PARTIALLY-CONTROLLED MARKOV DECISION PROCESSES FOR COLLISION AVOIDANCE SYSTEMS*

Mykel J. Kochenderfer and James P. Chryssanthacopoulos
Lincoln Laboratory, Massachusetts Institute of Technology, 244 Wood Street, Lexington, MA, U.S.A.

Keywords: Markov decision processes, Dynamic programming, Collision avoidance.

Abstract: Deciding when and how to avoid collision in stochastic environments requires accounting for the likelihood and relative costs of future sequences of outcomes in response to different sequences of actions. Prior work has investigated formulating the problem as a Markov decision process, discretizing the state space, and solving for the optimal strategy using dynamic programming. Experiments have shown that such an approach can be very effective, but scaling to higher-dimensional problems can be challenging due to the exponential growth of the discrete state space. This paper presents an approach that can greatly reduce the complexity of computing the optimal strategy in problems where only some of the dimensions of the problem are controllable. The approach is demonstrated on an airborne collision avoidance problem where the system must recommend maneuvers to an imperfect pilot.

1 INTRODUCTION

Manually constructing a robust collision avoidance system, whether it be for an autonomous or human-controlled vehicle, is challenging because the future effects of the system cannot be known exactly. Due to their safety-critical nature, collision avoidance systems must maintain a high degree of reliability while minimizing unnecessary path deviation. Recent work has investigated formulating the problem of collision avoidance as a Markov decision process (MDP) and solving for the optimal strategy using dynamic programming (DP) (Kochenderfer and Chryssanthacopoulos, 2010; Kochenderfer et al., 2010a; Temizer et al., 2010). One limitation of this approach is that the computation and memory requirements grow exponentially with the dimensionality of the state space. Hence, these studies focused on MDP formulations that capture only a subset of the relevant state variables at the expense of impaired performance.

This paper presents a new approach for significantly reducing the computation and memory requirements for partially-controlled collision avoidance problems. The approach involves decomposing the problem into two separate subproblems, one controlled and one uncontrolled, that can be solved independently offline using dynamic programming. During execution, the results from the offline computation are combined to determine the approximately optimal action from the current state.

The approach is demonstrated on an airborne collision avoidance system that recommends vertical maneuvers to an imperfect pilot. Although the pilot may maneuver horizontally, it is assumed that the collision avoidance system does not influence the horizontal motion. The problem is naturally represented using seven state variables, which is impractical to solve with a reasonable level of discretization. By carefully decomposing the problem into two lower-dimensional problems, a solution can be obtained quickly and stored in primary physical memory. The performance of the optimized system is compared in simulation against the Traffic Alert and Collision Avoidance System (TCAS), which is currently mandated worldwide on all large transport aircraft (RTCA, 2008).

The next section briefly summarizes related work on collision avoidance. Section 3 reviews Markov decision processes. Section 4 describes the solution method and outlines the required assumptions. Section 5 applies the method to an airborne collision avoidance problem. Section 6 evaluates the suc-

*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.

J. Kochenderfer M. and P. Chryssanthacopoulos J..
PARTIALLY-CONTROLLED MARKOV DECISION PROCESSES FOR COLLISION AVOIDANCE SYSTEMS.
DOI: 10.5220/0003135800610070
In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence* (ICAART-2011), pages 61-70
ISBN: 978-989-8425-40-9
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

cess of the method in simulation and presents results that show that the method significantly outperforms TCAS. Section 7 concludes and outlines further work.

2 RELATED WORK

A common technique for collision avoidance in autonomous and semi-autonomous vehicles is to define conflict zones for each obstacle and then use a deterministic model, such as linear extrapolation, to predict whether a conflict will occur (Bilimoria, 2000; Dowek et al., 2001; Chamlou, 2009). If a conflict is anticipated, the collision avoidance system selects the maneuver that provides the minimal path deviation while preventing the conflict. Such an approach requires very little computation and can prevent collision much of the time, but it lacks robustness because the deterministic model ignores the stochastic nature of the environment. Although one may mitigate collision risk to some extent by artificially enlarging the conflict zones to accommodate uncertainty in the future behavior of the vehicles, this approach frequently results in unnecessary path deviation. The TCAS collision avoidance logic adopts an approach along these lines but incorporates a large collection of hand-crafted, heuristic rules to enhance robustness to unexpected behavior.

Several other approaches to collision avoidance can be found in the literature that do not use a probabilistic model of vehicle behavior, including potential field methods (Khatib and Maitre, 1978; Duong and Zeghal, 1997; Eby and Kelly, 1999) and rapidly-expanding random trees (LaValle, 1998; Kuwata et al., 2008; Saunders et al., 2009). However, avoiding collision with a high degree of reliability while keeping the rate of path deviation low requires the use of a probabilistic model that accounts for future state uncertainty. Several methods have been suggested that involve using a probabilistic model to estimate the probability of conflict and to choose the maneuver that keeps the probability of conflict below some set threshold (Yang and Kuchar, 1997; Carpenter and Kuchar, 1997; Kochenderfer et al., 2010a). One limitation of these threshold-based approaches is that they do not model the effects of delaying the avoidance maneuver. In many cases, it can be beneficial to see how the encounter develops before committing to a particular maneuver. The dynamic programming approach pursued in this work, in contrast, takes into account every possible future sequence of actions taken by the collision avoidance system and their outcomes when making a decision.

3 MARKOV DECISION PROCESSES

An MDP is defined by a transition function T and cost function C. The probability of transitioning from state s to state s' after executing action a is given by $T(s,a,s')$. The immediate cost when executing action a from state s is given by $C(s,a)$. In this paper, the state space S and action space A are assumed to be finite (Puterman, 1994; Bertsekas, 2005).

A policy is a function π that maps states to actions. The expected sum of immediate costs when following a policy π for K steps starting from state s is denoted $J_K^\pi(s)$ and is often called the cost-to-go function. The optimal solution to an MDP with a receding horizon of K is a policy π_K^* that minimizes the cost to go from every state.

One way to compute π_K^* is to first compute J_K^*, the cost-to-go function for the optimal policy, using a dynamic programming algorithm known as value iteration. The function $J_0^*(s)$ is set to zero for all states s. If the state space includes terminal states with immediate cost $C(s)$, then $J_0^*(s) = C(s)$ for those terminal states. The function $J_k^*(s)$ is computed from J_{k-1}^* as follows:

$$J_k^*(s) = \min_a \left[C(s,a) + \sum_{s'} T(s,a,s') J_{k-1}^*(s') \right]. \quad (1)$$

The iteration continues until horizon K.

The expected cost to go when executing a from s and then continuing with an optimal policy for $K-1$ steps is given by

$$J_K^*(s,a) = C(s,a) + \sum_{s'} T(s,a,s') J_{K-1}^*(s'). \quad (2)$$

An optimal policy may be obtained directly from $J_K^*(s,a)$:

$$\pi_K^*(s) = \arg\min_a J_K^*(s,a). \quad (3)$$

If the state space contains continuous variables, which is common for collision avoidance problems, the state space can be discretized using a multidimensional grid or simplex scheme (Davies, 1997). The transition function $T(\mathbf{x},a,\mathbf{x}')$ in continuous space can be translated into a discrete transition function $T(s,a,s')$ using a variety of sampling and interpolation methods (Kochenderfer et al., 2010a). Once the state space, transition function, and cost function have been discretized, $J^*(s,a)$ may be computed for each discrete state s and action a. For a continuous state $\mathbf{x}$ and action a, $J^*(\mathbf{x},a)$ may be approximated using, for example, multilinear interpolation. The best action to execute from continuous state $\mathbf{x}$ is simply $\arg\min_a J^*(\mathbf{x},a)$.

Discretizing the full state space can result in a large number of discrete states, exponential in the number of dimensions, which makes computing J^* infeasible for many problems. This "curse of dimensionality" (Bellman, 1961) has led to a variety of different approximation methods (Powell, 2007).

4 PARTIAL CONTROL

This paper explores a new solution technique for partially-controlled MDPs that is applicable to certain collision avoidance problems. It may be applied to interception-seeking or goal-oriented problems as well by incorporating negative costs. So long as the problem satisfies a set of assumptions, this solution method will provide a finite-horizon solution. The approach involves independently solving a controlled subproblem and an uncontrolled subproblem and combining the results online to identify the approximately optimal action.

4.1 Assumptions

It is assumed that the state is represented by a set of variables, some controlled and some uncontrolled. The state space of the controlled variables is denoted S_c, and the state space of the uncontrolled variables is denoted S_u. The state of the controlled variables at time t is denoted $s_c(t)$, and the state of the uncontrolled variables at time t is denoted $s_u(t)$. The solution technique may be applied when the following three assumptions hold:

1. The state $s_u(t+1)$ depends only upon $s_u(t)$. The probability of transitioning from s_u to s'_u is given by $T(s_u, s'_u)$.
2. The episode terminates when $s_u \in G \subset S_u$ with immediate cost $C(s_c)$.
3. In nonterminal states, the immediate cost $c(t+1)$ depends only upon $s_c(t)$ and $a(t)$. If the controlled state is s_c and action a is executed, the immediate cost is denoted $C(s_c, a)$.

Figure 1 shows the influence diagram for this model.

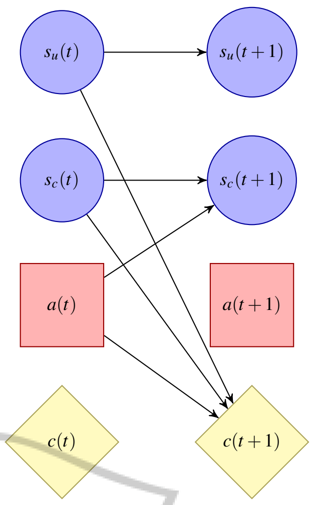


Figure 1: Influence diagram illustrating partial control in a Markov decision process.

4.2 Controlled Subproblem

Solving the controlled subproblem involves computing the optimal policy for the controlled variables under the assumption that the time until s_u enters G, denoted τ, is known. In an airborne collision avoidance context, τ may be the number of steps until another aircraft comes within 500 ft horizontally. Of course, τ cannot be determined exactly from $s_u(t)$ because it depends upon an event that occurs in the future, but this will be addressed by the uncontrolled subproblem (Section 4.3).

The cost to go from s_c given τ is denoted $J_\tau(s_c)$. The series $J_0, \ldots, J_K$ is computed recursively, starting with $J_0(s_c) = C(s_c)$ and iterating as follows:

$$J_k(s_c) = \min_a \left[C(s_c, a) + \sum_{s'_c} T(s_c, a, s'_c) J_{k-1}(s'_c) \right]. \tag{4}$$

The expected cost to go from s_c when executing a for one step and then following the optimal policy is given by

$$J_k(s_c, a) = C(s_c, a) + \sum_{s'_c} T(s_c, a, s'_c) J_{k-1}(s'_c). \tag{5}$$

The K-step expected cost to go when $\tau > K$ is denoted $J_{\bar{K}}$. It is computed by initializing $J_0(s_c) = 0$ for all states and iterating equation 4 to horizon K. The series $J_0, \ldots, J_K, J_{\bar{K}}$ is saved in a table in memory, requiring $O(K|A||S_c|)$ entries.

4.3 Uncontrolled Subproblem

Solving the uncontrolled subproblem involves using the probabilistic model of the uncontrolled dynamics to infer a distribution over τ for each uncontrolled state s_u. This distribution is referred to as the entry time distribution because it represents the distribution over the time for s_u to enter G. The probability that s_u enters G in τ steps is denoted $D_\tau(s_u)$ and may be computed using dynamic programming. The probability that $\tau = 0$ is given by

$$D_0(s_u) = \begin{cases} 1 & \text{if } s_u \in G, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The probability that $\tau = k$, for $k > 0$, is computed from D_{k-1} as follows:

$$D_k(s_u) = \begin{cases} 0 & \text{if } s_u \in G, \\ \sum_{s'_u} T(s_u, s'_u) D_{k-1}(s'_u) & \text{otherwise.} \end{cases} \quad (7)$$

Depending on s_u, there may be some probability that s_u does not enter G within K steps. This probability is denoted $D_{\bar{K}}(s_u)$ and may be computed from $D_0, \ldots, D_K$:

$$D_{\bar{K}}(s_u) = 1 - \sum_{k=0}^{K} D_k(s_u). \quad (8)$$

The sequence $D_0, \ldots, D_K, D_{\bar{K}}$ is stored in a table with $O(K|S_u|)$ entries. Multilinear interpolation of the distributions may be used to determine $D_\tau(\mathbf{x}_u)$ at an arbitrary continuous state $\mathbf{x}_u$.

4.4 Online Solution

After $J_0, \ldots, J_K, J_{\bar{K}}$ and $D_0, \ldots, D_K, D_{\bar{K}}$ have been computed offline, they are used together online to determine the approximately optimal action to execute from the current state. For any discrete state s in the original state space, $J^*_K(s,a)$ may be computed as follows:

$$J^*_K(s,a) = D_{\bar{K}}(s_u) J_{\bar{K}}(s_c,a) + \sum_{k=0}^{K} D_k(s_u) J_k(s_c,a), \quad (9)$$

where s_u is the discrete uncontrolled state and s_c is the discrete controlled state associated with s. Combining the controlled and uncontrolled solutions online in this way requires time linear in the size of the horizon. Multilinear interpolation can be used to estimate $J^*_K(\mathbf{x},a)$ for an arbitrary state $\mathbf{x}$, and from this the optimal action may be obtained.

The memory requirements for directly storing $J^*_K(s,a)$ is $O(|A||S_c||S_u|)$. However, the method presented in this section allows the solution to be represented using $O(K|A||S_c| + K|S_u|)$ storage, which can be a tremendous savings when $|S_c|$ and $|S_u|$ are large. For the collision avoidance problem discussed in the next section, this method allows the cost table to be stored in 500 MB instead of over 1 TB. The offline computational savings are even more significant.

An alternative to using dynamic programming for computing the entry time distribution offline is to use Monte Carlo to estimate the entry time distribution online. A Monte Carlo approach does not require the uncontrolled variables to be discretized and does not require $D_0, \ldots, D_K, D_{\bar{K}}$ to be stored in memory. However, using Monte Carlo increases the amount of computation required online. In addition, for problems where the conflict region is small, the number of samples required to produce an adequate distribution may be large. Importance sampling and other sampling methods may be used to help improve the quality of the estimates of the entry time distribution (Chryssanthacopoulos et al., 2010).

5 AIRBORNE COLLISION AVOIDANCE SYSTEM

This section demonstrates the approach suggested in the previous section on an MDP representing an airborne collision avoidance problem. In this problem, the collision avoidance system issues resolution advisories to pilots who then adjust their vertical rate to avoid coming within 500 ft horizontally and 100 ft vertically of an intruding aircraft. This section considers a simplified version of the collision avoidance problem in which one aircraft equipped with a collision avoidance system, called the own aircraft, encounters only one other unequipped aircraft, called the intruder aircraft. The remainder of the section outlines the assumptions and decomposes the problem into controlled and uncontrolled subproblems.

5.1 Assumptions

In this problem, s_c represents the state of the vertical motion variables, and s_u represents the state of the horizontal motion variables. This problem defines coming within 500 ft horizontally and 100 ft vertically of an intruder as a conflict. This definition matches the near mid-air collision (NMAC) definition used in prior TCAS studies (RTCA, 2005).

The first assumption in Section 4.1 requires that $s_u(t+1)$ depend only upon $s_u(t)$. In this collision avoidance problem, it is assumed that pilots randomly maneuver horizontally, and that the advisories issued by the collision avoidance system do not influence the horizontal motion.

The second assumption requires the episode to terminate when s_u enters G. In this problem, G is the set of states where there is a horizontal conflict, defined to be when an intruder comes within 500 ft horizontally. The immediate cost when this occurs is given by $C(s_c)$, which is one when the intruder is within 100 ft vertically and zero otherwise. In simulation, the episode does not terminate when s_u enters G, since entering G does not necessarily imply that there has been a conflict (e.g., the two aircraft may have safely missed each other by 1000 ft vertically). However, it is generally sufficient to plan up to the moment where s_u enters G because adequate separation at that moment usually indicates that the encounter is resolved.

The third assumption requires that for states where $s_u \notin G$ the immediate cost function depends on the controlled state variables and the action. As outlined in Section 5.2.3, the nonterminal cost function only depends on the advisory state and the advisory being issued.

5.2 Controlled Subproblem

The controlled subproblem, formulated as an MDP, is defined by the available actions, the dynamics, and the cost function. The dynamics are determined by the pilot response model and aircraft dynamic model. The cost function takes into account both safety and operational considerations. In addition to describing these components of the MDP, this section discusses the resulting optimal policy.

5.2.1 Resolution Advisories

The airborne collision avoidance system may choose to issue one of two different initial advisories: climb at least 1500 ft/min or descend at least 1500 ft/min. Following the initial advisory, the system may choose to either terminate, reverse, or strengthen the advisory. An advisory that has been reversed requires a vertical rate of 1500 ft/min in the opposite direction of the original advisory. An advisory that has been strengthened requires a vertical rate of 2500 ft/min in the direction of the original advisory. After an advisory has been strengthened, it can then be weakened to reduce the required vertical rate to 1500 ft/min in the direction of the original advisory.

5.2.2 Dynamic Model

The state is represented using four variables:

- h: altitude of the intruder relative to the own aircraft,
- $\dot{h}_0$: vertical rate of the own aircraft,
- $\dot{h}_1$: vertical rate of the intruder aircraft, and
- s_{RA}: the state of the resolution advisory.

The discrete variable s_{RA} contains the necessary information to model the pilot response, which includes the active advisory and the time to execution by the pilot. Five seconds are required for the pilot to begin responding to an initial advisory. The pilot then applies a 1/4 g acceleration to comply with the advisory. Subsequent advisories are followed with a 1/3 g acceleration after a three second delay. When an advisory is not active, the pilot applies an acceleration selected at every step from a zero-mean Gaussian with 3 ft/s^2 standard deviation. At each step, the intruder pilot independently applies a random acceleration from a zero-mean Gaussian with 3 ft/s^2 standard deviation.

The continuous state variables are discretized according to the scheme in Table 1. The discrete state transition probabilities were computed using sigma-point sampling and multilinear interpolation (Kochenderfer et al., 2010a). This discretization scheme produces a discrete model with 213 thousand discrete states.

Table 1: Controlled Variable Discretization.

Variable	Grid Edges
h	$-1000, -900, \ldots, 1000$ ft
$\dot{h}_0$	$-2500, -2250, \ldots, 2500$ ft/min
$\dot{h}_1$	$-2500, -2250, \ldots, 2500$ ft/min

5.2.3 Cost Function

An effective collision avoidance system must satisfy a number of competing objectives, including maximizing safety and minimizing the rate of unnecessary alerts and path deviation. These objectives are encoded in the cost function. In addition to incurring a cost for conflict, it may be desirable to incur a cost for other events such as alerting or strengthening the advisory. The costs of various events are summarized in Table 2. A small negative cost is awarded at every time step the system is not alerting to provide some incentive to discontinue alerting after the encounter has been resolved.

Table 2: Event Costs.

Conflict	Alert	Strengthening	Reversal	Clear of Conflict
1	0.001	0.009	0.01	$-1 \cdot 10^{-4}$

5.2.4 Optimal Policy

The optimal cost-to-go tables $J_0, \ldots, J_K, J_{\bar{K}}$ were computed offline in less than two minutes on a single 3 GHz Intel Xeon core using a horizon of $K = 39$ steps. Storing only the values for the valid state-action pairs requires 263 MB using a 64-bit floating point representation. Figure 2 shows plots of the optimal policy through different slices of the state space where the own aircraft is initially climbing at 1500 ft/min and the intruder is level.

Figure 2(a) shows the policy when an alert has not been issued. The blue region indicates where the logic will issue a descend advisory, and the green region indicates where the logic will issue a climb advisory. The optimal policy will sometimes issue a climb even when the intruder is above. This occurs when the aircraft are closely separated in altitude and little time remains until potential conflict. Because the own aircraft is already climbing, there is insufficient time to accelerate downward to avoid conflict. Climbing above the intruder is more effective. Another notable feature of the plot is that no advisory is issued when $\tau \leq 5$ s. Because an advisory has no effect until five seconds after it is issued in this model, alerting less than five seconds prior to conflict is no more effective than not alerting.

In Figure 2(b), a strengthened climb advisory was issued two seconds previously. Should the logic continue issuing the strengthened climb advisory, the pilot will begin climbing to 2500 ft/min in one second. The white region indicates where the logic will discontinue the advisory. The advisory is typically discontinued when the vertical separation is large and there is much time until conflict. The gold region indicates where the logic will continue to issue the strengthened advisory. In the red region, the logic will reverse the climb to a descend when the intruder is above the own aircraft and maintaining the climb might induce conflict. In the teal region of the plot, the logic will weaken the advisory when the intruder is far enough above the own aircraft that reversing the advisory is unnecessary.

5.3 Uncontrolled Subproblem

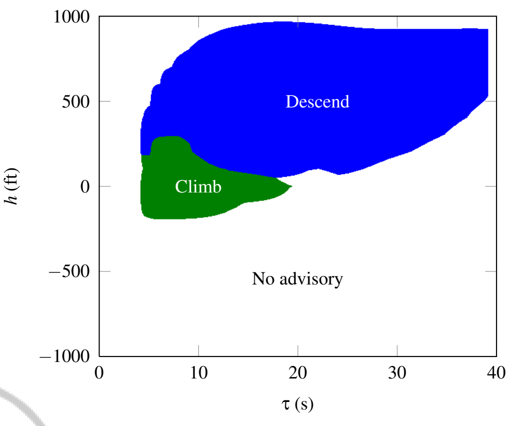
The uncontrolled subproblem involves estimating the distribution over τ (i.e., the time until the aircraft are separated less than 500 ft horizontally) given the current state. This section describes the horizontal dynamics and three methods for estimating the entry time distribution.

5.3.1 Dynamic Model

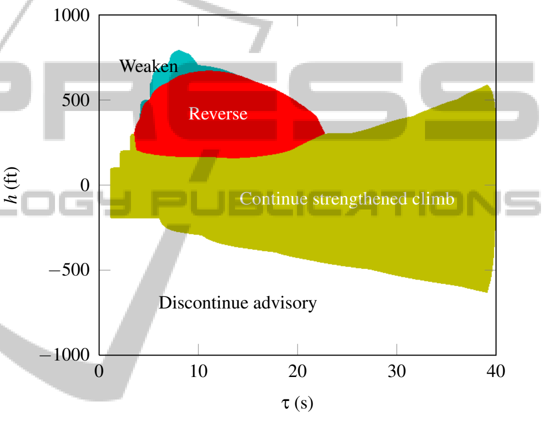
The aircraft move in the horizontal plane in response to independent random accelerations generated from a zero-mean Gaussian with a standard deviation of 3 ft/s^2. The motion can be described by a three-dimensional model, instead of the typical four-dimensional (relative positions and velocities) model, due to rotational symmetry in the dynamics. The three state variables are as follows:

- 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.

These variables are illustrated in Figure 3.



(a) s_{RA} = "no advisory"



(b) s_{RA} = "increase climb in one second"

Figure 2: Optimal action plots for $\dot{h}_0 = 1500$ ft/min, $\dot{h}_1 = 0$ ft/min.

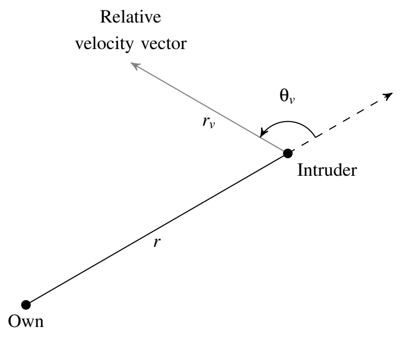


Figure 3: Three-variable model of horizontal dynamics.

5.3.2 Dynamic Programming Entry Time Distribution

The entry time distribution can be estimated offline using dynamic programming as discussed in Section 4.3. The state space was discretized using the scheme in Table 3, resulting in 730 thousand discrete states. The offline computation required 92 seconds on a single 3 GHz Intel Xeon core. Storing $D_0, \ldots, D_{39}$ in memory using a 64-bit floating point representation requires 222 MB.

Table 3: Uncontrolled Variable Discretization.

Variable	Grid Edges
r	$0, 50, \ldots, 1000, 1500, \ldots, 40000$ ft
r_v	$0, 10, \ldots, 1000$ ft/s
θ_v	$-180°, -175°, \ldots, 180°$

5.3.3 Monte Carlo Entry Time Distribution

Monte Carlo estimation can be used online to estimate the entry time distribution as explained at the end of Section 4.3. The experiments in this paper use 100 Monte Carlo samples to estimate τ.

5.3.4 Simple Entry Time Distribution

A point estimate of τ can be obtained online as follows. The range rate is given by

$$\dot{r} = r_v \cos(\theta_v). \tag{10}$$

If the aircraft are converging in range, then τ can be approximated by $r/|\dot{r}|$. Otherwise, τ is set beyond the horizon.

6 RESULTS

This section evaluates the performance of the collision avoidance system using simulated encounters and compares it against the current version of TCAS, Version 7.1.

6.1 Encounter Initialization

Encounters are initialized in the horizontal plane by randomly and independently generating the initial ground speeds of both aircraft, s_0 and s_1, from a uniform distribution between 100 and 500 ft/s. The horizontal range between the aircraft is initialized to $r = t_{\text{target}}(s_0 + s_1) + u_r$, where u_r is a zero-mean Gaussian with 500 ft standard deviation. The parameter t_{target}, nominally set to 40 s, controls how long until the aircraft come into conflict.

The bearing of the intruder aircraft with respect to the own aircraft, χ, is sampled from a zero-mean Gaussian distribution with a standard deviation of 2°. The heading of the intruder with respect to the heading of the own aircraft, β, is sampled from a Gaussian distribution with a mean of 180° and a standard deviation of 2°. When $\beta = 180°$, the intruder is heading directly toward the own aircraft.

The initial vertical rates $\dot{h}_0$ and $\dot{h}_1$ are drawn independently from a uniform distribution spanning -1000 and 1000 ft/min. The initial altitude of the own aircraft, h_0, is set to 43,000 ft. The initial altitude of the intruder is $h_0 + t_{\text{target}}(\dot{h}_0 - \dot{h}_1) + u_h$, where u_h is a zero-mean Gaussian with 25 ft standard deviation.

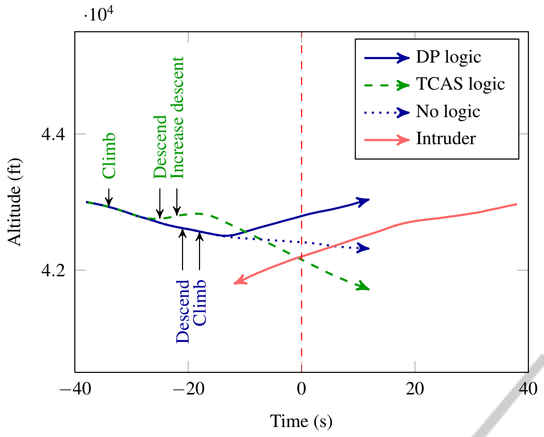
6.2 Example Encounter

Figure 4 shows an example encounter comparing the behavior of the system using the DP entry time distribution against the TCAS logic. Figure 5 shows the entry time distribution computed using the three methods of Section 5.3 at the points in time when the system issues alerts.

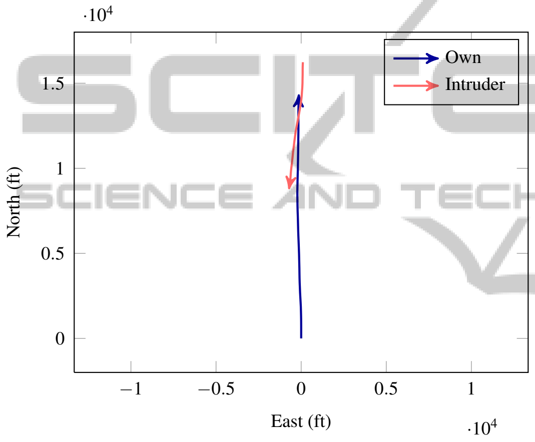
Seventeen seconds into the encounter, the DP logic issues a descend to pass below the intruder. The expected cost to go for issuing a descend advisory is approximately 0.00928, lower than the expected cost to go for issuing a climb advisory (0.0113) or for not issuing an advisory (0.00972). The DP entry time distribution at this time has a conditional mean $\mathrm{E}[\tau \mid \tau < 40\,\text{s}]$ of approximately 12.01 s, and a considerable portion of the probability mass ($\sim$40%) is assigned to $\tau \geq 40$ s. The Monte Carlo entry time distribution, in comparison, has less support but a comparable conditional mean of 17.12 s. Only 15% of the probability mass is concentrated on $\tau \geq 40$ s. The point estimate of τ using the simple method is 21.65 s.

After the descend advisory is issued, the intruder begins to increase its descent, causing the DP logic to reverse the descend to a climb 20 seconds into the encounter. The pilot begins the climb maneuver three seconds later. Once the aircraft are safely separated, the DP logic discontinues the advisory at $t = 31$ s. The minimum horizontal separation is 342 ft, at which time the vertical separation is 595 ft. No conflict occurs.

TCAS initially issues a climb advisory four seconds into the encounter because it anticipates, using straight-line projection, that by climbing it can safely pass above the intruder. Nine seconds later, when the own aircraft is executing its climb advisory, TCAS reverses the climb to a descend because it projects that maintaining the climb will not provide the required separation. TCAS strengthens the advisory three seconds later, but fails to resolve the conflict. The aircraft miss each other by 342 ft horizontally and 44 ft vertically. Although the TCAS logic alerts earlier and more often, the DP logic still outperforms it in this example encounter.

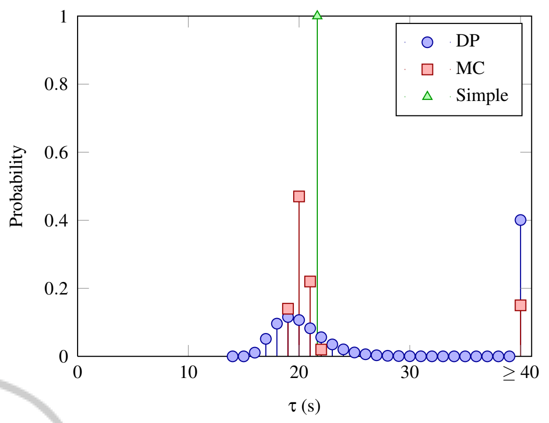


(a) Vertical profile.

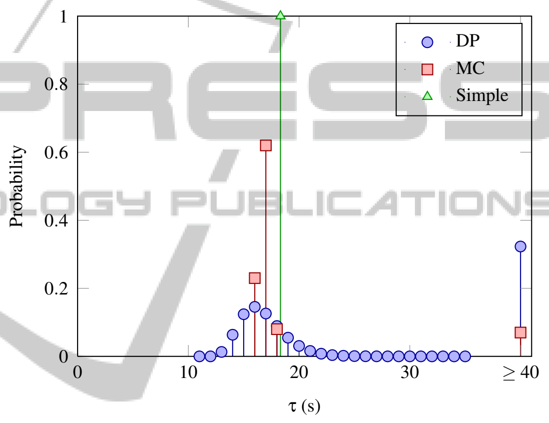


(b) Horizontal profile.

Figure 4: Example encounter comparing the system with the DP entry time distribution against TCAS.



(a) $t = 17$ s.



(b) $t = 20$ s.

Figure 5: Entry time distribution computed using dynamic programming (DP), Monte Carlo (MC), and the simple (Simple) methods at the two alerting points of the DP logic in the example encounter of Figure 4.

6.3 Performance Evaluation

Table 4 summarizes the results of simulating the DP logic and the TCAS logic on one million encounters generated by the model of Section 5. The table summarizes the number of conflicts, alerts, strengthenings, and reversals.

As the table shows, the DP logic can provide a much lower conflict rate while significantly reducing the alert rate. The Monte Carlo entry time distribution results in a greater number of conflicts, but it alerts less frequently than the other methods. Increasing the number of samples used generally improves performance but increases online computation time. The DP logic using the simple point estimate of τ resolves all but one conflict while rarely reversing or strengthening the advisory, but alerts more frequently than Monte Carlo.

Table 4: Performance Evaluation.

	DP Logic (DP Entry)	DP Logic (MC Entry)
Conflicts	2	11
Alerts	540,113	400,457
Strengthenings	39,549	37,975
Reversals	1242	747
	DP Logic (Simple Entry)	**TCAS Logic**
Conflicts	1	101
Alerts	939,745	994,640
Strengthenings	26,485	45,969
Reversals	129	193,582

6.4 Safety Curve

The results of Section 6.3 considered the performance of the system optimized using the fixed event costs of Table 2. Figure 6 shows the safety curves for the DP logic and TCAS when different parameters are varied.

The DP logic safety curves were produced by varying the cost of alerting from zero to one while keeping the other event costs fixed. Separate curves were produced for the three methods for estimating the entry time distribution. The upper-right region of the plot corresponds to costs of alerting near zero and the lower-left region corresponds to costs near one.

The safety curve for TCAS was generated by varying the sensitivity level of TCAS. The sensitivity level of TCAS is a system parameter of the logic that increases with altitude. At higher sensitivity levels, TCAS will generally alert earlier and more aggressively to prevent conflict.

The safety curves show that the DP logic can exceed or meet the level of safety provided by TCAS while alerting far less frequently. The safety curves can aid in choosing an appropriate value for the cost of alerting that satisfies a required safety threshold.

Figure 6 also reveals that the DP and Monte Carlo methods for estimating τ offer similar performance and that they both outperform the simple method, especially when the cost of alerting is high and the logic can only alert sparingly to prevent conflict. In the upper-right region of the plot, the three methods are nearly indistinguishable.

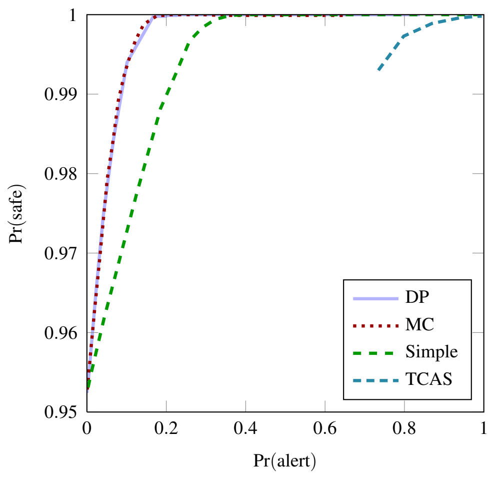


Figure 6: Safety curves. Each point on the curves was estimated from 10,000 simulations.

7 CONCLUSIONS AND FURTHER WORK

This paper presented a method for solving large MDPs that satisfy certain assumptions by decomposing the problem into controlled and uncontrolled subproblems that can be solved independently offline and recombined online. The method was applied to airborne collision avoidance and was compared against TCAS, a system that was under development for several decades and has a proven safety record.

The experiments demonstrate that the collision avoidance logic that results from solving the MDP using the method presented in this paper reduces the risk of collision by a factor of 50 while issuing fewer alerts than TCAS in the simulated encounters. The system reverses less than 1% of the time that TCAS reverses, and the system strengthens less frequently as well. It should be emphasized that further simulation studies using more realistic encounter models are required to quantify the expected performance of the DP logic (Kochenderfer et al., 2010b).

Real collision avoidance systems have imperfect sensors, which results in state uncertainty. TCAS currently relies on radar beacon surveillance, which results in somewhat significant uncertainty in the intruder bearing. When state uncertainty is significant, the uncertainty must be taken into account when choosing actions. With a sensor model, the problem may be transformed into a partially-observable Markov decision process (POMDP) and solved approximately using various methods (Smith and Simmons, 2005; Kurniawati et al., 2008).

ACKNOWLEDGEMENTS

This work is the result of research sponsored by the TCAS Program Office at the Federal Aviation Administration. The authors appreciate the support provided by the TCAS Program Manager, Neal Suchy. This work has benefited from discussions with Leslie Kaelbling and Tomas Lozano-Perez from the MIT Computer Science and Artificial Intelligence Laboratory.

REFERENCES

Bellman, R. E. (1961). *Adaptive control processes: A guided tour*. Princeton University Press.

Bertsekas, D. P. (2005). *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, Belmont, Mass., 3rd edition.

Bilimoria, K. D. (2000). A geometric optimization approach to aircraft conflict resolution. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Denver, Colo.

Carpenter, B. D. and Kuchar, J. K. (1997). Probability-based collision alerting logic for closely-spaced parallel approach. In *AIAA 35th Aerospace Sciences Meeting*, Reno, NV.

Chamlou, R. (2009). Future airborne collision avoidance—design principles, analysis plan and algorithm development. In *Digital Avionics Systems Conference*.

Chryssanthacopoulos, J. P., Kochenderfer, M. J., and Williams, R. E. (2010). Improved Monte Carlo sampling for conflict probability estimation. In *AIAA Non-Deterministic Approaches Conference*, Orlando, Florida.

Davies, S. (1997). Multidimensional triangulation and interpolation for reinforcement learning. In Mozer, M. C., Jordan, M. I., and Petsche, T., editors, *Advances in Neural Information Processing Systems*, volume 9, pages 1005–1011. MIT Press, Cambridge, Mass.

Dowek, G., Geser, A., and Muñoz, C. (2001). Tactical conflict detection and resolution in a 3-D airspace. In *4th USA/Europe Air Traffic Management R&D Seminar*, Santa Fe, New Mexico.

Duong, V. N. and Zeghal, K. (1997). Conflict resolution advisory for autonomous airborne separation in low-density airspace. In *IEEE Conference on Decision and Control*, volume 3, pages 2429–2434.

Eby, M. S. and Kelly, W. E. (1999). Free flight separation assurance using distributed algorithms. In *IEEE Aerospace Conference*, volume 2, pages 429–441.

Khatib, O. and Maitre, J.-F. L. (1978). Dynamic control of manipulators operating in a complex environment. In *Symposium on Theory and Practice of Robots and Manipulators*, pages 267–282, Udine, Italy. Elsevier.

Kochenderfer, M. J. and Chryssanthacopoulos, J. P. (2010). A decision-theoretic approach to developing robust collision avoidance logic. In *IEEE International Conference on Intelligent Transportation Systems*, Madeira Island, Portugal.

Kochenderfer, M. J., Chryssanthacopoulos, J. P., Kaelbling, L. P., and Lozano-Perez, T. (2010a). Model-based optimization of airborne collision avoidance logic. Project Report ATC-360, Massachusetts Institute of Technology, Lincoln Laboratory.

Kochenderfer, M. J., Edwards, M. W. M., Espindle, L. P., Kuchar, J. K., and Griffith, J. D. (2010b). Airspace encounter models for estimating collision risk. *Journal of Guidance, Control, and Dynamics*, 33(2):487–499.

Kurniawati, H., Hsu, D., and Lee, W. (2008). SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems*.

Kuwata, Y., Fiore, G. A., Teo, J., Frazzoli, E., and How, J. P. (2008). Motion planning for urban driving using RRT. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1681–1686.

LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Computer Science Department, Iowa State University.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley, Hoboken, NJ.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley series in probability and mathematical statistics. Wiley, New York.

RTCA (2005). Safety analysis of proposed change to TCAS RA reversal logic, DO-298. RTCA, Inc., Washington, D.C.

RTCA (2008). Minimum operational performance standards for Traffic Alert and Collision Avoidance System II (TCAS II), DO-185b. RTCA, Inc., Washington, D.C.

Saunders, J., Beard, R., and Byrne, J. (2009). Vision-based reactive multiple obstacle avoidance for micro air vehicles. In *American Control Conference*, pages 5253–5258.

Smith, T. and Simmons, R. G. (2005). Point-based POMDP algorithms: Improved analysis and implementation. In *Uncertainty in Artificial Intelligence*.

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

Yang, L. C. and Kuchar, J. K. (1997). Prototype conflict alerting system for free flight. *Journal of Guidance, Control, and Dynamics*, 20(4):768–773.