AIAA Aviation
13-17 June 2016, Washington, D.C.
AIAA Modeling and Simulation Technologies Conference

AIAA 2016-3673

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

Multi-Rotor Aircraft Collision Avoidance using Partially Observable Markov Decision Processes

Eric Mueller* and Mykel J. Kochenderfer†
Stanford University, Stanford, CA 94305

This paper presents an extension to the ACAS X collision avoidance algorithm to multi-rotor aircraft capable of using speed changes to avoid close encounters with neighboring aircraft. The ACAS X family of algorithms currently use either turns or vertical maneuvers to avoid collision. We present a formulation of the algorithm in two dimensions that uses horizontal plane accelerations for resolution maneuvers and propose a set of metrics that directly specify aircraft behavior in terms of separation from other aircraft and minimizing deviation from the desired trajectory. The maneuver strategy is optimized with respect to a partially observable Markov decision process model using dynamic programming. The parameters of the model strongly influence the performance tradeoff between metrics such as alert rate and safety. Finding the parameters that provide the appropriate tradeoff was aided by a Gaussian process-based surrogate model. The algorithm is shown to successfully achieve the competing goals of maintaining a minimum separation standard and closely tracking the mission trajectory in simulations involving thousands of collision encounters. Further, sets of algorithm parameters were generated that provide a tradeoff between the two goals. These parameter sets allow a user of the collision avoidance algorithm to select a desired separation minimum appropriate for their application that also minimizes trajectory deviations.

I. Introduction

A new family of collision avoidance algorithms called Airborne Collision Avoidance System X (ACAS X) is being developed and flight tested for a variety of vehicles, from commercial transport to unmanned aircraft.[1–3] These algorithms have shown improvement over the current collision avoidance algorithm, the Traffic Alert and Collision Avoidance System (TCAS), by improving safety while also reducing the rate at which collision alerts are issued.[1] However, the algorithms were designed for fixed-wing aircraft and therefore only use turns or vertical maneuvers in order to achieve safe separation from neighboring aircraft. Other classes of vehicles, particularly rotorcraft and small unmanned aircraft, will be able to effectively use speed changes in addition to turns and vertical maneuvers for separation. The ACAS X methodology for determining optimal collision avoidance maneuvers for any given relative state between two aircraft has not been extended to maneuvers in all three of these control dimensions or tailored to the dynamics of aircraft capable of hovering.

The contributions of this paper are a formulation of the collision avoidance problem that employs horizontal plane accelerations for resolution maneuvers and the computation of an approximately optimal solution to the problem. The problem is formulated as a partially observable Markov decision process (POMDP), and dynamic programming is used to solve for the approximately optimal state-action combinations. A systematic approach is used to tune the reward parameters of the POMDP using a Gaussian process (GP) surrogate model, an optimization technique that has been successfully applied to POMDPs for large aircraft collision avoidance.[4] The benefit of this optimization is that it allows the algorithm designer to select the overall behavior of the algorithm in terms relevant to aircraft flight (e.g., required separation distance) rather than working with abstract parameters embedded in the POMDP. An approximation method known as QMDP

*Graduate Student, Aeronautics and Astronautics, Stanford University, and Aerospace Engineer, NASA, Eric.Mueller@nasa.gov. AIAA Associate Fellow.
†Assistant Professor, Aeronautics and Astronautics, Stanford University, AIAA Senior Member.

Copyright © 2016 by the American Institute of Aeronautics and Astronautics, Inc. The U.S. Government has a royalty-free license to exercise all rights under the copyright claimed herein for Governmental purposes.

is used to increase the robustness of the collision avoidance policy to state and dynamic uncertainty. The offline optimization of the collision avoidance policy can be accomplished in a few hours by selecting a coarse discretization of each state variable, which both minimizes the number of states and preserves the desired collision avoidance behavior. The QMDP policy is evaluated in simulations with initially stationary intruders, intruders moving at uniform velocity with a variety of relative heading angles, and with intruders whose trajectories were generated from a novel hobbyist unmanned aircraft encounter model. Five levels of state and dynamic uncertainty were added to the encounters, adding to the realism of the evaluation and providing a basis for determining the robustness of the algorithm to real-world effects.

The following section formulates the problem of collision avoidance for small, multi-rotor aircraft as a POMDP, introducing the states, actions, and system dynamics. The QMDP solution approach is given in Section III. Section IV describes the optimization approach used to select reward function parameters for a range of desired separations. In Section V the metrics and intruder characteristics used to evaluate the QMDP algorithm are briefly introduced, further information may be found in companion papers.[5,6] The results of the simulation evaluation are given in Section VI and a summary of the conclusions in Section VII.

II. Collision Avoidance Problem Formulation

This section presents the formulation of the collision avoidance algorithm as a POMDP and motivates the fundamental distinction between use of turns and vertical maneuvers for this application versus speed changes. It also describes the dynamic model used to predict transitions to new states given an action.

A. Partially Observable Markov Decision Processes

The collision avoidance problem may be represented as a *partially observable* Markov decision process (MDP), where an MDP is comprised of a set of states that completely define the system under consideration, a set of actions that may be taken in those states, a transition model that specifies the probability of moving from one state to another given an action, and a reward function that indicates the benefit of taking a particular action in a particular state.[7] A "policy" is a mapping from states to actions, and the optimal policy is one that will maximize the expected sum of future rewards. One of the key characteristics of an MDP is that the probability of transitioning to a particular state given one is in a certain state and takes a certain action is conditionally independent of all previous states and actions. The benefit of posing a sequential decision making problem this way is that an optimal policy may be found by repeated application of the Bellman update:

$$Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s'} T(s' \mid s,a) \max_{a'} Q(s',a'). \tag{1}$$

where $R(s,a)$ is the immediate reward received from taking action a in state s, γ is the factor by which future rewards are discounted, and $T(s' \mid s,a)$ is the probability of transitioning to state s' given action a is taken from state s.

The POMDP formulation is an extension of the MDP in which the values of the states are not known precisely. Instead, noisy observations of the states are available, which may be used to update the belief about the current state of the system. The approach taken in this research to incorporate uncertainty, known as QMDP, does not use an explicit observation model and instead calculates beliefs only during algorithm execution.

B. Longitudinal Maneuvers

Many trajectory planning and collision avoidance algorithms allow speed changes as a degree of freedom, but usually only within a narrow range that is consistent with the aerodynamic constraints (e.g., overspeed and stall thresholds) of the vehicle.[8–10] This paper allows movement in the horizontal plane at speeds ranging from a hover up to some maximum speed. The difference this causes in the selection of a resolution maneuver is subtle but important. For a fixed-wing vehicle, the lateral or vertical maneuvers that it would employ to increase separation from an intruder will tend to preserve the time to closest point of approach (CPA), steadily counting down to CPA until the forward motion of the vehicle has carried it beyond the intruder. When a vehicle can slow down to a hover, the maneuver selection becomes more complicated because the best initial risk-reduction maneuver is to decrease speed, which prolongs the encounter. At some point, the

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

algorithm must decide to take action laterally or vertically rather than continue to postpone actions that would actually resolve the encounter.

The implication of a aircraft's ability to hover is that, for optimal performance, tracking of the planned trajectory should be built directly into the collision avoidance maneuver logic. Decomposing the encounter into a collision resolution phase and an independent return-to-path phase would require not only two different algorithms for the two phases, the algorithms would need to undergo extensive interoperability verification checks. On the other hand, incorporating the two phases into a single optimization problem exponentially increases its size and the time required to find an optimal solution to it. Compounding this exponential increase is the fact that encounters no longer count down reliably to CPA, which means that solution approaches like Gauss-Seidel value iteration, employed by the first implementation of ACAS X, can no longer be used to find the optimal policy in just a single iteration.[7] These special considerations in the use of longitudinal (speed) maneuvers means that simple extensions to algorithms designed to use vertical or horizontal maneuvers are unlikely to be successful.

C. Dynamics

The collision avoidance problem is formulated in the two horizontal dimensions (2D) in order to assist in tuning and visualization of results. The eight states that characterize the 2D problem are defined in Fig. 1 and are described in Table 1: two relative range states (r_x and r_y), two velocities for the ownship (v_{ox} and v_{oy}) and two for the intruder (v_{ix} and v_{iy}), and two states that indicate the absolute displacement from the desired trajectory at that time (d_x and d_y). The desired trajectory is normalized to unit velocity in the x-axis and zero velocity in the y-axis. This simplification reduces the number of states required to specify the desired trajectory and avoids loss of generality because the coordinate frame of the ownship may be rotated and scaled into the POMDP coordinate frame to find the appropriate action. A companion paper contains a detailed explanation of how the dimensionless distance units are computed in real time so they are compatible with the algorithm.[5] The algorithm may be extended to three dimensions by adding relative altitude between the ownship and intruder, absolute altitude of the ownship (so that collisions with the ground may be avoided), and absolute vertical velocities of the ownship and intruder.

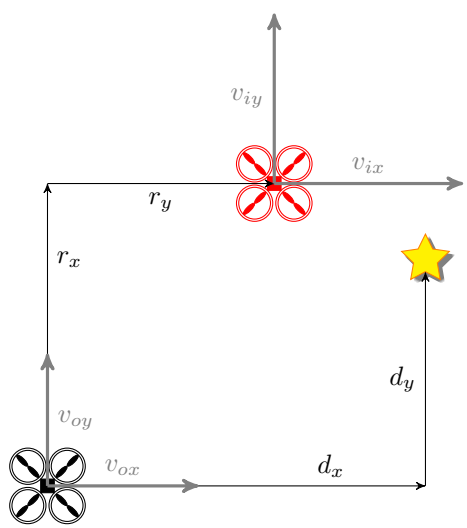


Figure 1: States used to formulate the collision avoidance algorithm

The dynamic equations used to model the aircraft trajectories are relatively simple because the prediction horizon they are used over is very short, with updates done every 0.1 to 1 seconds. Related work has also not found a benefit to using more complex dynamic equations.[7]

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

$$
\begin{aligned}
\dot{r_x} &= v_{ix} - v_{ox} \\
\dot{r_y} &= v_{iy} - v_{oy} \\
\dot{v_{ox}} &= a_x + N_{ox} \\
\dot{v_{oy}} &= a_y + N_{oy} \\
\dot{v_{ix}} &= N_{ix} \\
\dot{v_{iy}} &= N_{iy} \\
\dot{d_x} &= v_{tx} - v_{ox} \\
\dot{d_y} &= v_{ty} - v_{oy}
\end{aligned}
\tag{2}
$$

In Eq. (2), the velocity of the nominal trajectory, rotated and scaled into the ownship-centric reference frame, is represented by v_{tx} and v_{ty}. The accelerations commanded by the ownship in each axis are a_x and a_y. The acceleration noise is represented by N and is subscripted according to whether the uncertainty is applied to the ownship (o), intruder (i), x-axis, or y-axis. The noise variables are assumed to have a Gaussian distribution with zero mean and standard deviation of 0.30 and $0.45\,\mathrm{s}^{-2}$ for the ownship and intruder, respectively. When applying the Bellman update of Eq. (1), the distribution over the next state is discretized using sigma-point sampling as done in prior work.[15] Sigma-point sampling in the context of this model is used to generate nine sets of weighted noise values for the state noise variables, N_{ox}, N_{oy}, N_{ix}, and N_{iy}.

The actions allowed to the ownship are accelerations in each axis, as indicated in Eq. (2). Accelerations were selected rather than velocities because a discrete acceleration command can still result in a continuous set of realizable velocities. Discretized velocity actions would have required orders of magnitude more state-action combinations to be evaluated. Acceleration commands are also consistent with the dynamics and control modes of a rotorcraft because they translate directly into a set of fixed roll or pitch angles. A small set of five possible actions is allowed in each state: no acceleration, positive or negative x-axis acceleration, and positive or negative y-axis acceleration. For each acceleration command only a single value of 1 s^{-2} is allowed.

The aerodynamic performance constraints of a vehicle are imposed by returning a large negative reward during the Bellman updates (see $R(s, a)$ in Eq. (1)) whenever an action in a particular state would violate the constraints. This negative reward is equal to the reward received when a collision occurs because each are equally costly; a failure to maintain appropriate separation is as bad an outcome as a policy that does not respect the real aerodynamic limitations of the vehicle. For a small multi-rotor aircraft this limit can be a simple maximum horizontal rate, while a fixed wing aircraft's performance limits may be maximum and minimum longitudinal velocities along with velocity-dependent lateral accelerations. Either type of constraint may be seamlessly encoded in the optimal policy without being explicitly represented in the dynamics or reward function.

The states that record the distance to the desired trajectory point were added after initial testing showed that the six-state variable formulation that did not include them resulted in the ownship simply flying directly away from the intruder. Although this behavior is understandable and could perhaps have been anticipated, it is not behavior that typically occurs in algorithms that use predicted trajectories with checks for future separation violations and are limited to lateral or vertical maneuvers. In such algorithms, heading or vertical rate changes quickly remove the predicted separation violation, at which point the aircraft stops maneuvering and "coasts" until the range to the intruder begins increasing.

The POMDP formulation does not explicitly predict separation violations and future trajectories so its expected future reward is maximized by accelerating away from the intruder at maximum rate. When the desired trajectory state variables are added the optimal action is now balanced between appropriate separation from the intruder and acceptable deviation from the desired trajectory. The computational cost associated with extra trajectory variables is mitigated by solving for a policy with a normalized trajectory velocity that does not change (in this case unit velocity in x and zero velocity in y). The alternative would be to incorporate trajectory velocity as yet another pair of state variables, greatly increasing the time to find an optimal policy. If the unit velocity assumption is not appropriate for a given trajectory, for example one in which the vehicle is attempting to hover at a specific location, then a new policy can simply be generated for that trajectory.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

D. Reward

The goal of the collision avoidance algorithm is to select actions that maximize the expected sum of future rewards received by the ownship. The reward, specified in the following equation, penalizes actions of any kind, though the overall contribution of this term is quite small compared with the separation and deviation terms. The separation reward term is inversely proportional to the squared range between the intruder and ownship, so a large penalty is placed on close encounters and a steep gradient to that penalty exists as the separation distance approaches zero. Finally, the deviation term provides a penalty that is proportional to the squared distance from the desired trajectory.

$$R(s,a) = \max\left\{R_{\min}, -(k_{ax}|a_x| + k_{ay}|a_y|) - K_s\frac{1}{k_{rx}r_x^2 + k_{ry}r_y^2} - K_T(k_{dx}d_x^2 + k_{dy}d_y^2)\right\} \tag{3}$$

Constants multiplying the x- and y-axis components of each of these states (e.g. k_{rx}, k_{ry}) allow the designer to differentiate between the reward for movement in the two axes, however for the current problem those constants are all unity. The key coefficients are K_s and K_T, which multiply the cost of proximity with the intruder and deviation from the desired trajectory relative to the cost of taking action, respectively. The last parameter that contributes to the reward function is an overall minimum reward, R_{min}, (i.e. maximum cost) that can be received for any particular state-action combination; it is used to ensure that the infinite cost of zero relative range between the aircraft does not numerically overwhelm rewards from larger separations during the solution process. It is also imposed when an action would result in a state outside the aerodynamic performance limits of the vehicle. Appropriate selection of the relative magnitudes of these three reward parameters, K_s, K_T and R_{min}, are critical to obtaining good overall algorithm performance.

III. Collision Avoidance Algorithm Solution

This section describes the approach to solving the collision avoidance problem formulated in the previous section. It describes the approximately optimal QMDP solution of the POMDP using value iteration, and the use of beliefs about the state of the system to select collision avoidance actions.

A. Distance Units

To preserve the generality of the algorithm for aircraft of different scales, distance units in this paper have been normalized so that they can represent any length scale. Variables describing physical sizes, like the separation distances, have no associated units. Only numeric values are assigned to them. Because time units have been preserved, variables describing velocities have s^{-1} units, while accelerations have s^{-2} units. Physical distances are calculated by scaling them according to the ownship's current velocity, which has the effect of increasing the required inter-aircraft separation as velocity increases. Further details are provided in the author's dissertation.[5] If this absence of a specific physical size is confusing, then the reader may assume that for a small hobbyist unmanned aircraft a single distance unit is approximately 15 ft. This length scale implies a typical unmanned aircraft velocity of about 10 kts, and required accelerations are achieved with roll or pitch angles of 24 deg.

B. State Variable Discretization

Determining a solution to the collision avoidance problem formulated in the previous section by applying the Bellman update, Eq. (1), requires a discrete set of states while the problem is naturally continuous. Although there are methods for maintaining the continuous nature of the states,[11,12] the typical solution to this discrepancy is to discretize each state variable (e.g., velocity of the intruder) and solve for the value, $Q(s,a)$, of taking action a in state s. Because all real-world measurements of the system state will not be precisely on one of the discretized states, we interpolate between the neighboring values of Q for each action in turn, selecting the action that would provide the highest expected sum of future rewards.

Table 1 shows both a coarse and fine discretization of the state variables. The values surrounded by brackets are used only in the fine discretization. These coarse values were selected to balance computation with collision avoidance performance. The approximately optimal solution of the POMDP using this discretization may be found on a single-core processor using 2 GB of RAM in under three hours. The solution

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

approach, value iteration (described in Section C), can be parallelized and so this computation time could be reduced significantly using multiple cores.

Table 1: State variable descriptions and coarse and fine discretization points

State variable	State Description	Discretization
r_x, r_y	Intruder range components	$-15, [-7, -3], -1, 0, 1, [3, 7], 15$
v_{ox}, v_{oy}	ownship velocity components	$-5, -3, -1, 0, 1, 3, 5\ s^{-1}$
v_{ix}, v_{iy}	intruder velocity components	$-5, [-3], -1, 0, 1, [3], 5\ s^{-1}$
d_x, d_y	desired trajectory distance	$-10, [-3], -1, 0, 1, [3], 10$

The coarse set of grid discretizations provided adequate separation and trajectory deviation performance with a reasonable computational burden, but improved performance was possible by more finely discretizing each of the states. This finer discretization included the state values shown in Table 1 that are surrounded by brackets. While the coarse set contained a total of 765,625 discrete states, the finely discretized version was more than twelve times larger with 9,529,569. The coarse set was used to select an optimal set of reward parameters, as will be discussed in section IVB, because offline convergence time was reasonable, but the results presented in Section VI were generated with the finer set once the optimal reward parameters had been selected. Convergence of this finer scheme took more than 11 days, even with the coarsely discretized $Q(s, a)$ matrix as a starting point, but performance improved significantly in several important ways that will be discussed in section VIE, and so that policy was implemented for the batch simulations.

C. Value Iteration

The optimal policy can be obtained to arbitrary precision by following a basic dynamic programming approach known as value iteration.[7] The Bellman update (Eq. (1)) is applied to each of the discretized state-action combinations in turn, calculating the state-action value function at every point using the sigma-point sampling approach described previously. When every point has been evaluated the process is repeated, iterating until a stopping criterion is met that indicates the current policy will not change substantially with further iterations. For this problem, the stopping criteria were a combination of the maximum error in $Q(s, a)$ from one iteration to the next and a maximum number of optimal action changes from one iteration to the next.

Considerable improvement in convergence speed can be obtained by initializing the estimate of the value function to a previously evaluated policy with the same state-action discretization and a similar value of the maximum negative reward parameter. This latter parameter was found to have the largest impact on convergence speed of the reward parameters. The discount factor, γ, also had an important effect on convergence speed, with larger values around 0.99 taking hundreds of iterations to converge. This large discount factor was necessary to ensure the optimal policy would return to the desired trajectory after passing the intruder instead of taking a "greedy" strategy of maximizing short-term reward by simply accelerating away from the intruder.

Several examples of the optimal policies that resulted from the value iteration solution to the POMDP are shown in Fig. 2. In those policy plots, the ownship is shown as a quadrotor aircraft at the origin, and the colors surrounding it represent the optimal actions to take based on the location of the intruder. The entire eight-dimensional space cannot be represented here, so a "slice" through that state space showing the effect of intruder range is shown. In the left diagram, both the intruder and ownship velocities are zero and the ownship has no error from the desired trajectory. In the right diagram, the ownship is moving in the positive y-axis direction at 1 s^{-1} with zero trajectory error and the nominal trajectory matches this velocity. The intruder is stationary. For the stationary case on the left, the colors indicate, for example, that when the intruder position lies in the black region centered around a relative range of $(0, -10)$ the optimal action is to move in the positive y-axis direction. The examples shown here are consistent with intuition, but to show that the algorithm is working properly it is impossible to manually examine all relevant "slices" of the state space. Instead, metrics that represent desired algorithm behavior must be formulated and evaluated for a large number of realistic encounter trajectories.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

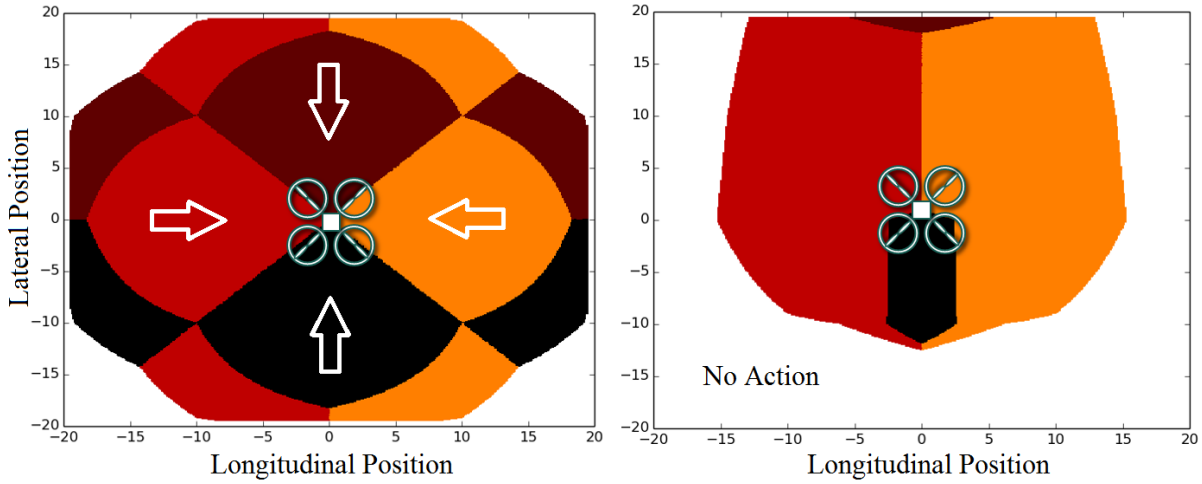


Figure 2: Policies resulting from value iteration solution of POMDP formulation. Arrows indicate actions.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

D. Beliefs

A POMDP is an extension of the MDP formulation in which the current state is not precisely known. Instead, observations are received at each step and those are used to update one's "belief" about the true state of the system. This "belief state" may be a continuous probability distribution over the states, or, as is done in this approach to solving the POMDP, a discrete set of possible states, each with an associated probability. In this QMDP approach, the MDP is solved using the Bellman update defined in Eq. (1) without any consideration of state uncertainty (the dynamic equations do contain acceleration uncertainties). State uncertainty is incorporated only when actions are selected during algorithm execution: a set of potential states is calculated from the observations received at each step using a sigma point sampling technique.[15] These potential states become the beliefs used to select an action:

$$\pi(b) = \max_a \left[\sum_k Q\left(s^{(k)}, a\right) b^{(k)} \right] \tag{4}$$

The states, $s^{(k)}$, are unlikely to lie exactly on one of the discretized grid points given in Table 1. The value of $Q\left(s^{(k)}, a\right)$ is therefore calculated for each action by interpolating between the 2^n nearest neighbor states using rectangular interpolation, or between the $n+1$ neighbor states using simplex interpolation. Although the eight state variables require 256 interpolants under the rectangular method, the much more complex determination of the 9 interpolants in the simplex method means that rectangular interpolation is actually faster. The sum of $Q\left(s^{(k)}, a\right)$, weighted by $b^{(k)}$, is computed for each action, and the action with the highest value is executed. This method of extending MDPs to partial observability is one of the simplest ways to incorporate state uncertainty, and it was employed here because prior work has found little benefit to using more sophisticated approaches.[15]

IV. Collision Avoidance Algorithm Optimization

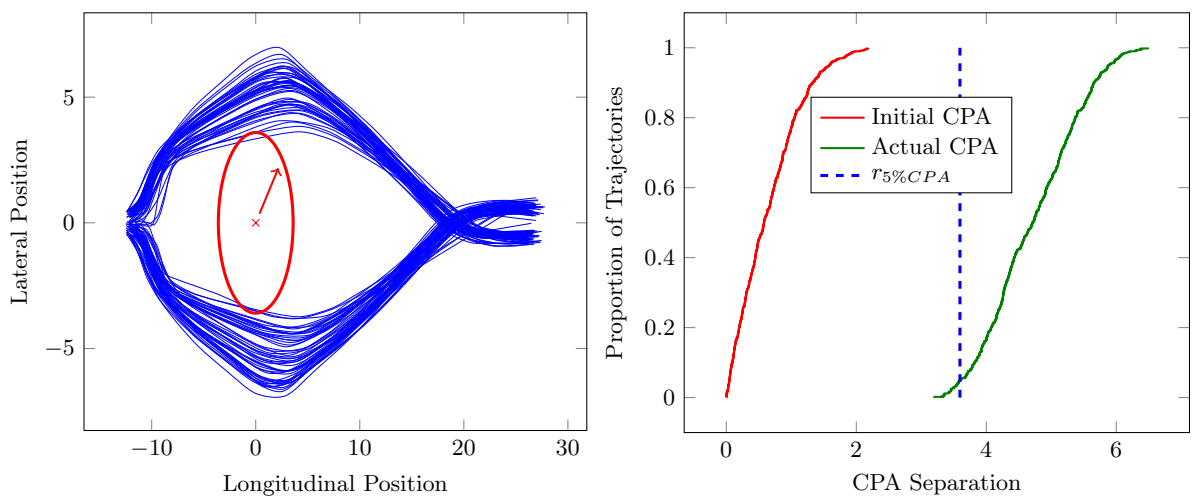
The collision avoidance policy obtained by value iteration is guaranteed to be optimal for the specific reward parameters, discount factor, state-action discretization, and assumed model of the POMDP formulation. However, if the parameters are not selected carefully then the algorithm may not provide the appropriate avoidance behavior. Surrogate modeling has been shown to be effective in improving collision avoidance problems posed as POMDPs.[4] This section describes an approach to automatically tune the reward parameters to balance the separation with an intruder and the deviation from the desired trajectory.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

A. Reward Parameter Tuning Metrics

The primary goal of a collision avoidance algorithm is to remain safely separated from intruder aircraft. The first metric used to evaluate the QMDP algorithm, $r_{5\%CPA}$, therefore, is the separation achieved in most, but not all, encounters. The cumulative distribution of separations for 500 simulated encounters with stationary intruders shown in Fig. 3 illustrates the desired behavior: for a desired separation of 3.5 units, approximately 5% have a CPA within this range and none are closer than 3.0 units. The red line shows the distribution of original predicted minimum separations before any avoidance actions are taken. Specifying the separation metric as the horizontal distance at CPA achieved by 95% of encounters improves the robustness of the algorithm to noise by allowing a small number of minor violations of the separation standard without requiring large trajectory deviations.

The second optimization metric, μ_{dev}, accounts for the desire to avoid course deviation. It is defined simply as the average deviation distance over time, averaged over all the simulated encounters. An example cumulative distribution of mean and maximum trajectory deviations for 500 simulated encounters is shown in Fig. 4. Cumulative distributions on the left side of the chart are desirable, they indicate smaller deviations from the desired trajectory. The maximum single-point deviation distribution for each encounter, which is another relevant algorithm evaluation metric, is also shown in Fig. 4. Each simulation is concluded shortly after the ownship has passed the intruder and had sufficient time to return to the desired trajectory.



(a) $r_{5\%CPA}$ separation distance

(b) Cumulative distributions of CPAs

Figure 3: Separation metric used to optimize QMDP algorithm parameters

B. Multi-Objective Optimization

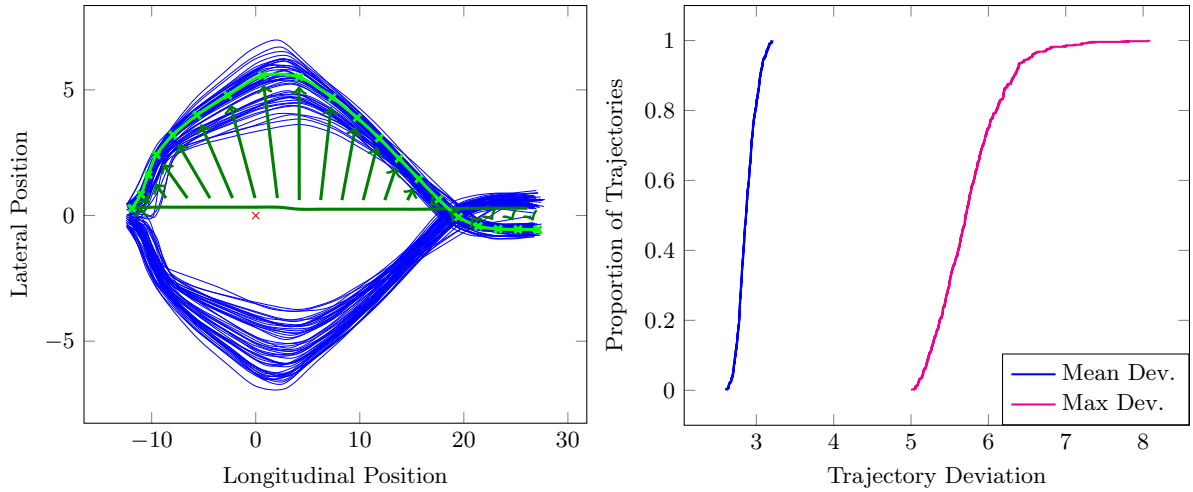
The overall optimization approach for this algorithm consists of two loops as shown in Fig. 5. In the first loop, a given set of reward parameters is selected and the collision avoidance algorithm is posed as a POMDP. This POMDP is solved via value iteration to find the optimal set of state-action combinations across a discretized state space as described in the previous section. The reward parameters that are varied in the inner part of the optimization loop include

$$R_P = (K_T, K_S, R_{min}). \tag{5}$$

The state-action pairs are next used in a batch simulation to determine the trajectories followed by the ownship for a set of stochastic initial conditions with ownship and intruder dynamic uncertainty (see section VB). These trajectories are used to calculate the separation and deviation metrics, and are combined into a single objective function through the use of a relative weighting factor, β, as shown in Eq. (6). The objective is to be minimized over the reward parameters used to generate the QMDP policy,

$$\min_{K_S, K_T, R_{min}} F(R_P) = \left(\beta \times (r_{5\%CPA})^{-1} + (1-\beta) \times \mu_{dev}\right). \tag{6}$$

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673



(a) μ_{dev} deviation metric

(b) Cumulative distributions of deviations

Figure 4: Deviation metric used to optimize QMDP algorithm parameters

In order to combine metrics in which one is to be maximized (separation) and the other minimized (deviation), the inverse of the CPA metric is used instead of its actual value. The relationship between the metrics and objective function evaluations is modeled by a GP, which is then sampled to determine the point at which the objective function is expected to have the largest improvement, $E\left[I\left(F(R_P)\right)\right]$, over that of the current minimum. This set of reward parameters, R_P, is then passed back into the MDP optimization and the process repeats until the convergence criteria are met.

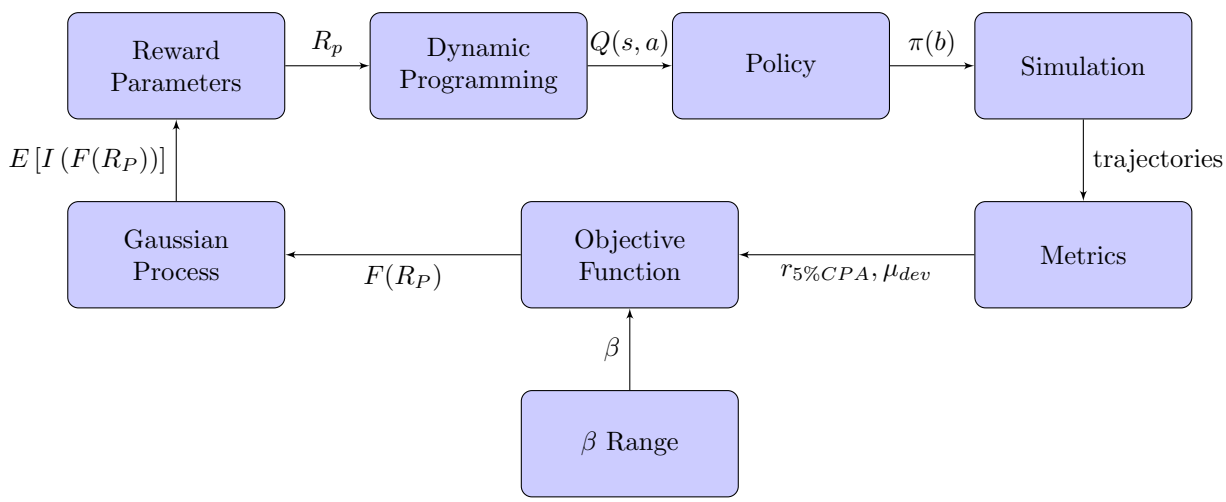


Figure 5: Process for tuning POMDP reward parameters

C. Gaussian Process Surrogate Model

A GP approach[16] to determining the set of parameters, R_P, that provided the minimum objective function (see Eq. (6)) was selected because it required far fewer evaluations of the objective function compared with traditional gradient-based optimization techniques (e.g., quasi-Newton methods[17]) or even direct methods (e.g., Nelder-Meade[18]). This characteristic is important because each optimization and batch simulation loop may take up to three hours, so sparse sampling of the objective function is critical.

The GP surrogate model was created by first selecting a grid of reward parameters, R_P, based on previous experience with the algorithm. Sampling of the grid was done using several Latin hypercubes

to ensure adequate coverage across the parameter search space. These "seeds" condition the probability distribution of the GP, giving it essentially a first guess at the topography of the objective function. A squared exponential kernel function was selected for the covariance, with an initial length scale based on observations of the features of the objective function during the grid evaluation step. That length scale could be varied automatically based on a mean-square error metric.[19] The parameter set with the maximum expected improvement upon the current minimum of the objective function, $E\left[I\left(F(R_P)\right)\right]$, was used to select the next set of reward parameters, R_P, for the subsequent optimization iteration. The uncertainty in the two metrics was estimated through repeated evaluations of the same POMDP-optimized algorithm in batch simulation with stationary intruders (see Section B), and the variance of those measurements was added as a regularization term to the GP covariance function to avoid over-fitting the objective.

D. Optimization Output

An outer loop around the multi-objective optimization step described in Section B varies the relative metric weighting factor, β. This second loop determines the tradeoff between the separation and deviation metrics achieved by the collision avoidance algorithm, and the metrics resulting from every evaluation were sorted to find the non-dominated set of design options. A non-dominated design is one in which no other design has better performance on both the separation and deviation metrics. The set of non-dominated design options are referred to as Pareto optimal, and the set of lines joining each adjacent design is referred to as the Pareto-optimal front.

The existence of design options along the Pareto front allow an algorithm user to select a desired separation based on, for example, their intruder surveillance system performance and ownship's navigation accuracy. They would then select the point most closely approximating that separation standard and use its reward parameters to generate an optimal policy that minimizes deviation while maintaining this level of separation. The inner optimization loop was run for thirteen different values of the tradeoff parameter, β, between 0.01 and 0.99. The GP surrogate model guided the search for the optimal set of reward parameters for a given value of β. A total of 194 different reward parameter sets were evaluated over the course of approximately three weeks, resulting in nine non-dominated, Pareto-optimal designs.

E. Multiple Intruder Collision Avoidance

The QMDP algorithm has been extended to handle an arbitrary number of intruders. The POMDP formulation remains the same, and the optimization step that determines a policy for the multiple-intruder algorithm is identical to that of the single-intruder case. The difference between the algorithms comes in the selection of the optimal action: rather than simply picking the action with the highest weighted average of the belief state over the $Q(s,a)$ matrix as is done in the single intruder case, the action that has the highest *minimum* weighted average over all the intruders is selected. This is expressed as

$$\pi(b) = \max_a \left[\min_i \left[\sum b(s_i)Q(s_i,a)\right]\right] \tag{7}$$

where s_i is the state relative to the ith intruder. This approach to selecting a collision avoidance action has been shown to be superior to selecting the action with the highest mean $Q(s,a)$ value over the intruders.[20] All subsequent results will be presented with a single intruder as development of metrics and evaluation methods for multiple intruder encounters is beyond the scope of this paper.

V. Simulation

This section describes the simulation environment used to evaluate the QMDP algorithm. It introduces first the metrics that provide quantitative measures of performance, the levels of uncertainty that are applied to the system dynamics and intruder trajectories, and finally the types of encounters created to challenge the algorithm.

A. Metrics

The separation and deviation metrics used to optimize the reward parameters are again used to evaluate the algorithm's performance. The separation at CPA is the fundamental trajectory-level *separation* metric,

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

while the median and 5% threshold separations are the aggregate *separation* metrics (see Fig. 3). The mean trajectory deviation over the entire encounter and the absolute maximum deviation at any point in the encounter are the two trajectory-level *deviation* metrics used to evaluate the QMDP algorithm.

No explicit separation requirement is imposed on the algorithm, instead the reward parameters were tuned to typically achieve a given separation (see section IV D). In the results that follow, an algorithm design was selected with reward parameters that deliver a typical separation of at least 4.0. With this typical value, a serious separation violation is defined as a CPA separation under 3.0.

B. Uncertainty

Sources of uncertainty can be divided roughly into three categories based on their origin, magnitude, and how they affect intruder state predictions. The first category is called state uncertainty (also sometimes referred to as parametric uncertainty), and it is related to inaccuracy in measuring the current location or velocity of the aircraft.[21] When projecting the state uncertainty forward in time its magnitude does not grow, so its predominant effect is on actions that depend on the current state rather than ones that depend on predictions. State uncertainty is added to the true values of each state variable during simulations. The QMDP algorithm makes assumptions about the values of these errors, which increases its robustness to the uncertain state measurements. The normally distributed sensor errors used in the optimization and simulation evaluations are shown in Table 2, however it is important to note that any type of distribution may be used to model uncertainty.

Table 2: State and dynamic uncertainties

Uncertainty Parameter	Used in	Mean	Standard Deviation
Ownship acceleration, N_{ox}, N_{oy}	Opt and sim	$0\,\mathrm{s}^{-2}$	$0.30\,\mathrm{s}^{-2}$
Intruder acceleration, N_{ix}, N_{iy}	Opt and sim	$0\,\mathrm{s}^{-2}$	$0.45\,\mathrm{s}^{-2}$
Relative range, r_x, r_y	Simulation	0	0.30
Ownship absolute velocity, v_{ox}, v_{oy}	Simulation	$0\,\mathrm{s}^{-1}$	$0.075\,\mathrm{s}^{-1}$
Intruder absolute velocity, v_{ix}, v_{iy}	Simulation	$0\,\mathrm{s}^{-1}$	$0.15\,\mathrm{s}^{-1}$
Distance to trajectory, d_x, d_y	Simulation	0	0.15

The second category of error is dynamic uncertainty, which affects the accuracy with which the vehicles are able to fly along a particular path. The sources of this uncertainty may be unmodeled wind velocities or an inaccurate navigation system, both of which would cause the error in the predicted positions of aircraft to grow as the prediction horizon increases. The uncertainty in the dynamics of the collision avoidance encounter are expressed as normally distributed acceleration noise on both the intruder and ownship at every time step during the encounter. The values of the uncertainties and the stage of the QMDP algorithm in which they are used are given in Table 2.

The third category of error arises from the lack of knowledge about where the intruder intends to go. Although similar to dynamic uncertainty in that errors in predicted state grow with prediction horizon, this category is particularly difficult to handle because turns, climbs and descents (i.e. accelerations) cause much larger differences between the predicted and actual states than other forms of dynamic uncertainty. This error is introduced through the use of encounter model trajectories.

The uncertainties represented in Table 2 are referred to as "1.0× uncertainty." Five different levels of uncertainty were computed, each a fraction of this full amount of uncertainty, to understand the effect that uncertainty has on the algorithm's performance. When referring to an uncertainty factor, the fraction will be reported and followed by an "×" to indicate that it is a multiplicative factor of the standard deviation in Table 3.

C. Intruder Trajectories

The selection of realistic and appropriate intruder trajectories is critical to the accurate evaluation of a collision avoidance algorithm. Two different classes of intruder trajectories are used in this research: intruders

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

nominally moving with uniform velocity and maneuvering intruders that follow realistic trajectories compared with real-world unmanned aircraft. Only the uniform velocity trajectories are subject to dynamic uncertainty, while the measurement of the state of every aircraft in either class is subject to state uncertainty. All ownship trajectories are also subject to dynamic uncertainty. The initial conditions of each encounter are set for direct collision (i.e., separation at CPA is 0.0), however the effects of dynamic uncertainty divert one or both aircraft from a direct collision. In nearly every encounter at all levels of uncertainty a maneuver is still required to maintain the desired separation. No encounters were designed to trigger "false alerts."

Intruder aircraft moving with uniform velocity are a typical encounter case and one which allows evaluation of overall avoidance performance as a function of a range of relative headings and speeds. The relative heading is the difference between the ownship and intruder heading at CPA, so a value of 180° is a head-on encounter. Each combination of velocity and heading is replayed ten times under different sequences of state and dynamic uncertainty in order to ensure a range of reasonable behavior is obtained for each encounter type. The parameters that were systematically varied for this class of intruders and their values are shown in Table 3. A total of 1320 uniform velocity encounters were simulated for each experimental condition (i.e., level of state and dynamic uncertainty).

Table 3: Encounter parameters for uniform velocity intruders

Encounter Parameter	Minimum Value	Step Size	Maximum Value
Relative heading (deg)	30	30	330
Relative velocity(s^{-1})	0.25	0.25	3.0
Number of uncertainty histories	10		

The intruder trajectories generated by the encounter model are meant to simulate realistic flights by hobbyist unmanned aircraft and represent the most difficult encounter conditions: maneuvering (or, equivalently, accelerating) intruders. In some cases it may be impossible for the ownship to avoid a close encounter when an intruder maneuvers at close range to the ownship with high relative velocity. Details of the characteristics of these intruder trajectories and how they were shown to be realistic are provided in a companion paper.[6] A total of 7000 encounter model intruders were simulated for each experimental condition.

VI. Results

This section presents the results of simulations using several types of initial conditions and levels of surveillance uncertainty. The available design tradeoffs for different sets of reward parameters are shown, each of which delivers a different combination of separation and deviation distances. Example collision avoidance resolution trajectories are discussed next. The sections following this discussion describe the algorithm's performance in terms of separation and deviation for a single set of reward parameters, and the last section compares the performance of the coarsely and finely discretized state variables specified in Table 1.

A. Optimal Design Tradeoffs

One of the goals of formulating the collision avoidance algorithm as a POMDP was to allow the optimization process to automatically deliver different tradeoffs between separation and deviation through the selection of appropriate reward parameters. The points shown in Fig. 6 represent the metrics that result from particular selections of reward parameters during the inner loop search for the Pareto-optimal front. Each objective evaluation made during that inner loop search, including initial and intermediate sub-optimal reward parameter sets, is shown. The points that are non-dominated, meaning no other design point is able to improve on both separation and deviation metrics, are connected by a green line.

A selection of the reward parameters and resulting metrics for a representative set of points on the Pareto front are given in Table 4. The results presented in the previous section were from an algorithm optimized with the bold reward parameters. This figure and table indicate that the algorithm may be automatically

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

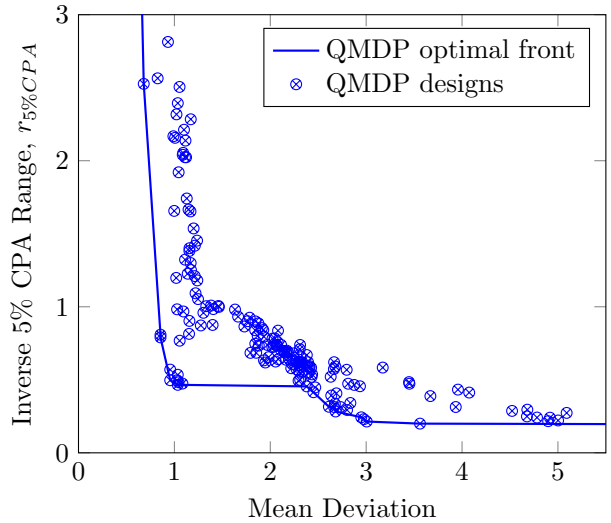


Figure 6: Pareto-optimal front of reward parameter design options

tuned to the high-level goals of the aircraft user and do not require laborious modification and testing of maneuver logic that can be tuned to a desired separation distance.

Table 4: Reward parameters and evaluation metrics for stationary intruder

Separation Coef, K_S	Deviation Coef. K_T	Min Reward, $R_{\min}$	Mean Deviation	95% CPA
148.5	18.2	-5	0.56	0.13
1000.0	26.2	-106	0.68	0.40
979.9	27.6	-206	0.86	1.26
1000.0	28.9	-307	0.96	1.76
986.6	**26.9**	**-508**	**1.04**	**2.15**
225.0	1.0	-10000	2.26	3.83
1.0	0.1	-6245	4.68	4.01
1.0	0.1	-7653	5.00	4.51

B. Resolution Trajectories

An important test of any collision avoidance algorithm is whether the maneuvers it recommends are consistent with the types of maneuvers an aircraft operator would select when faced with the same encounter. To evaluate the degree to which the resolution maneuvers are reasonable, two sets of resolution trajectories are shown in Fig. 7. The left diagram shows uniform velocity intruders and the right shows a hovering ownship. In both diagrams the intruder always begins at the $(0, 0)$ point, and the CPA locations of each encounter are indicated with green diamonds connected by a light green line. The red lines indicate the intruders' trajectories while the blue lines indicate ownship trajectories. Three encounters are shown on each plot for convenience; in each encounter only a single intruder is simulated.

In the first set of three initial conditions on the left side of Fig. 7, the ownship begins at a distance of 15.0 from the predicted CPA and the intruder is moving at unit velocity towards a near collision with the ownship at one of three relative heading angles. The intruder is also subject to random accelerations and so, despite the "uniform velocity" label, does not exactly follow a linear trajectory. The two encounters in which the ownship started on the left side of the diagram unfold in a typical way for a collision avoidance algorithm. The ownship makes a small maneuver between 17 and 33 deg to pass to the left or in front of the intruder, then shortly after CPA turns back towards the desired trajectory and approaches it at a rate roughly proportional to the trajectory error. In the third condition, the intruder is approaching at an acute angle, which is usually difficult to resolve in the horizontal plane when the two aircraft are flying at nearly

the same velocity because a turn will only prolong the encounter. The QMDP algorithm takes advantage of the ownship's ability to slow down and simply waits for the intruder to pass in front, then accelerates back to the desired trajectory. This collision avoidance behavior, which is natural in non-aviation contexts, is not possible with traditional collision avoidance algorithms or when constrained to follow the dynamics of fixed-wing aircraft.

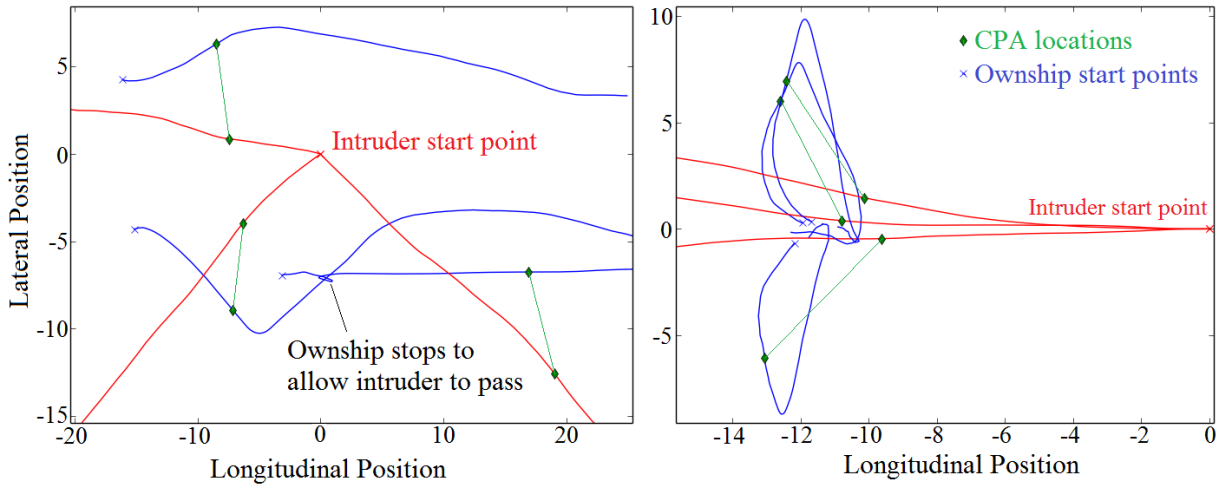


Figure 7: QMDP collision avoidance maneuver examples

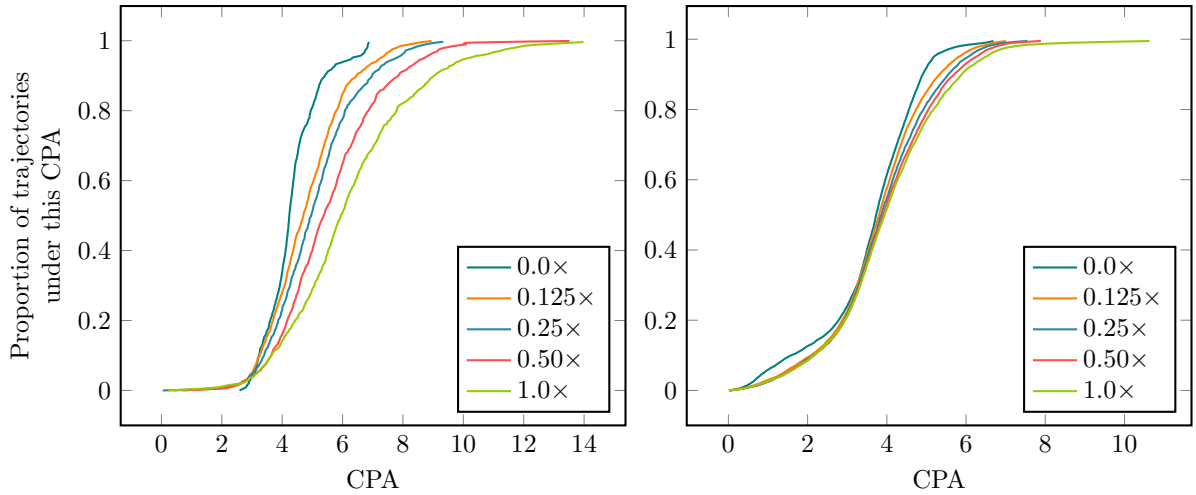
To evaluate the second set of initial conditions, shown on the right side of Fig. 7, a new collision avoidance policy was generated in which the nominal trajectory of the ownship was set to zero velocity (i.e. a hover). The ownship is trying to remain stationary at the initial point and is being approached by an intruder moving along the negative x-axis at unit velocity. The ability for the POMDP formulation and QMDP solution to handle both types of ownship initial conditions with only a small change to the desired trajectory parameters is an important test of its flexibility and robustness. In each encounter the ownship initially moves perpendicular to the intruder's velocity and slightly away from it, which has the effects of immediately increasing the predicted separation at CPA and lengthening the time to CPA. When the intruder reaches 5.0 units longitudinally and 3.0 units laterally, the ownship begins accelerating in the positive longitudinal direction, which hastens the time at which the aircraft begin to diverge. Shortly after CPA, the ownship accelerates back towards its starting point and holds position there. The behavior of the collision avoidance-equipped aircraft in both of these cases confirms the algorithm is working in a reasonable manner and is ready for aggregate evaluation.

C. Separation Metrics

The cumulative distribution of separations for the QMDP algorithm are shown in Fig. 8. Examining the pool of 1320 uniform velocity intruders first, uncertainty has the primary effect of spreading out the proportion of separations at the largest values while leaving the minimum separation distances below 3.0 nearly identical. This spread is largely due to the algorithm increasing separation only when the CPA metric is predicted to be low, which compresses the distributions together at the bottom end of the CPA chart. As dynamic uncertainty increases the intruders have the potential to be driven farther from a direct collision, which spreads out the CPA distance at the higher separation end of the distribution. In general, the rate of serious separation violations (i.e., CPA distance is less than 3.0) is low for all these levels of uncertainty.

The 7,000 encounter model trajectories on the right side of Fig. 8 have a smaller spread of CPA distances than were seen in the uniform velocity encounters. Part of the explanation for this difference is the fact that stochastic dynamic uncertainty is not present for these intruders; their acceleration uncertainties come from the encounter model itself. The second reason state and dynamic uncertainty have less of an effect on minimum separations is that intent uncertainty is dominating the encounter. The unexpected accelerations that comprise such uncertainties govern the CPA separations, and those accelerations are not affected by the level of uncertainty. The failure of the algorithm to prevent the 19% of encounters that have CPA separations under 3.0 is an outcome of the intent uncertainty as well. It is only at the 30% of encounters

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673



(a) Uniform velocity intruders

(b) Encounter model intruders

Figure 8: Cumulative distributions of CPA separations as a function of uncertainty factor

with the highest CPA separations, greater than about 5.0, that the metric's distributions spread out and the uncertainty levels can be distinguished.

D. Trajectory Deviation Metrics

The maximum deviation values for the QMDP algorithm as a function of uncertainty are shown in Fig. 9. The deviations are a weak function of uncertainty for all of the uniform velocity intruders, with the maximum deviation decreasing as uncertainty increases. This effect is largely due to dynamic uncertainty moving the intruder away from a direct collision and making the required amount of deviation lower. In approximately 80% of the simulations with encounter model intruders the maximum deviation is insensitive to uncertainty, but the deviations increase significantly with increasing uncertainty for the last 20%. This effect is due to the belief state computations that improve the robustness of the QMDP algorithm: the aircraft maneuvers to a larger maximum deviation in order to reduce the probability that an unanticipated accelerating intruder will cause a close encounter. When the deviation distributions are viewed along with the separation metrics the conclusion is that the algorithm is relatively consistent in providing appropriate separation distances while ensuring trajectory deviations are reasonable.

E. State Discretization Effects

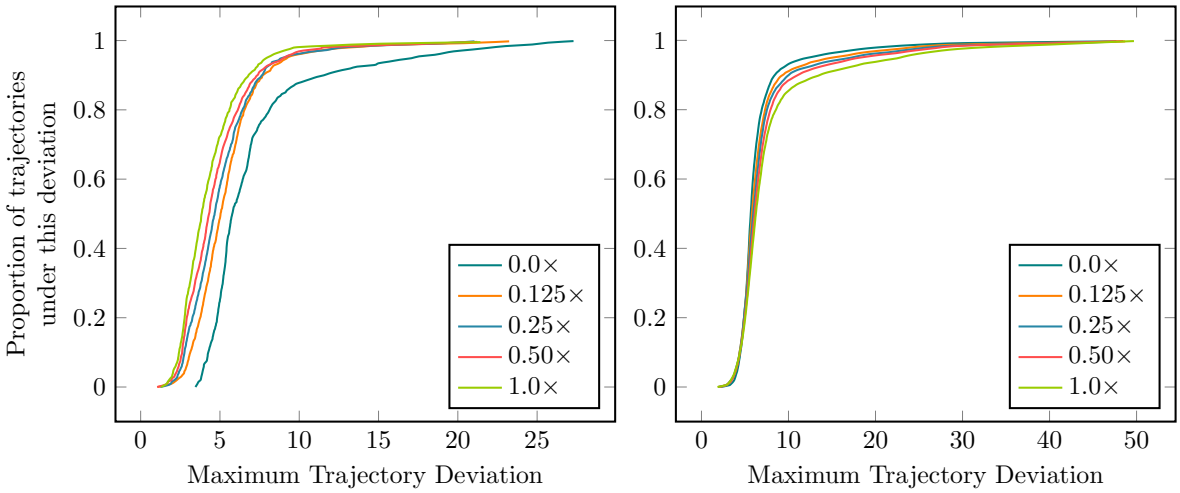
The cumulative distributions in Fig. 10 show the CPA separations and maximum trajectory deviations with the two different discretization schemes from Table 1. The reward parameters used to generate the policies were identical, so the separations and deviations they were targeting were identical. They both were relatively successful at providing separations greater than about 2.0, and the distributions lie nearly on top of each other for these close encounters. However, the coarse discretization does not possess grid points at ranges between 1.0 and 15.0, so it provides more than the required separation for a majority of the encounters. The implication of this excessive separation is that the maximum trajectory deviations are also much larger than required. The selection of an appropriate discretization scheme has a major effect on the performance of the QMDP algorithm, so it is important that care is taken to balance the additional computational costs of a finely discretized scheme against that increased performance.

VII. Conclusions

This paper presented a formulation of the collision avoidance problem for multi-rotor aircraft as a partially observable Markov decision process (POMDP). The approach extends the ACAS X methodology to incorporate speed, in the form of horizontal plane accelerations, as a significant degree of freedom. The

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

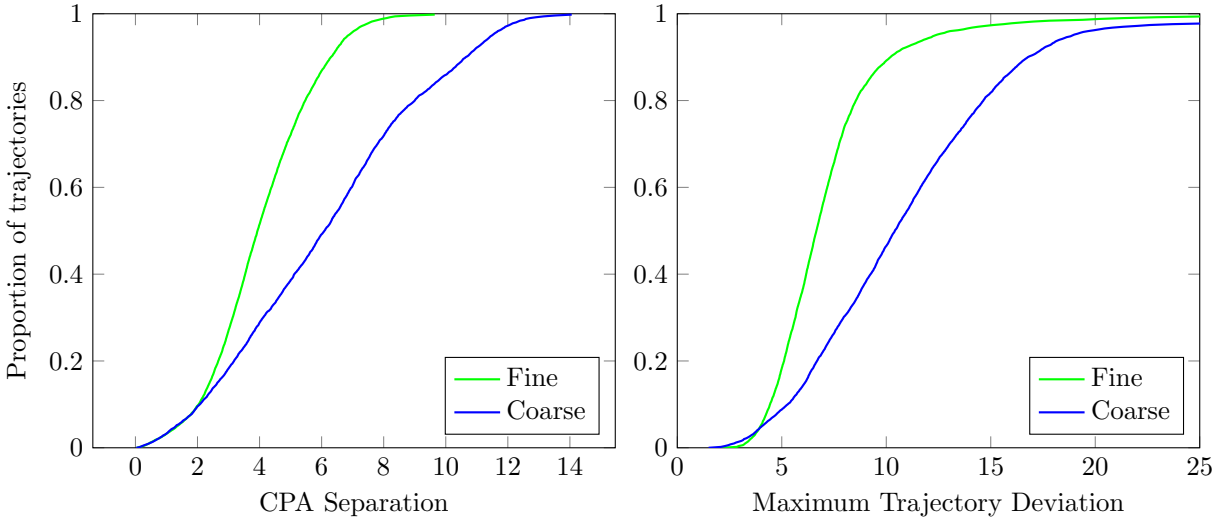
Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673



(a) Uniform velocity intruders

(b) Encounter model intruders

Figure 9: Cumulative distributions of maximum deviations as a function of uncertainty factor



(a) CPA separations

(b) Maximum trajectory deviations

Figure 10: Cumulative distributions of encounter model metrics as a function of state discretization

parameters for a horizontal, two-dimensional version of this algorithm were presented, and a set of metrics proposed that would be directly applicable to defining aircraft performance by the user of such an algorithm. These metrics, which relate to the minimum separation from an intruder and the deviation from the desired trajectory, are used to judge the performance of the algorithm for a given parameter set. An optimization loop was created to automatically select reward parameters that balance separation requirements with deviation. That optimization loop includes the POMDP formulation, an approximately optimal QMDP method for selecting actions, a simulation capability to evaluate the metrics, and a Gaussian process surrogate model to select reward parameters for subsequent iterations. The optimization was run for a range of values of relative importance between the two metrics in the inner loop.

The Gaussian process-based optimization scheme generated a large set of collision avoidance algorithms, each with a different set of reward parameters, that provided different tradeoffs between separation and trajectory deviation. One of those algorithms was used to show the individual trajectories flown by a simulated vehicle in the presence of state and dynamic uncertainty for a variety of initial conditions, including a stationary ownship and a moving intruder and ownship with three different relative headings. The resulting trajectories match the expectations for typical collision avoidance behavior and take advantage of the speed degree of freedom in normally difficult-to-resolve encounters. This use of speed, enabled by the novel formulation of the collision avoidance problem presented in this paper, could provide significant benefit over algorithms that only allow turns or vertical maneuvers. Aggregate statistics measuring the performance of this algorithm over thousands of encounters were also presented, illustrating the robustness of the algorithm to different initial conditions and degrees of uncertainty.

Acknowledgments

The authors would like to thank NASA for supporting Eric Mueller during the completion of his PhD research, support that resulted, in part, in the work presented in this paper. The authors also wish to thank the FAA TCAS Program Manager, Neal Suchy.

References

[1]Kochenderfer, M. J., Chryssanthacopoulos, J. P., and Weibel, R. E., "A new approach for designing safer collision avoidance systems," *Air Traffic Control Quarterly*, Vol. 20, No. 1, 2012, pp. 27–45.

[2]Kochenderfer, M. J., Holland, J. E., and Chryssanthacopoulos, J. P., "Next generation airborne collision avoidance system," *Lincoln Laboratory Journal*, Vol. 19, No. 1, 2012, pp. 17–33.

[3]Asmar, D. M. and Kochenderfer, M. J., "Optimized Airborne Collision Avoidance in Mixed Equipage Environments," Project Report ATC-408, Massachusetts Institute of Technology, Lincoln Laboratory, 2013.

[4]Smith, K. A., Vela, A. E., Kochenderfer, M. J., and Olson, W. A., "Optimizing a Collision-Avoidance System for Closely Spaced Parallel Operations," *Journal of Aerospace Information Systems*, Vol. 12, No. 10, 2015, pp. 618–633.

[5]Mueller, E. R., *Multi-Rotor Aircraft Collision Avoidance using Partially Observable Markov Decision Processes*, Ph.D. thesis, Stanford University, 2016.

[6]Mueller, E. R. and Kochenderfer, M. J., "Simulation Comparison of Collision Avoidance Algorithms for Small Multi-Rotor Aircraft," *AIAA Modeling and Simulation Technologies Conference*, 2016.

[7]Kochenderfer, M. J., *Decision Making Under Uncertainty: Theory and Application*, MIT Press, 2015.

[8]Nieuwenhuisen, M., Schadler, M., and Behnke, S., "Predictive Potential Field-Based Collision Avoidance for Multicopters," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XL-1/W2, September 2013.

[9]Kuwata, Y. and How, J. P., "Three Dimensional Receding Horizon Control for UAVs," *AIAA Guidance, Navigation and Control Conference*, August 2004.

[10]van den Berg, J., Wilkie, D., Guy, S. J., Niethammer, M., and Manocha, D., "LQG-obstacles: Feedback control with collision avoidance for mobile robots with motion and sensing uncertainty," *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 346–353.

[11]Bai, H., Hsu, D., Kochenderfer, M. J., and Lee, W. S., "Unmanned Aircraft Collision Avoidance using Continuous-State POMDPs," *Robotics: Science and Systems*, Los Angeles, California, 2011.

[12]Doya, K., "Reinforcement Learning in Continuous Time and Space," *Neural Computation*, Vol. 12, No. 1, Jan. 2000, pp. 219–245.

[13]Kochenderfer, M. J. and Chryssanthacopoulos, J. P., "A decision-theoretic approach to developing robust collision avoidance logic," *IEEE Intelligent Transporation Systems Conference*, 2010, pp. 1837–1842.

[14]Kochenderfer, M. J. and Chryssanthacopoulos, J. P., "Robust Airborne Collision Avoidance through Dynamic Programming," Project Report ATC-371, Massachusetts Institute of Technology, Lincoln Laboratory, Jan. 2011.

[15]Kochenderfer, M. J., Chryssanthacopoulos, J. P., Kaelbling, L. P., Lozano-Perez, T., and Kuchar, J. K., "Model-Based Optimization of Airborne Collision Avoidance Logic," Project Report ATC-360, Massachusetts Institute of Technology, Lincoln Laboratory, 2010.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673

[16]Engel, Y., Mannor, S., and Meir, R., "Reinforcement learning with Gaussian processes," *International Conference on Machine Learning*, 2005, pp. 201–208.

[17]Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., editors, *Numerical Recipes*, Cambridge University Press, 2007.

[18]Nelder, J. A. and Mead, R., "A simplex method for function minimization," *The Computer Journal*, Vol. 7, 1965, pp. 308–313.

[19]Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.

[20]Ong, H. Y. and Kochenderfer, M. J., "Short-term conflict resolution for unmanned aircraft traffic management," *Digital Avionics Systems Conference (DASC)*, Sept 2015.

[21]Jarrell, J. A., *Employ Sensor Fusion Techniques for Determining Aircraft Attitude and Position Information*, Ph.D. thesis, West Virginia University, 2008.

Downloaded by PURDUE UNIVERSITY on July 8, 2016 | http://arc.aiaa.org | DOI: 10.2514/6.2016-3673