 \cdot 10^{-1}$	$4.76 \cdot 10^{-2}$	$2.35 \cdot 10^{-4}$
	$2.69 \cdot 10^{-4}$	$4.75 \cdot 10^{-1}$	$1.37 \cdot 10^{-2}$	$2.01 \cdot 10^{-3}$
	$6.45 \cdot 10^{-5}$	$1.29 \cdot 10^{-1}$	$4.30 \cdot 10^{-2}$	$6.87 \cdot 10^{-4}$
	$6.73 \cdot 10^{-5}$	$1.30 \cdot 10^{-1}$	$3.76 \cdot 10^{-2}$	$8.13 \cdot 10^{-4}$
Quad.	$9.31 \cdot 10^{-5}$	$1.15 \cdot 10^{-1}$	$4.75 \cdot 10^{-2}$	$1.82 \cdot 10^{-4}$
	$2.06 \cdot 10^{-4}$	$4.75 \cdot 10^{-1}$	$1.35 \cdot 10^{-2}$	$1.77 \cdot 10^{-3}$
	$5.86 \cdot 10^{-5}$	$1.29 \cdot 10^{-1}$	$6.01 \cdot 10^{-2}$	$6.02 \cdot 10^{-4}$
	$7.28 \cdot 10^{-5}$	$1.30 \cdot 10^{-1}$	$5.61 \cdot 10^{-2}$	$1.15 \cdot 10^{-3}$
Det.	$5.78 \cdot 10^{-5}$	$1.15 \cdot 10^{-1}$	$5.37 \cdot 10^{-2}$	$2.81 \cdot 10^{-4}$
	$1.13 \cdot 10^{-4}$	$4.75 \cdot 10^{-1}$	$1.24 \cdot 10^{-2}$	$2.75 \cdot 10^{-3}$

Note: The systems appear in the following order within each cell: linear DP, quadratic DP, deterministic DP, and TCAS.

VIII. Acknowledgments

This research was sponsored by the TCAS Program Office at the United States Federal Aviation Administration. The authors appreciate the assistance provided by the TCAS Program Manager, Neal Suchy. This work has benefited from conversations with J. Daniel Griffith of MIT Lincoln Laboratory.

References

[1] J. K. Kuchar and A. C. Drumm, "The Traffic Alert and Collision Avoidance System," *Lincoln Laboratory Journal*, vol. 16, no. 2, pp. 277–296, 2007.

[2] M. J. Kochenderfer and J. P. Chryssanthacopoulos, "A decision-theoretic approach to developing robust collision avoidance logic," in *IEEE International Conference on Intelligent Transportation Systems*, Madeira Island, Portugal, 2010.

[3] ——, "Robust airborne collision avoidance through dynamic programming," Massachusetts Institute of Technology, Lincoln Laboratory, Project Report ATC-371, 2011.

[4] ——, "Partially-controlled Markov decision processes for collision avoidance systems," in *International Conference on Agents and Artificial Intelligence*, Rome, Italy, 2011.

[5] M. J. Kochenderfer, J. P. Chryssanthacopoulos, and R. E. Weibel, "A new approach for designing safer collision avoidance systems," in *Ninth USA/Europe Air Traffic Management Research and Development Seminar*, Berlin, Germany, 2011.

[6] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Belmont, Mass.: Athena Scientific, 2005, vol. 1.

[7] RTCA, "Safety analysis of proposed change to TCAS RA reversal logic," DO-298, RTCA, Inc., Washington, D.C., Nov. 2005.

[8] M. J. Kochenderfer, M. W. M. Edwards, L. P. Espindle, J. K. Kuchar, and J. D. Griffith, "Airspace encounter models for estimating collision risk," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 2, 2010.

[9] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, "Learning policies for partially observable environments: Scaling up," in *International Conference on Machine Learning*, 1995.

[10] S. Temizer, M. J. Kochenderfer, L. P. Kaelbling, T. Lozano-Pérez, and J. K. Kuchar, "Collision avoidance for unmanned aircraft using Markov decision processes," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Canada, 2010.

[11] J. P. Chryssanthacopoulos and M. J. Kochenderfer, "Accounting for state uncertainty in collision avoidance," *Journal of Guidance, Control, and Dynamics*, 2011.