Collision Avoidance for Unmanned Aircraft using Markov Decision Processes[*]

Selim Temizer[†], Mykel J. Kochenderfer[‡], Leslie P. Kaelbling[§]
Tomás Lozano-Pérez[¶] and James K. Kuchar[‖]

Before unmanned aircraft can fly safely in civil airspace, robust airborne collision avoidance systems must be developed. Instead of hand-crafting a collision avoidance algorithm for every combination of sensor and aircraft configuration, we investigate the automatic generation of collision avoidance algorithms given models of aircraft dynamics, sensor performance, and intruder behavior. By formulating the problem of collision avoidance as a Markov Decision Process (MDP) for sensors that provide precise localization of the intruder aircraft, or a Partially Observable Markov Decision Process (POMDP) for sensors that have positional uncertainty or limited field-of-view constraints, generic MDP/POMDP solvers can be used to generate avoidance strategies that optimize a cost function that balances flight-plan deviation with collision. Experimental results demonstrate the suitability of such an approach using four different sensor modalities and a parametric aircraft performance model.

I. Introduction

Because of the potential for commercial, military, law-enforcement, scientific, and other purposes, unmanned aircraft have received considerable attention in recent years. However, unmanned aircraft are not currently permitted access to civil airspace in the United States without special permission from the Federal Aviation Administration (FAA). One of the primary concerns with integrating unmanned aircraft is their inability to robustly sense and avoid other aircraft. Although sensor information can be transmitted to a ground pilot who can then maneuver the aircraft to avoid collision, there are concerns about communication latency and reliability. In order to provide the high level of safety required by the FAA, an automated airborne collision avoidance system is likely to be necessary.

The deployment of any collision avoidance system requires a lengthy development process followed by a rigorous certification process. Development of the Traffic Alert and Collision Avoidance System (TCAS), currently mandated onboard all large transport aircraft worldwide, started in the 1950s but was not certified for operational use until relatively recently.[1] The system issues vertical rate resolution advisories to pilots who are then responsible for maneuvering the aircraft. TCAS is not certified for autonomous use, and it is likely that the certification of an autonomous system will require even more extensive testing and analysis.

Further complicating the certification process of collision avoidance systems for unmanned aircraft is the diversity of their aircraft performance characteristics and sensor capabilities. Unmanned aircraft can range from under a pound to many tons with wildly varying flight dynamics. Several sensor modalities have been considered for supporting collision avoidance, including electro-optical/infrared (EO/IR), radar, TCAS, and Automatic Dependent Surveillance-Broadcast (ADS-B). As Table 1 illustrates, these sensor modalities vary

[*] The Lincoln Laboratory portion of this work was supported under Air Force Contract #FA8721-05-C-0002. The MIT CSAIL portion of this work was supported by the Office of the Director of Defense Research and Engineering. Interpretations, opinions, and conclusions are those of the authors and do not reflect the official position of the United States government.

[†] Ph.D. Student, Massachusetts Institute of Technology, 32-G585, 32 Vassar Street, Cambridge, MA 02139.

[‡] Technical Staff, Lincoln Laboratory, Massachusetts Institute of Technology, 244 Wood Street, Lexington, MA 02420.

[§] Professor of Computer Science and Engineering, Massachusetts Institute of Technology, 32-G486, 32 Vassar Street, Cambridge, MA 02139.

[¶] Professor of Computer Science and Engineering, Massachusetts Institute of Technology, 32-G492, 32 Vassar Street, Cambridge, MA 02139.

[‖] Associate Group Leader, Lincoln Laboratory, Massachusetts Institute of Technology, 244 Wood Street, Lexington, MA 02420.

in their capabilities. It would be very difficult to develop and certify a different collision avoidance system for every combination of sensor configuration and aircraft platform. Current efforts in the unmanned aircraft industry have focused on proprietary solutions for specific platforms and sensors, but a common system that would accommodate different sensor configurations and flight characteristics would significantly reduce the cost of development and certification.

Table 1. Qualitative performance characteristics of various sensor modalities. Green, yellow, and red indicate good, moderate, and poor performance, respectively (FoV stands for *field-of-view*).

	Measurement Accuracy			Coverage		
Modality	Range	Azimuth	Elevation	FoV	Range	Traffic
TCAS	🟩	🟨	🟩	🟩	🟩	🟨
Radar	🟩	🟩	🟩	🟨	🟩	🟩
EO/IR	🟥	🟩	🟩	🟨	🟨	🟩
ADS-B	🟩	🟩	🟩	🟩	🟩	🟧

Such a system would take as input models of the flight dynamics, intruder behavior, and sensor characteristics and attempt to optimize the avoidance strategy so that a predefined cost function is minimized. The cost function could take into account competing objectives, such as flight plan adherence and avoiding collision. One way to formulate a problem involving the optimal control of a stochastic system is as a Markov Decision Process (MDP), or more generally as a Partially Observable Markov Decision Process (POMDP) to also account for observation uncertainty. POMDPs have been studied in the operations research and artificial intelligence communities, but only in the past few years have generic POMDP solution methods been developed that can approximately solve problems with moderate to large state spaces in reasonable time. In this work we investigate the feasibility of applying state-of-the-art MDP and POMDP solution methods to the problem of collision avoidance.

The experiments we present in this document show that we can model collision avoidance systems using MDPs, and such systems perform very well in terms of both reducing the risk of collision and having very little deviations from the flight plan at the same time, especially with sensors that precisely locate intruder aircraft. We also present experiments with POMDP models built for sensors with limited observation capabilities that demonstrate how we can still achieve low risk of collision by maneuvering a little more in order to counterbalance the limitations in observability of intruder aircraft.

The remainder of this document is organized as follows: Section II reviews MDP and POMDP formulations and solution methods. Section III presents our parametric aircraft and sensor models. Section IV introduces the simulation and evaluation framework used to test collision avoidance systems. Sections V, VI and VII describe our collision avoidance models with increasing complexities to handle different sensor models. Section VIII elaborates on limitations of developed models and suggests ways to improve them. Section IX discusses main conclusions from this work.

II. Review of MDPs and POMDPs

An MDP is a stochastic process where the state of the system changes probabilistically according to the current state and action. POMDPs extend MDPs by including an observation process that probabilistically generates observations conditioned on the current state and action, and hence they have more expressive power. We will briefly review POMDPs in this section.

The solution to a POMDP is a *policy*, or way of behaving, that selects actions in a way that takes into account both the current uncertainty about the underlying state of the system (e.g., exact relative position of the intruder aircraft), as well as future uncertainty about how the system state will evolve (e.g., what kinds of maneuvers the intruder aircraft will make), by aiming to maximize the expected accumulation of some predefined reward (or minimize the expected accumulation of some predefined cost).[2] Due to their rich descriptive power, POMDPs have found many uses in computer science and robotics applications such as robust mobile robot navigation,[3] machine vision,[4,5] robust dialogue management,[6,7] autonomous helicopter control,[8,9] and high-level robot control,[10] as well as in many other areas like machine maintenance,[11] network troubleshooting,[12] medical diagnosis[13] and preference elicitation.[14] Cassandra provides a comprehensive survey of applications utilizing POMDPs.[15]

Several formulations of POMDPs have been studied in the literature, but this work focuses on the discrete-time formulation with discrete state and action spaces. We briefly present below a POMDP formulation and discuss solution techniques.

II.A. Formulation

In this document, we use $\mathcal{S}$ to represent the state space, $\mathcal{A}$ to represent the action space, and Ω to represent the observation space, all assumed discrete. The *state-transition function* $T : \mathcal{S} \times \mathcal{A} \to \Pi(\mathcal{S})$ determines the probability distribution over the next states given the current state and action taken. The probability of transitioning to state s' after taking action a from state s is written $T(s, a, s')$. The *observation function* $O : \mathcal{S} \times \mathcal{A} \to \Pi(\Omega)$ determines the probability distribution over the observations received after taking some action resulting in state s'. The probability of receiving observation o after taking action a and landing in state s' is written $O(s', a, o)$.

In general, the initial state is unknown. The uncertainty in the initial state is represented by a probability distribution $b_0 : \mathcal{S} \to \mathbb{R}$, where the probability of starting in state s is written $b_0(s)$. The space of possible beliefs is denoted $\mathcal{B}$. The belief-state b is initialized to b_0 and updated with each observation according to Bayes' rule. If the current belief-state is b and action a is taken resulting in an observation o, the new belief-state b' is given by

$$\begin{aligned} b'(s) &= \Pr(s' \mid o, a, b) \\ &\propto \Pr(o \mid s', a, b) \Pr(s' \mid a, b) \\ &= \Pr(o \mid s', a) \sum_{s \in \mathcal{S}} T(s, a, s') b(s) \\ &= O(s', a, o) \sum_{s \in \mathcal{S}} T(s, a, s') b(s) \,. \end{aligned}$$

The belief-update process is often referred to as *state estimation.*

Given the current belief-state, the objective is to chose an action that maximizes the *expected discounted return.* The discounted return for a sequence of states s_t and actions a_t is given by

$$\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,, \tag{1}$$

where $\gamma \in [0, 1)$ is a *discount factor* and $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the *reward function.* The reward for taking action a from state s is written $R(s, a)$.

The solution to a POMDP is a *policy* $\pi : \mathcal{B} \to \mathcal{A}$ that specifies which action maximizes the expected discounted reward given a belief-state. It is known that optimal policies can be represented as a collection α-vectors, denoted Γ. Each α-vector is a vector consisting of $|\mathcal{S}|$ components and is associated with a particular action. The expected discounted return when starting with belief b is

$$V(b) = \max_{\alpha \in \Gamma} (\alpha \cdot b) \,, \tag{2}$$

where $\alpha \cdot b$ is the inner product of an α-vector with a vector representation of the belief-state. The function V is known as the *value function.* The policy evaluated at belief-state b is the action associated with the α-vector that maximizes the inner product.

II.B. Solution Methods

Finding the collection of α-vectors that represents the optimal policy can be challenging, even for relatively small problems. A variety of exact solution methods can be found in the literature, but generally these methods do not scale well to large problems. Approximate solution methods generally scale much better and many of them provide bounds on the *regret* for the policies they find. The regret of a policy π is the difference between the expected discounted return starting at b_0 when following π and the expected discounted return starting at b_0 when following an optimal policy π^*.

In recent years, point-based methods for finding approximate solutions to POMDPs (for example, Point-Based Value Iteration, PBVI[16]) have received attention because of their ability to solve problems that are orders of magnitude larger than was previously possible. Point-based methods involve sampling from the belief space $\mathcal{B}$. In this work we initially used solvers based on Heuristic Search Value Iteration (HSVI2) algorithm.[17] We later switched to a solver that uses Successive Approximations of the Reachable Space under Optimal Policies (SARSOP) algorithm[18–20] as it performed better on our problems. An implementation of SARSOP is publicly available[a] and we were able to use the software without any modification. SARSOP takes as input a standard textual representation of a POMDP, including γ, b_0, R, T, and O. When the regret bounds fall below some preset value or the user interrupts the solution process, SARSOP outputs a policy file represented as a collection of α-vectors.

Crucially, although it may require considerable computation to find a near-optimal policy, this work is done offline. Once a policy has been computed, it can be executed very efficiently online. In the course of this work we have developed a new algorithmic technique to make the execution process even more efficient, making it entirely suitable for execution online, in real time, on an aircraft.

III. Aircraft and Sensor Models

The aircraft model we developed for our collision avoidance systems is parametric and can be modified to mimic different types of aircraft. In our implementation, parameter values are based on Global Hawk, an unmanned aerial vehicle (UAV) used by the United States Air Force as a surveillance aircraft. Table 2 shows performance limits for Global Hawk. Our collision avoidance models use a subset of these values; maximum and minimum velocities and maximum climb/descent rates.

Table 2. Global Hawk performance limits.

Maximum velocity	180 kts
Minimum velocity	100 kts
Maximum climb rate	3500 fpm
Maximum descent rate	4000 fpm
Maximum bank angle	35 deg
Maximum bank rate	8 deg/s
Maximum pitch rate	2 deg/s
Maximum turn rate	2.5 deg/s

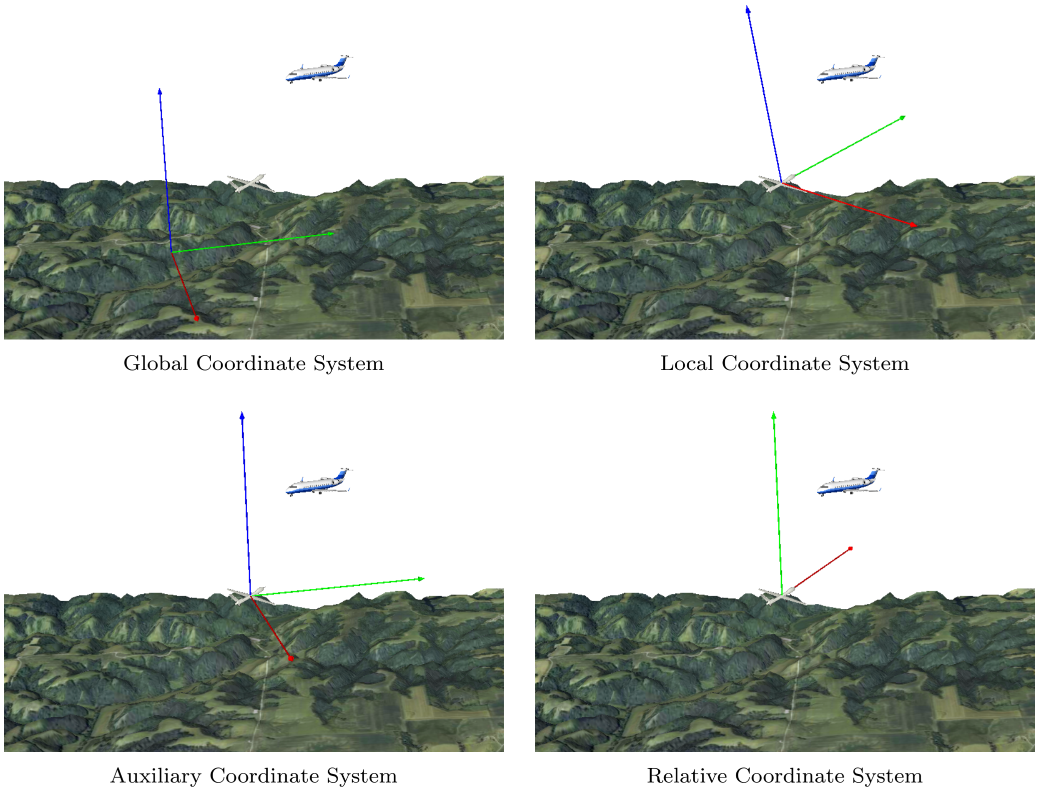
Before describing our sensor models, let us introduce four coordinate systems shown in Figure 1 that we will refer to in the rest of this document:

- **Global Coordinate System (GCS)**: This coordinate system is also known as the *Earth Coordinate System*. The origin is an arbitrary point chosen by the model simulation and evaluation framework. Positive x is *east*, positive y is *north*, and positive z is *altitude*.
- **Local Coordinate System (LCS)**: The origin of LCS is ownship center of mass (i.e., LCS is an *egocentric* coordinate system). Positive x is in the direction of the right wing, positive y is the direction of the nose, and positive z is upwards.
- **Auxiliary Coordinate System (ACS)**: This is also an *egocentric* coordinate system whose x-y-z axes are aligned with the *east-north-altitude* axes of GCS, respectively.
- **Relative Coordinate System (RCS)**: This is another *egocentric* coordinate system which is obtained by rotating ACS around its z axis until the y-z plane intersects with intruder aircraft center of mass. RCS is a 2-dimensional coordinate system. The x and y axes of RCS are the y and z axes of the rotated ACS, respectively. The RCS is also referred to as the projection plane.

Input to our collision avoidance systems may come from various sensors with different characteristics and sensing ranges (usually expressed by radii in nautical miles, *NM*) onboard the UAV. We developed four detailed sensor models that are capable of simulating following types of erroneous measurements and noise:

- **False positive measurements**: We may detect an intruder when, in fact, there is no intruder aircraft in the sensor range (for example, a bird in sensing range might cause false positive measurements).

[a] M^{2}AP Research Group at NUS, *POMDP Planning*, http://motion.comp.nus.edu.sg/projects/pomdp/pomdp.html (December 2009).



Global Coordinate System

Local Coordinate System

Auxiliary Coordinate System

Relative Coordinate System

Figure 1. Coordinate systems.

- **False negative measurements**: We may fail to detect an intruder when one is present in the sensor range.
- **Measurement errors**: We may detect the intruder aircraft in a position or at an angle that is not correct.

The probabilities of false positive and false negative measurements (p_{fp} and p_{fn}) are usually specific to different sensor hardware, and the measurement errors are computed according to realistic error models.

The four sensor models studied in this research are as follows:

1. **Perfect sensor**: This is a hypothetical omnidirectional sensor with no noise and no false positive/negative detections ($p_{fp} = p_{fn} = 0$). The sensor reading consists of *east*, *north* and *altitude* coordinates of intruder aircraft in GCS. With this sensor, it is possible to localize intruder aircraft to an exact point in either GCS or LCS.

2. **TCAS sensor**: This is a model of the actual TCAS sensor.[21] It is based on listening to transponder replies from nearby aircraft and is omnidirectional. It provides *bearing* in LCS, *altitude* in GCS, and *range* (the line-of-sight distance between ownship and intruder aircraft, also referred to as *slant range*). The error in *range* measurement is Gaussian with zero mean and 50 ft standard deviation. The error in *bearing* estimate is Gaussian with zero mean and 10 deg standard deviation. The altitude of intruder aircraft is measured with 25 ft quantization. There is also an altimetry error bias that remains constant during an encounter with an intruder aircraft, and is Laplacian with zero mean and 40 ft scale. Probability density function for the Laplace distribution is shown in Figure 2. In the TCAS sensor model, $p_{fp} = 0$ (since detection is based on broadcast signals) and $p_{fn} = 0.01$. With a noiseless TCAS sensor, intruder aircraft could be localized to a point in LCS, but considering the given error model, the region that the intruder could be residing in has approximately the shape of a distorted truncated spherical cone.

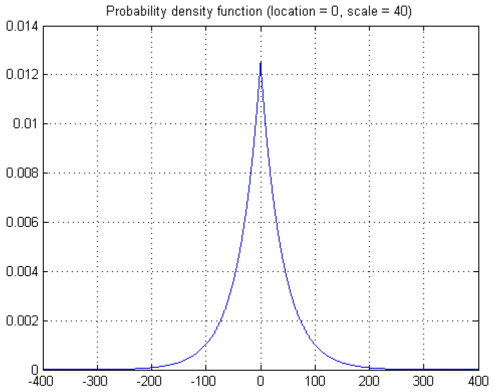


Figure 2. Probability density function for the Laplace distribution (location = 0, scale = 40).

3. **Radar sensor**: Our radar sensor model has a limited field-of-view (FoV), ±15 deg elevation and ±110 azimuth. It provides *bearing* and *elevation* readings in LCS, and *range* and *range rate* information. As with TCAS, the error in the *range* measurement is Gaussian with zero mean and 50 ft standard deviation. *Range rate* error is Gaussian with zero mean and 10 ft/s standard deviation. The error in the *bearing* estimate is Gaussian with zero mean and 10 deg standard deviation. *Elevation* error estimate is Gaussian with zero mean and 1 deg standard deviation. For the radar sensor, $p_{fp} = p_{fn} = 0.01$. Intruder aircraft can be localized approximately into a distorted truncated spherical cone in LCS.

4. **Electro-optical/infrared (EO/IR) sensor**: Our EO/IR sensor model is very similar to the radar sensor with less angular measurement noise and without a *range* reading. It has a limited FoV, ±15 deg elevation and ±110 azimuth. Sensor reading consists of *bearing* and *elevation* angles in LCS, and *line-of-sight rate* information. Error in both angular measurements is Gaussian with zero mean and 0.5 deg standard deviation. *Line-of-sight rate* error is Gaussian with zero mean and 0.5 deg/s standard deviation. For the EO/IR sensor, $p_{fp} = p_{fn} = 0.01$. Intruder aircraft can be localized approximately into a distorted spherical cone in LCS.

Parameters are given in Table 3, and sensor capabilities are summarized in Table 4.

IV. Simulation and Evaluation Framework

The performance of our collision avoidance systems were evaluated using a simulation framework developed for prior TCAS studies[22] and sense-and-avoid systems for unmanned aircraft.[23] We used an encounter model derived from 9 months of national radar data[24] to generate 15,000 scripted encounters between pairs of aircraft and allowed our collision avoidance systems to control of one of the aircraft. For comparison, we evaluated the performance of other collision avoidance systems to baseline performance. This section describes our simulation and evaluation process.

IV.A. Simulation Framework

Figure 3 provides an overview of the simulation framework. An encounter model is used to generate initial conditions and scripted maneuvers for both aircraft involved in the encounter. These initial conditions and scripts are fed into a 6 degree-of-freedom, point-mass dynamic model. The sensor model takes as input the current state from the dynamic model and produces an observation, or sensor measurement. The state estimation process updates the belief-state based on the observation. The MDP/POMDP policy is evaluated on the updated belief-state and an optimal, or approximately optimal, action is executed. The dynamic model updates the state, and the process continues until the end of the encounter.

IV.B. Baseline Collision Avoidance Systems

We compared the performance of our system against the following three baseline systems:

Table 3. Complete list of sensor parameter values.

Perfect		
Range	5	NM
False positive measurement probability	0.00	
False negative measurement probability	0.00	
TCAS		
Range	5	NM
Altitude quantization	25	ft
Range error standard deviation	50	ft
Bearing error standard deviation	10	deg
Altimetry error scale	40	
False positive measurement probability	0.00	
False negative measurement probability	0.01	
Radar		
Range	5	NM
Minimum azimuth	−110	deg
Maximum azimuth	110	deg
Minimum elevation	−15	deg
Maximum elevation	15	deg
Range error standard deviation	50	ft
Bearing error standard deviation	1	deg
Elevation error standard deviation	1	deg
Range rate error standard deviation	10	ft/s
False positive measurement probability	0.01	
False negative measurement probability	0.01	
EO/IR		
Range	5	NM
Minimum azimuth	−110	deg
Maximum azimuth	110	deg
Minimum elevation	−15	deg
Maximum elevation	15	deg
Bearing error standard deviation	0.5	deg
Elevation error standard deviation	0.5	deg
Line-of-sight rate error standard deviation	0.5	deg/s
False positive measurement probability	0.01	
False negative measurement probability	0.01	

Table 4. Sensor capabilities. Green, yellow, and red indicate good, moderate, and poor performance, respectively.

	Perfect	TCAS	Radar	EO/IR
Range		🟩	🟩	
Bearing		🟥	🟩	🟩
Altitude	🟩	🟨	🟩	
Elevation			🟩	🟩
Range rate			🟩	
Line-of-sight rate				🟩
Position	🟩			

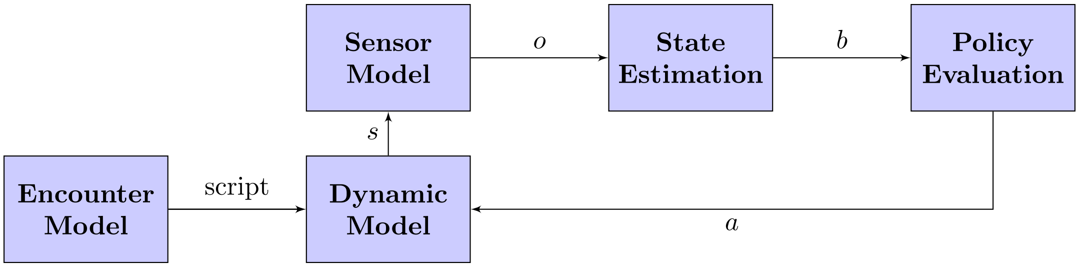


Figure 3. Simulation framework.

- **TCAS Version 7:** The TCAS Version 7 system uses only the TCAS sensor readings as input. The behavior of this system is as specified in the TCAS II standard.[21]
- **Basic Collision Avoidance System (Basic CAS)**: It is possible to use all four sensor models with Basic CAS, but the performance decreases severely with the limited field-of-view sensors. The collision avoidance logic is very simple: If an intruder aircraft is detected inside the sensing region, and the projection of the intruder position on RCS has a positive y value (i.e., the intruder is "above"), then ownship accelerates down with 0.25 g until next observation is received. Similarly, if the intruder is "below" (projection of its position on RCS has a negative y value), then ownship accelerates up with 0.25 g until the next observation.
- **Analytic Collision Avoidance System (Analytic CAS)**: Analytic CAS is based on collecting position data for ownship and intruder aircraft, and estimating their motion (velocities and accelerations) in full 3-dimensional coordinates by simple differentiation. Therefore, it is best suited for use with perfect and TCAS sensors, which are omnidirectional, have none or very little noise (compared to other sensor models) and hence allow the intruder to be localized with high accuracy at each simulation step. The collision avoidance logic works as follows: Based on regularly collected and updated position, velocity and acceleration estimates, a *clear-of-danger* test is performed using simple quadratic equations of motion at each simulation step. If there is no danger of a collision or a close encounter in the future, ownship continues to follow the scripted maneuver, but if the test fails (i.e., the minimum distance between the extrapolated trajectories of both aircraft is below some threshold), an evasive maneuver is performed, which is simply to increase ownship's altitude by 200 ft as quickly as possible within the performance limits. After the maneuver is completed, the collision avoidance logic resumes tests and triggers further evasive maneuvers as necessary. We implemented two versions of the *clear-of-danger* test: The first version, called Analytic CAS 1-D, checks if only the vertical distance between two aircraft will drop below a threshold, and the second version, called Analytic CAS 3-D, checks if the intruder will invade a predefined 3-D volume surrounding ownship (which is usually in the shape of a hockey puck that is 200 ft thick and 1000 ft in diameter).

V. Perfect Sensing

The first case we will consider is sensing with no noise, and for that purpose we will assume that ownship is equipped with a perfect sensor. When there is no observation uncertainty, we can model the collision avoidance system as an MDP. Note that we allow uncertainty about the behavior of the intruder aircraft, and MDP formulation lets us capture this uncertainty in the state-transition model. In this section, we will look at the general structure of the state and action spaces and the details of the reward and state-transition models that will compose our MDP collision avoidance system.

V.A. MDP Collision Avoidance System

The true state space model in the collision avoidance problem is continuous and consists of the following components for both aircraft present in the encounter:

- Position specified in GCS
- Orientation specified as yaw, pitch and roll angles

- Air speed, air speed acceleration
- Vertical rate, vertical acceleration
- Yaw rate, pitch rate and roll rate

This set of components is referred to as the *aircraft state vector.* This is a very high-dimensional continuous space (26 dimensions for both aircraft together). The action space for a UAV is also continuous as it is possible to choose and apply any vertical and/or horizontal accelerations within ownship's performance limits.

In this work, we consider a simplified version of the problem in which ownship can only maneuver vertically, but not in azimuth, to evade intruders, similar to TCAS II. We also work with discretized spaces with less number of dimensions that are carefully selected to incorporate important information from the true spaces.

V.A.1. State Space

The size of a discretized state space is exponential in the dimension and in the case of 26 dimensions, we could not stand to have even two discrete values per dimension. So, before we discretize the state space, we must first represent it in a much lower-dimensional subspace that captures the essence of the encounter.

To encode relative positions and velocities of the aircraft, we chose RCS as our main representation. In this coordinate system, the state consists of the following components:

- X: horizontal distance from ownship to intruder aircraft;
- Y: vertical distance from ownship to intruder aircraft;
- *RelativeVx*: (relative) velocity in X, representing the horizontal closure rate;
- *OtherVy*: vertical velocity of intruder aircraft; and
- *OwnVy*: vertical velocity of ownship.

This 5-dimensional state space is discretized by dividing each dimension into a finite number of *bins.* The sizes of the bins may be non-uniform. The overall state-space is then a set of 5-orthotopes (5-dimensional boxes or hyperrectangles) that exhaust a continuous piece of the overall 5-dimensional state space. We augment the state space with two sets of special states: START states and DONE states. These states are used to model situations when the state space is initialized (and the encounter has not started), and when the encounter is over, respectively. Because the vertical velocity of ownship is always known, we always include it in the state space. So, the START and DONE state sets both contain a member for each bin of *OwnVy*, modeling flight at some vertical velocity before the start of or after the termination of, an encounter. Having discretized the state space in this way, a state may be represented simply as an index into the set of boxes spanning the space, or an index to one of the START or DONE states.

V.A.2. Action Space

We adopted a simple discrete action-space model that consists of commands to ownship to apply positive or negative fixed vertical accelerations for a fixed duration (usually 1 s). For the MDP CAS, our action space consists of 17 uniform samples from the $\pm 8\,\mathrm{ft/s^2}$ ($\pm 0.25\,\mathrm{g}$) acceleration range imposed by the aircraft performance limits; $\mathcal{A} = \{-8, -7, \cdots, -1, 0, 1, \cdots, 7, 8\}$. It is possible to sample the range of vertical accelerations more densely, but the solvers would require more time to find policies with tight regret bounds.

V.A.3. Reward Model

The reward function in our MDP formulation is in the form of costs (or negative rewards) rather than positive rewards. It is designed with the following three objectives in mind:

- As the primary goal of the collision avoidance algorithm, the intruder aircraft should never occupy the same bin as ownship in the RCS, which implies a collision or a very dangerous encounter. Note that ownship resides at the origin of the RCS, and it is possible that the origin might be on the edge or vertex of one or more bins rather than being inside a single bin due to the chosen vertical and horizontal division strategy. In that case, the collision avoidance algorithm should prevent the intruder from moving into any one of the bins that have any boundaries touching the origin.

- In addition to preventing collision, it is desirable to maintain some protected airspace around ownship where the intruder aircraft should not penetrate. This protected airspace is specified by two parameters: a vertical separation range and a horizontal separation range. In our tests for MDP CAS, we used 100 ft vertical and 500 ft horizontal separation ranges, same as that of the NMAC definition used in prior TCAS safety studies.[25–28] The second goal of the collision avoidance algorithm should be to prevent other aircraft moving into any bin that has some parts overlapping with the protected airspace.

- As the last goal, if there is no danger of collision or penetration of protected airspace, ownship should level off and try to maintain a zero vertical velocity. It may be argued that ownship should try to return to its commanded flight path. We have taken the position that, during the handling of a close encounter, it is enough to prefer level flight, and that after the encounter is over, standard navigational procedures can be resumed.

In order to satisfy these goals, the reward may be specified as a function of the state of the system. It is specified using three user-defined parameters:

- **Collision cost**: The cost of any state in which the intruder is in the same X and Y bins as ownship, currently set to -1000;

- **Protected airspace violation cost**: The cost of any state in which the intruder aircraft is within the protected airspace region in X and Y, currently also set to -1000; and

- **Vertical velocity penalty**: The cost for being in a state where the *OwnVy* bin does not contain 0 ft/s; for the MDP CAS, vertical velocity penalties are linearly proportional to the velocity values that correspond to the centers of the *OwnVy* bins. It is possible to vary the maximum penalty value in order to reach different equilibria in balancing evasive maneuvers and level flight.

All other states are assumed to have a reward of 0. Note that the solution to the MDP will remain the same for any linear scaling of reward values, so only the relative magnitudes have an effect.

V.A.4. State-Transition Model

The initial state distribution specifies that the system starts in a uniformly chosen START state. At each step, an action is taken and the probability distribution over the state space is updated according to the state-transition model.

Our assumption is that there is no actual stochasticity in the dynamics of the system. However, we model the uncertainty in intruder behavior as a random process; and the fact that the state space is discretized will introduce uncertainty in the transitions, even though they are governed by a deterministic physical process.

Our state-transition model is characterized by the following parameters:

- Controller frequency, ΔT: Duration between successive consultations of the MDP policy for choosing an action. This value is used by the MDP formulation to predict what the state will be in the next iteration.

- Magnitude of our vertical acceleration, *OwnAy*.

- Our vertical velocity limits, *OwnVyMin* and *OwnVyMax*.

- Probability of staying in START state when already in START state.

- Probability of making a transition into any other state when in START state.

- Intruder aircraft's horizontal and vertical acceleration models.

For the horizontal and vertical acceleration models, we used the distributions given in Table 5. These distributions roughly model a random walk process where the intruder aircraft is oblivious to ownship or we have no idea about the intention of the intruder aircraft.

Table 5. Horizontal and vertical acceleration models for intruder aircraft.

Horizontal Model		Vertical Model	
$\dot{v}$ (ft/s^2)	Probability	$\dot{v}$ (ft/s^2)	Probability
−300.0	0.05	−10.0	0.1
−200.0	0.05	−5.0	0.2
−100.0	0.05	0.0	0.4
−30.0	0.10	5.0	0.2
−20.0	0.10	10.0	0.1
−10.0	0.10		
0.0	0.10		
10.0	0.10		
20.0	0.10		
30.0	0.10		
100.0	0.05		
200.0	0.05		
300.0	0.05		

Given these parameters, we compute $\Pr(s' \mid s, a)$ as follows:

- First, we consider each possible pair of vertical and horizontal accelerations a_o that might be chosen by the intruder aircraft, and compute their probabilities p_o as the product of the probabilities in the intruder acceleration models.

- For each vertex of the bin s, we determine how that particular point in state space would be transformed given the execution of ownship acceleration a, and the intruder accelerations a_o.

- The result is a new box, B, in 5-dimensional space. For each new state s', we compute the percentage of B that overlaps s'; that overlap percentage is $\Pr(s' \mid s, a, a_o)$. Any probability mass outside the boundaries of the modeled state space is assigned $\Pr(\text{DONE}\ , \mathit{OwnVy} \mid s, a, a_o)$.

- Finally,
$$\Pr(s' \mid s, a) = \sum_{a_0} \Pr(s' \mid s, a, a_o) p_o\,.$$

This method of computing the physical evolution of the system analytically eliminates introducing additional discretization in the computation. Therefore, the effectiveness of the state-transition model depends only on the discretization of the state and action spaces and the fidelity of the vertical and horizontal acceleration models for the intruder aircraft. Having the acceleration models match closely to the actual intruder behaviour results in better state estimations, where the intruder aircraft would be localized more accurately.

V.B. Results

Table 6 summarizes the results of nominal flight (ownship following the scripted flight path without using any collision avoidance systems) and baseline collision avoidance systems on 15,000 encounters. The table shows the risk ratios, mean vertical velocity magnitudes in ft/s, and mean vertical acceleration magnitudes in ft/s^2 for different algorithms. The risk ratio associated with a particular system is the probability that an encounter leads to an NMAC using the system divided by the probability that an encounter leads to an NMAC without the system. Of course, better performance is indicated by a small risk ratio. It is desirable to have velocity and acceleration values as small as possible without sacrificing the risk ratio. Large values of mean velocity magnitude and mean acceleration magnitude indicate that ownship is maneuvering unnecessarily.

We experimented with gradually increasing the size of the state space (by increasing the number of bins along different dimensions in our discretization) until the time it takes for the solver to compute a policy increases beyond practical limits, and we ended up with an MDP model with 6768 states: 5, 10, 3, 5 and 9 bins for X, Y, $\mathit{RelativeVx}$, $\mathit{OtherVy}$ and OwnVy components of $\mathcal{S}$, respectively, and 9 START and 9 DONE

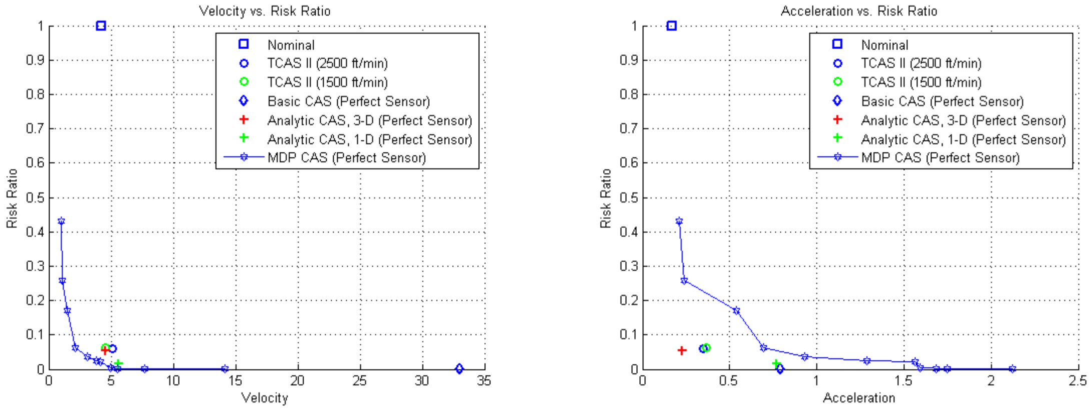
Table 6. Risk ratios for nominal flight and baseline collision avoidance systems.

	Ratio	Velocity	Acceleration
Nominal	1.000000	4.255460	0.172020
TCAS II (2500 ft/min)	0.061220	5.094360	0.345920
TCAS II (1500 ft/min)	0.062730	4.586190	0.366110
Basic CAS (perfect sensor)	0.000010	33.030760	0.790190
Analytic CAS, 3-D (perfect sensor)	0.054560	4.564330	0.224730
Analytic CAS, 1-D (perfect sensor)	0.016970	5.597470	0.768990

states. Solving an MDP using *value iteration*[29] is very efficient especially if the solver is implemented using sparse data structures. Therefore, instead of testing a single instance of an MDP, we were able to vary the vertical velocity penalty (reward) and generate multiple instances of our MDP CAS model to trace out system performance (SP) curves. SP curves are similar in nature to system operating characteristic (SOC) curves,[30–32] which generally involve plotting unnecessary alert against successful alert. Results for our MDP CAS is given in Table 7 and Figure 4 shows SP curves pertaining to our MDP model. In the SP curves, points close to the origin are more desirable as they represent low risk ratios and low velocity/acceleration values (less maneuvering), and our MDP model scores better than the other systems on the Velocity - Risk Ratio curve.

Table 7. Risk ratios for MDP collision avoidance system (perfect sensor).

Reward	Ratio	Velocity	Acceleration
−0.10	0.000692	14.174462	2.121009
−0.50	0.000980	7.721526	1.684897
−0.75	0.001428	5.505732	1.745723
−1.00	0.003075	4.970565	1.591075
−1.25	0.022785	4.133050	1.566663
−1.50	0.024709	3.820564	1.286228
−2.00	0.036734	3.125315	0.931763
−5.00	0.063469	2.159921	0.691902
−10.00	0.170806	1.460390	0.539181
−20.00	0.257840	1.059476	0.241147
−30.00	0.431986	0.973162	0.212496

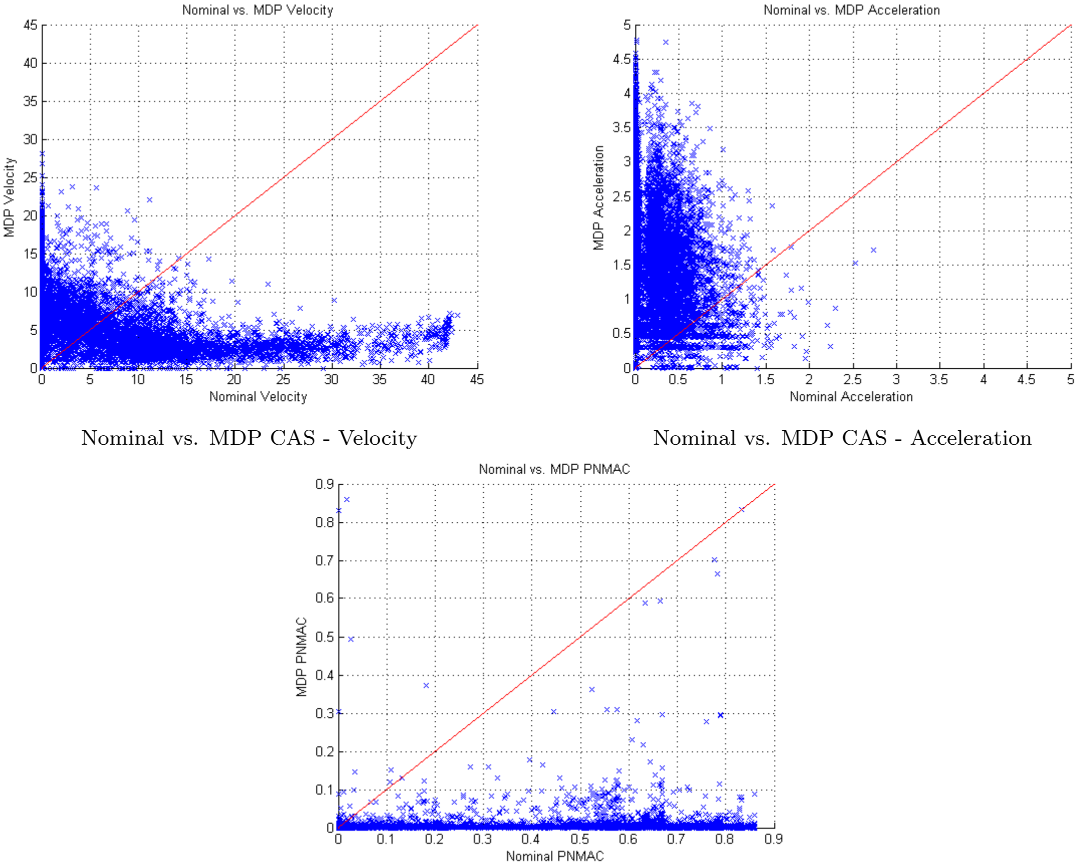


Velocity vs. Risk ratio.

Acceleration vs. Risk ratio.

Figure 4. System performance curves for MDP collision avoidance system.

Graphs displaying velocity, acceleration and probability of NMAC (PNMAC) values from 15,000 encounters using nominal flight strategy plotted against values from same encounters using our MDP collision avoidance logic are shown in Figure 5. In these graphs, scoring below the (red) diagonal are desirable as it indicates that an aircraft equipped with our collision avoidance system performs better (in terms of lower risk ratio or less maneuvering) than an aircraft that just follows the scripted maneuver for that particular encounter scenario. Note that our reward model is constructed to optimize velocities, therefore acceleration plots are not significant for our experiments in general, but presented as a reference. Also, in the PNMAC comparison, there are a few encounters where the MDP PNMAC is higher than Nominal PNMAC (points above the diagonal), which means that the collision avoidance system actually increased the risk ratio. This can happen in the following case: intruder aircraft performs a dangerous altitude crossing maneuver that happens to have a large miss distance when ownship follows the scripted maneuver, but the miss distance is small when ownship follows the commands from the collision avoidance system that is trying to avoid the same dangerously close intruder where the intruder behavior is modeled by a random walk process.



Nominal vs. MDP CAS - Velocity

Nominal vs. MDP CAS - Acceleration

Nominal vs. MDP CAS - PNMAC

Figure 5. Velocity, acceleration and PNMAC graphs, Nominal vs. MDP CAS.

Since our reward model is constructed with penalizing high vertical velocities as our goal, it is not surprising that we do not get low acceleration values as opposed to the optimization we get with velocities. In fact, the MDP CAS prefers using high acceleration values. The histogram in Figure 6 shows the total number of states an action is chosen as the *best action* by the MDP policy (this specific policy was generated with vertical velocity penalty $= -2.00$).

In conclusion, we can say that MDP CAS works well in the case of perfect sensing, and we can easily outperform baseline collision avoidance systems in terms of much lower risk ratios and velocities (without unnecessary maneuvering).

VI. Noisy Sensing

Our second case is omnidirectional sensing with noise, and we will use the TCAS sensor model as our input source. If we were to make use of the *bearing* estimate produced by the TCAS sensor in locating the intruder aircraft in any 3-dimensional coordinate system, the error might be considerably big (especially with distant intruders). However, we chose to work with projections of intruder aircraft on RCS and hence

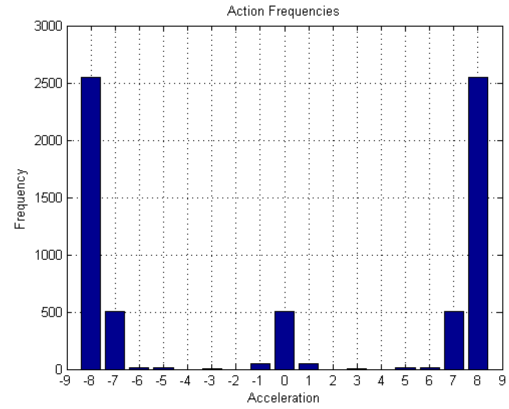


Figure 6. Frequencies of best actions in MDP policy (vertical velocity penalty = -2.00).

we do not need the *bearing* estimate at all. It is possible to accurately locate the intruder on RCS using other TCAS readings. This gives us the following two options in designing a collision avoidance system that uses the TCAS sensor:

- We treat this as a perfect sensing problem and use the same MDP model developed in Section V. To figure out the state, we either directly use the sensor reading neglecting the fact that it is noisy, or we use an estimator such as an alpha-beta tracker[33] or a Kalman Filtering based technique.[34] Since small observation noise does not affect action selection much in this specific problem, an alternative to using an external state estimator is to formulate the problem as a $\mathrm{Q}_{\mathrm{MDP}}$.[35]
- We define a discretized observation space Ω, and design an observation model for the TCAS sensor to augment the MDP model of Section V, and turn the problem into POMDP planning.

In this section, we first present results for an MDP collision avoidance model using an alpha-beta tracker to estimate the state, and then we look at a POMDP model.

VI.A. MDP Collision Avoidance System with State Estimator

The results for baseline collision avoidance systems with the TCAS sensor are shown in Table 8. Using a simple alpha-beta tracker for state estimation with $\alpha = \beta = 0.5$, we obtained the results in Table 9 with our MDP collision avoidance system for various vertical velocity penalty values. The SP curves are shown in Figure 7.

Table 8. Risk ratios for baseline collision avoidance systems (TCAS sensor).

	Ratio	Velocity	Acceleration
Basic CAS (TCAS sensor)	0.000010	32.909700	1.034700
Analytic CAS, 3-D (TCAS sensor)	0.080100	7.402750	1.096570
Analytic CAS, 1-D (TCAS sensor)	0.020500	19.557490	4.511640

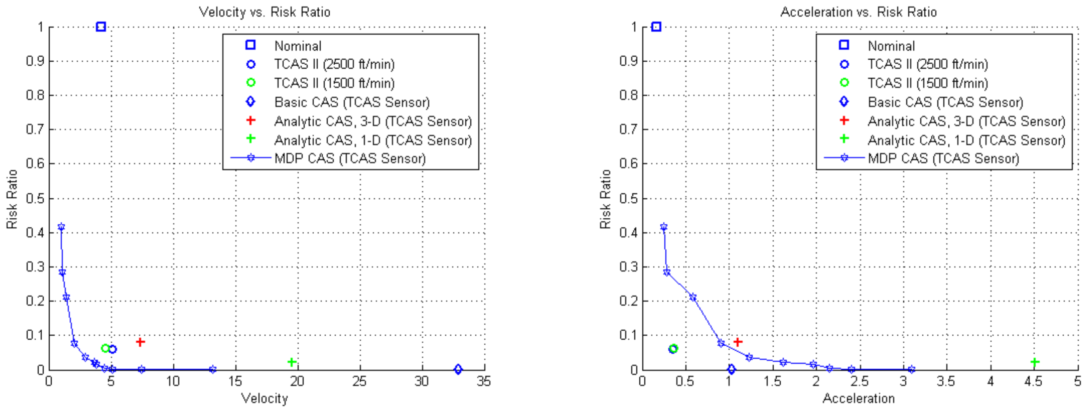
Even though alpha-beta tracking is a very simple state estimation method, the results are satisfactory. Using Kalman filters, interacting multiple model methods,[36] or nonlinear filters[37] may further improve the quality of state estimation.

VI.B. POMDP Collision Avoidance System with TCAS Sensor

The state space $\mathcal{S}$, and the state-transition model we built in Section V effectively capture important aspects of the encounter geometry and motion dynamics for both aircraft, respectively. Therefore, a POMDP collision avoidance model can be built on top of the MDP model of Section V by just adding an observation model. In this section, we will define the observation space and the observation model for the TCAS sensor, and we will also look at how we can slightly modify action space and reward model together to reduce the POMDP size and still obtain low risk ratios.

Table 9. Risk ratios for MDP collision avoidance system (TCAS sensor).

Reward	Ratio	Velocity	Acceleration
−0.10	0.000916	13.225057	3.088738
−0.50	0.001717	7.431411	2.404085
−0.75	0.002428	5.101627	2.398184
−1.00	0.003337	4.494725	2.151822
−1.25	0.015149	3.857991	1.967395
−1.50	0.023313	3.657201	1.618871
−2.00	0.037456	2.906691	1.221383
−5.00	0.077662	2.033404	0.902261
−10.00	0.212924	1.448597	0.576285
−20.00	0.285638	1.055002	0.284902
−30.00	0.415815	0.993773	0.243202



Velocity vs. Risk ratio.

Acceleration vs. Risk ratio.

Figure 7. System performance curves for MDP collision avoidance system.

VI.B.1. Observation Space

The discrete model of the observation space is constructed in a way similar to the discrete state space. There are two types of observational information: vertical velocity of ownship (*OwnVy*, which we assume is always completely and correctly observed), and possible single detection of an intruder aircraft using a sensor system. Observations for the TCAS sensor are discretized into the same bins as the X and Y components of the state space. The model could easily be changed to provide observations at a higher or lower granularity. In addition, there is a special *noObs* observation for the case when no intruder is detected (due to either an empty sensing region or a false negative measurement).

VI.B.2. Observation Model

The observation model of a POMDP specifies $\Pr(o \mid s, a)$, that is, the conditional probability of making each possible observation o, given that the actual state is s and the last action was a. All necessary information is encapsulated in s, so we will ignore dependence on a, and specify $\Pr(o \mid s)$ for all discrete o and s.

We assume that, at every step, the observation has two components: o_{ovy}, our measured vertical velocity, and o_d, the observed detection of the intruder, and that these are independent, so

$$\Pr(o_{ovy}, o_d \mid s) = \Pr(o_{ovy} \mid s) \Pr(o_d \mid s) \, .$$

The measurement of our vertical velocity is always correct, so $\Pr(o_{ovy} \mid s) = 1$ if o_{ovy} is equal to the *OwnVy* component of s, and 0 otherwise.

The observed detection is more complex due to false positive/negative measurements and measurement errors described in Section III. We assume fixed probabilities for false positives p_{fp} and false negatives p_{fn}, and assume that if there is a false positive detection, it is generated with uniform probability over the space of values of o_d.

When s is a START or DONE state (the encounter has not yet begun or has terminated) or when $Y > \mathit{maxRange}$, that is, when the distance to the other aircraft is greater than the range of the sensor, then $\Pr(o_d = \mathit{noObs} \mid s) = 1 - p_{fp}$. That is, with high probability, the observation is *noObs*. We used a value of 5 nautical miles for *maxRange* for all sensors. For any other observation $\Pr(o_d = d \mid s, \mathit{fp}) = |O_d|^{-1}$; that is, it is uniform over the space of possible actual detection observations.

Finally, if the intruder is within the modeled volume of the state space, there is some chance of not seeing the intruder: $\Pr(o_d = \mathit{noObs} \mid s) = p_{fn}$. Otherwise, with probability $1 - p_{fn}$, we make a detection d.

A precautionary *margin* is added to all four sides of the X, Y rectangle corresponding to the detection d. Then we consider all of the X, Y bins b_i that overlap the expanded detection bin, and the proportion of the expanded detection bin that overlaps b_i, called p_i. So,

$$\Pr(o_d = d \mid s) = (1 - p_{fp}) p_i + p_{fp} |O_d|^{-1} \quad ,$$

for any state in which the intruder is in X, Y bin b_i, for all bins b_i, and

$$\Pr(o_d = d \mid s) = p_{fp} |O_d|^{-1}$$

otherwise. For the TCAS sensor, we can define the *margin* in terms of standard TCAS sensor error parameters given in Table 3:

$$\begin{aligned} \mathit{margin} \quad = \quad & \text{Altitude quantization} + \\ & 3 \times \text{Range error standard deviation} + \\ & 3 \times \text{Altimetry error scale} \end{aligned}$$

Including full altitude quantization and 3 standard deviations worth of error in the *margin* gives us an unnecessarily conservative confidence region around the detection d which can, in fact, hinder intruder localization and render the observation model useless. The margin should be large enough so that it covers the region from which a noisy sensor reading may have originated, but it should be small enough to allow the POMDP to properly localize the intruder. Therefore, we used smaller margins in our experiments (half of altitude quantization and 0.5 standard deviations gave us reasonable risk ratios).

VI.B.3. Modifications

POMDP solvers work with belief-states instead of exact states and branch on actions and observations, therefore their memory and time demands are typically much higher than MDP solvers, especially if we would like to compute policies with tight regret bounds. A POMDP model with the same state and action spaces as the MDP model of Section V takes days to just initialize and generate the first heuristic policy in the iterative improvement process. We describe below how the parametric design of our POMDP model gave us leverage to reduce the size without decreasing performance.

As depicted in Figure 6, the MDP collision avoidance logic mostly uses very high, very low or zero acceleration options available when picking an action. This is in accordance with our reward model. Based on this observation, we used a new and smaller action space with only three actions, $\mathcal{A} = \{-8, 0, 8\}$, which correspond to accelerating up/down with maximum magnitude or maintaining vertical velocity.

We also used a slightly different discretization for the state space: 7, 10, 4, 4 and 3 bins for X, Y, *RelativeVx*, *OtherVy* and *OwnVy* components of $\mathcal{S}$, respectively, and 3 START and 3 DONE states, which bring the number of states down to 3366.

Based on some experimental results, we modified the reward model as follows: we increased the size of the protected airspace around ownship to 200 ft vertical and 1000 ft horizontal separation, and we set *protected airspace violation cost* to -500.

These modifications let the SARSOP solver initialize in about an hour and generate acceptable policies (in terms of low risk ratios) in 3 to 5 hours.

VI.C. Results

Tracing out SP curves for POMDP models is very time consuming, and essentially the more the solver runs, the better the generated policies perform. Therefore, we present the single best result we obtained for our POMDP model (in terms of low risk ratio) using a vertical velocity penalty of -0.1 in Table 10. The POMDP collision avoidance logic for the TCAS sensor is about 20 times safer than TCAS Version 7 currently used on manned aircraft. However, TCAS has a much lower mean vertical velocity magnitude, indicating that it maneuvers less frequently.

Table 10. Risk ratio for POMDP collision avoidance system (TCAS sensor).

	Ratio	Velocity	Acceleration
POMDP CAS (TCAS sensor)	0.002770	14.133030	1.759190

Although we use the same sensor model of the TCAS algorithm for our POMDP model and constrain the vertical rate magnitude to be within 2500 ft/min, the comparison is not entirely fair. TCAS was designed for pilot-in-the-loop control and assumes a delay between when the resolution advisory is issued and when the pilot responds. Although the POMDP algorithm has the advantage over the TCAS algorithm because it can maneuver instantaneously, the TCAS algorithm is permitted to make up to 0.35 g maneuvers whereas the POMDP was constrained to 0.25 g maneuvers. We use the standard model of pilot response to TCAS resolution advisories, which is a 0.25 g acceleration after a 5 s delay for the initial advisory and a 0.35 g acceleration after a 2.5 s delay for subsequent advisories.[38] Although a direct comparison between the POMDP model and TCAS algorithm cannot be made, we can be confident, at least, that the POMDP is performing well.

Considering the MDP model results and comparing both risk ratio and flight plan adherence, we conclude that, for the TCAS sensor, an MDP model is the right choice.

VII. Limited Field-of-View Sensing

As our final case, we look at POMDP collision avoidance using radar and EO/IR sensors. Both of these sensors have noise, and are effective only within a limited sensing region. Most important complications caused by these two sensors are the following:

- Unlike the TCAS sensor which provides an accurate *altitude* reading in GCS, these sensors provide *elevation* estimates that we need to use when projecting the intruder aircraft location on RCS. Using an angular measurement makes it difficult to localize distant intruders in RCS, so until the intruder is sufficiently close, the altitude estimate will not help the POMDP model much in choosing an evasive action.

- The sensing region is horizontally wide, but vertically, it is a very narrow band in front of ownship. Therefore, nearby aircraft can fly undetected most of the time (even when they are dangerously close). This also causes late detection of some ascending or descending intruders that suddenly enter the detection region, leaving very little space and time for an escape maneuver.
- During an escape maneuver, the sensor orientation (and hence the orientation of the detection region) changes as ownship accelerates (pitches) up or down. Most of the vertical maneuvers cause the intruder to move outside of the sensing region.

In terms of model implementation, there is very little work to do: We base our design on the POMDP model of Section VI.B (using the same state and action spaces described in Section VI.B.3) with some minor adjustments that we describe below, we use the previously introduced special observation, *noObs*, whenever there is no detection (for example, when a state falls outside sensing region), and we just employ the POMDP solver to design effective strategies for dealing with the limitations of sensing. This also validates one of our central premises for this work; POMDP models allow us to easily and quickly design collision avoidance strategies for different sensor configurations.

VII.A. POMDP Collision Avoidance System with Radar Sensor

For the radar sensor, we use the same observation model as the TCAS sensor with the following two modifications:

- An overly conservative *margin* can be defined in terms of standard error parameters from Table 3:

$$\begin{aligned} \textit{margin} \;\; = \;\; & 3 \times \text{Range error standard deviation } + \\ & \text{Longest distance to bin edges} \times \tan(3 \times \text{Elevation error standard deviation}) \end{aligned}$$

- The pitch angle of ownship (and hence the orientation of the sensing region) can be computed using ownship's vertical and horizontal velocity values, but ownship horizontal velocity is currently not part of the state space. In our observation model implementation, we compute some very loose upper and lower bounds for pitch angle using the maximum and minimum velocities of our aircraft model from Table 2, and use them to figure out which X, Y boxes fall outside sensing region. We believe that performance could further be improved with a better POMDP model that could accurately predict the field-of-view of the sensor. However, addressing this issue requires an extension to the state space and increases the POMDP size considerably.

VII.B. POMDP Collision Avoidance System with EO/IR Sensor

The EO/IR sensor reports the elevation angle of the intruder aircraft, therefore the projection of the intruder on RCS can be constrained to lie on a ray (with noise) rather than a point.

Detections from the EO/IR sensor are nominal angles that can be thought of as the centers of angular bins, which are not necessarily uniform. For each state s in which the intruder is located in the modeled X, Y space, we can compute a *nominal* elevation angle $d^*(s)$ to the intruder. We assume that the probability of observing a detection angle d when the actual angle is d^* is proportional to a Gaussian density with mean at d^*; so,

$$\Pr(o_d = d \mid s) = (1 - p_{fp})\frac{1}{z}e^{(d-d^*(s))^2} + p_{fp}|O_d|^{-1} \;\; ,$$

where

$$z = \sum_s e^{(d-d^*(s))^2}$$

is the normalization constant.

We also use the same pitch angle approximation of the radar sensor described in Section VII.A to assign *noObs* to X, Y boxes that fall outside the sensing region.

VII.C. Results

Table 11 summarizes results for baseline and POMDP collision avoidance systems using radar and EO/IR sensors. The vertical velocity penalty was set to −0.1 for the POMDP models.

Table 11. Risk ratios for baseline and POMDP collision avoidance systems (radar and EO/IR sensors).

	Ratio	Velocity	Acceleration
Basic CAS (radar sensor)	0.050830	19.232450	2.409710
Basic CAS (EO/IR sensor)	0.047240	19.450350	2.308330
POMDP CAS (radar sensor)	0.063370	23.628310	1.261540
POMDP CAS (EO/IR sensor)	0.035100	28.610760	1.476910

As expected, radar and EO/IR sensors have higher risk of collision than TCAS and perfect sensors since their performance is inherently limited by their field-of-view constraints.

There are also two important observations here that we would like to emphasize:

- On one side we have the radar sensor that provides an additional *range* reading that allows (horizontal) localization of intruder aircraft in RCS, and on the other side we have the EO/IR sensor with a smaller the error in *elevation* estimate which allows better vertical localization. Even though a simple comparison is not possible, by looking at the risk ratios we can conclude that accurate vertical localization is more important than accurate horizontal localization for collision avoidance systems that perform evasive maneuvers in the vertical dimension.
- A POMDP solver can in fact generate non-trivial (if not superior) collision avoidance strategies that can compete with hand-crafted ones. The EO/IR sensor, with its limited field-of-view and lack of horizontal localization ability, provides us a good example where the POMDP strategy scores a lower risk ratio than the Basic collision avoidance system using the same sensor. As an example of a non-trivial behavior, we observed that the POMDP strategy for the EO/IR sensor commands ownship to pitch up and down successively especially at the beginning of encounters, which would help to actively search for intruders that might be outside the sensing region and/or to better localize ones that are inside the sensing region. This is a sacrifice in terms of more maneuvering, but it results in low risk ratios that is in accordance with the reward model used. Even though a *policy* generated by a solver might not be easy to verify and validate, it can at least inspire hand-crafted techniques and/or serve as a baseline, which is another central premise of our work.

VIII. Discussion

In this section, we briefly discuss the limitations of our collision avoidance models and suggest ways to improve them.

VIII.A. Discretization

Our state space representation captures most of the features that are necessary in selecting an action to avoid collisions, but there is a loss of information when we go from two 13-dimensional aircraft state vectors to a 5-dimensional state space. One way to improve performance is to augment the state space with more features from the underlying true state space. Another way to improve performance is to use a finer grained discretization, which involves adding more bins along each dimension. But we should also note that both of those approaches cause huge growth in the size of the state space and the time it takes to compute policies.

VIII.B. Parameter Values

Our models contain many parameters (most of them are externally configurable and some of them are internal to implementation) that have not been tuned to the encounter model. Many of the parameter values were chosen by experimentation. We believe that performance can be significantly improved by better matching the internal model used for decision making to the encounter model used for evaluation.

VIII.C. Missing State Information

There are certain features that may improve performance that are currently not part of our state space. One such feature is ownship *roll angle*. With limited field-of-view sensors, sometimes whether the intruder falls into the active angular range of the sensor or not depends on how much ownship is banking. In the current formulation, there is no way to estimate the current roll angle from a given state, therefore we cannot project the active angular range of sensors onto the projection plane to determine intruder detectability. We currently assume a fixed (0 degree) roll angle, and add some precautionary margins, but this affects the performance in one of two ways:

- If our roll angle is actually 0, we would be assigning positive probabilities to some undetectable bins that are inside the margins.
- If our roll angle is larger than the margins and the intruder is detectable as a result of this geometric configuration, we would be assigning zero probability for a case that is actually possible. When we have one of those cases during policy execution, we end up with a *belief-state crash* (a belief-state update resulting in an invalid belief-state with 0's assigned to all states).

VIII.D. Observation Models

Error models for most of the sensor measurements are Gaussian. In our implementations, we used a method to coarsely discretize a Gaussian distribution as shown in Figure 8 and applied it to 2-dimensional observation bins. We believe that a better Gaussian discretization scheme or an analytical solution would further improve results, as better observation models help localize the intruder aircraft in RCS with more precision, and that results in better action selection.

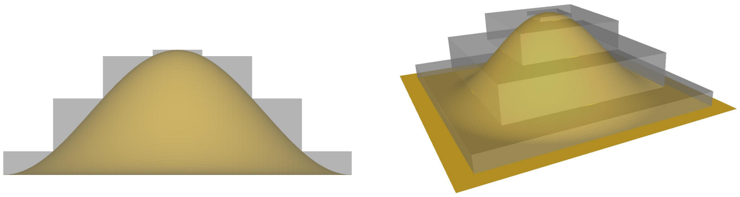


Figure 8. Gaussian distribution approximated by four flat distributions stacked on top of each other.

VIII.E. Estimation of Vertical Velocity of Intruder Aircraft

Evasive maneuvers are performed only in the vertical dimension, therefore it is important to estimate the vertical velocity of intruder aircraft as accurately as possible. Unfortunately, this requires a much finer discretization of the heights of the 2-D bins in the projection plane, which in turn increases the size of the state space.

VIII.F. Estimation of Closure Rate

In order to keep the state space small and still be able to cover a very large projection plane, we used variable sized boxes (both in vertical and horizontal directions). We observed that putting narrow boxes close to the RCS origin and making the boxes wider as we move away from the origin works well for most of our purposes. However, a wide box also means that we will be getting the same observation repeatedly until the projection of the intruder falls into another box. These kinds of observation patterns affect both vertical velocity and relative horizontal velocity (closure rate) estimations, as successively getting the same observation creates the illusion of a stationary intruder, and suddenly getting a different observation results in velocity estimations that are much higher than they really are. As in the vertical velocity case, a finer discretization is required to alleviate this problem.

IX. Conclusions

This paper has shown that the MDP/POMDP formulation is flexible enough to accommodate a variety of sensor modalities, intruder behavior, aircraft dynamics, and cost functions. Complex policies produced by MDP/POMDP solvers can be implemented in real time. Both state estimation and policy execution are quite efficient for the state spaces considered. Current state-of-the-art solvers, using a simplified representation of the aircraft dynamics, can generate useful collision avoidance behavior. Improvements to the problem formulation may further improve performance. In particular, we have limited our formulation to representing motions in two (relative) dimensions. Moving to full three-dimensional motion in a discretized formulation will take the size of the state space beyond the range of existing solvers. We need to investigate other representations for the state space and new types of solvers.

Acknowledgments

The authors wish to thank Nikhil Khanna for his contributions to the early stages of this work while he was an undergraduate at MIT. Special thanks are due to Dan Griffith, an associate staff member at Lincoln Laboratory, who has been tremendously helpful in integrating the MDP collision avoidance system into the simulation framework.

References

[1] Abdul-Baki, B., Baldwin, J., and Rudel, M.-P., "Independent Validation and Verification of the TCAS II Collision Avoidance Subsystem," *AIAA 18th Annual Digital Avionics Systems Conference*, 1999.

[2] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R., "Planning and Acting in Partially Observable Stochastic Domains," *Artificial Intelligence*, Vol. 101, 1998, pp. 99–134.

[3] Simmons, R. and Koenig, S., "Probabilistic Robot Navigation in Partially Observable Environments," *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995, pp. 1080–1087.

[4] Bandera, C., Vico, F. J., Bravo, J. M., Harmon, M. E., and Iii, L. C. B., "Residual Q-Learning Applied to Visual Attention," *13th International Conference on Machine Learning*, 1996, pp. 20–27.

[5] Darrell, T. and Pentland, A., "Active Gesture Recognition using Partially Observable Markov Decision Processes," *ICPR96*, 1996, pp. 984–988.

[6] Roy, N., Pineau, J., and Thrun, S., "Spoken Dialogue Management Using Probabilistic Reasoning," *Proceedings of the 38^{th} Annual Meeting of the Association for Computational Linguistics (ACL2000)*, Hong Kong, 2000.

[7] Horvitz, E. and Paek, T., "DeepListener: Harnessing Expected Utility to Guide Clarification Dialog in Spoken Language Systems," *Proceedings of the Sixth International Conference on Spoken Language Processing (ICSLP 2000)*, Vol. 1, Beijing, China, October 2000, pp. 226–229.

[8] Bagnell, J. A. and Schneider, J. C., "Autonomous Helicopter Control using Reinforcement Learning Policy Search Methods," *International Conference on Robotics and Automation*, IEEE Press, 2001, pp. 1615–1620.

[9] Ng, A. Y., Kim, H. J., Jordan, M. I., and Sastry, S., "Autonomous Helicopter Flight via Reinforcement Learning," *NIPS*, 2003.

[10] Pineau, J. and Thrun, S., "High-level robot behavior control using POMDPs," *AAAI Workshop notes*, 2002.

[11] Puterman, M. L., *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc., New York, NY, USA, 1994.

[12] Thiébaux, S., Cordier, M.-O., Jehl, O., and Krivine, J.-P., "Supply Restoration in Power Distribution Systems - A Case Study in Integrating Model-Based Diagnosis and Repair Planning," *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, 1996, pp. 525–532.

[13] Hauskrecht, M., *Planning and Control in Stochastic Domains with Imperfect Information*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1997.

[14] Boutilier, C., "A POMDP formulation of preference elicitation problems," *Eighteenth National Conference on Artificial intelligence*, American Association for Artificial Intelligence, Menlo Park, CA, USA, 2002, pp. 239–246.

[15] Cassandra, A., "A Survey of POMDP Applications," 1998, Presented at the AAAI Fall Symposium.

[16] Pineau, J., Gordon, G., and Thrun, S., "Point-based value iteration: An anytime algorithm for POMDPs," *International Joint Conference on Artificial Intelligence (IJCAI)*, August 2003, pp. 1025–1032.

[17] Smith, T. and Simmons, R. G., "Point-Based POMDP Algorithms: Improved Analysis and Implementation," *Proc. Int. Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2005.

[18] Kurniawati, H., Hsu, D., and Lee, W., "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," *Proc. Robotics: Science and Systems*, 2008.

[19] Hsu, D., Lee, W., and Rong, N., "A point-based POMDP planner for target tracking," *Proc. IEEE Int. Conf. on Robotics & Automation*, 2008, pp. 2644–2650.

[20] Hsu, D., Lee, W., and Rong, N., "What makes some POMDP problems easy to approximate?" *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[21] RTCA, Inc., "Minimum Operational Performance Standards for Traffic Alert and Collision Avoidance System II (TCAS II) Airborne Equipment," Tech. Rep. RTCA/DO-185A, SC-147, Washington, DC, December 1997.

[22] RTCA, "Safety Analysis of Proposed Change to TCAS RA Reversal Logic, DO-298," RTCA, Inc., Washington, D.C., Nov. 2005.

[23] Billingsley, T., *Safety Analysis of TCAS on Global Hawk using Airspace Encounter Models*, Master's thesis, Massachusetts Institute of Technology, 2006.

[24] Kochenderfer, M. J., Espindle, L. P., Kuchar, J. K., and Griffith, J. D., "Correlated Encounter Model for Cooperative Aircraft in the National Airspace System," Project Report ATC-344, 2008.

[25] Lebron, J. E., Zeitlin, A. D., Spencer, N. A., Andrews, J. W., and Harman, W. H., "System Safety Study of Minimum TCAS II (Traffic Alert and Collision Avoidance System)," Tech. Rep. MTR-83W241, MITRE, Dec. 1983.

[26] Drumm, A., "Lincoln Laboratory evaluation of TCAS II Logic Version 6.04a," Project Report ATC-240, 1996.

[27] McLaughlin, M. P., "Safety study of the Traffic Alert and Collision Avoidance System (TCAS II)," Tech. Rep. MTR 97W32, MITRE Corporation, June 1997.

[28] Chludzinski, B., "Lincoln Laboratory evaluation of TCAS II logic version 7," Project Report ATC-268, 1999.

[29] Russell, S. J. and Norvig, P., *Artificial Intelligence: A Modern Approach (Second Edition)*, Prentice Hall, 2003.

[30] Kuchar, J. K., *A Unified Methodology for the Evaluation of Hazard Alerting Systems*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, January 1995.

[31] Kuchar, J. K., "Methodology for Alerting-System Performance Evaluation," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 2, 1996, pp. 438–444.

[32] Winder, L. F. and Kuchar, J. K., "Evaluation of Collision Avoidance Maneuvers for Parallel Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 6, 1999, pp. 801–807.

[33] Tsang, S. M., "Method and Apparatus for Performing Three-Dimensional Alpha/Beta Tracking," U.S. Patent No. 6,236,899 B1, Filed August 18, 1998, and issued May 22, 2001.

[34] Gray, R. L., "Method of Kalman Filtering for Estimating the Position and Velocity of a Tracked Object," U.S. Patent No. 5,051,751, Filed February 12, 1991, and issued September 24, 1991.

[35] Littman, M. L., Cassandra, A. R., and Kaelbling, L. P., "Learning Policies for Partially Observable Environments: Scaling Up," *ICML*, 1995, pp. 362–370.

[36] Mazor, E., Averbuch, A., Bar-Shalom, Y., and Dayan, J., "Interacting Multiple Model Methods in Target Tracking: A Survey," *IEEE Transactions on Aerospace Electronic Systems*, Vol. 34, January 1998, pp. 103–123.

[37] Tanizaki, H., *Nonlinear Filters: Estimation and Applications*, Springer-Verlag, second (revised and enlarged) ed., 1996.

[38] International Civil Aviation Organization, "Surveillance, radar and collision avoidance," *International Standards and Recommended Practices*, Vol. IV, annex 10, 4th ed., July 2007.