JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 7, August 2010

Real-Time Trajectory Generation for Target Localization Using Micro Air Vehicles

Jeffrey B. Corbets* and Jack W. Langelaan†
Pennsylvania State University, University Park, Pennsylvania 16802

DOI: 10.2514/1.47834

This paper presents an approach to near-optimal target localization for small and micro unmanned aerial vehicles using a family of precomputed parameterized trajectories. These trajectories are precomputed for a set of nominal target locations uniformly distributed over the sensor field of view and stored offline in a nondimensionalized form. Trajectories are parameterized and stored as a sequence of nondimensional waypoints. Upon target detection, a trajectory corresponding to the nearest nominal target location is selected and dimensionalized. An onboard navigation controller follows the dimensionalized trajectory. Thus, trajectory generation occurs in near-constant time, which allows for fast adaptation as the target state estimate is refined. Nondimensionalization of the trajectories with respect to relative vehicle speed, sensor range, and sensor update rate allows the same table to be used for various combinations of sensor package and vehicle or vehicle operating conditions. Results of Monte Carlo simulations show the utility of the proposed approach.

I. Introduction

THIS paper describes a technique for fast, adaptive trajectory planning suitable for deployment on micro air vehicles (MAVs) or autonomous submunitions. Missions envisioned for these vehicles typically include surveillance and target tracking. The research was motivated by a combination of the limited computing power typically available on these vehicles and the limited sensing which can be carried. The sensing limitations complicates the problem of target tracking due to the limited information which can be obtained about the target. Generally, only the availability of a bearing sensor, such as a monocular camera, is assumed. The target tracking problem is further complicated by the nonlinearity of the measurement model.

The combination of limited information (a bearing to the target provides no information about the range to the target) and the nonlinearity of the measurement model leads to a problem of dynamic observability.

This paper a) describes a framework for adaptive trajectory generation based on a nondimensionalized table of optimal trajectories; b) describes the process of generating the trajectory table; c) presents simulation results demonstrating the performance of this table-based approach to trajectory planning.

A. Motivation

Actually being able to complete a surveillance or target tracking mission using an MAV requires advances in several fields, including though not limited to — flight control, sensing systems, obstacle avoidance, state estimation, and trajectory planning. The small size of MAVs complicates the problem because of the limited power, sensing, and computation which can be carried on board. Managing the tradeoffs requires careful system design.

Received 26 October 2009; accepted for publication 10 May 2010.

* Now with Lockheed Martin.
† Assistant Professor, Department of Aerospace Engineering, The Pennsylvania State University, Senior Member AIAA.

This research is concerned with target tracking and state estimation, specifically trajectory design to maximize the information gained about the target. It is assumed that observer vehicle state is known precisely (for example, using global positioning system (GPS)) and that the only sensing available is a monocular camera fixed to the observer vehicle.

A monocular camera provides bearings to targets (or features) in the environment. A single bearing provided by a monocular camera does not provide enough information to localize a target. Fusing bearings from multiple vantage points, however, allows target localization by triangulating measurements. This problem of triangulation can readily be cast as a nonlinear estimation problem and a recursive estimator such as an extended Kalman filter or unscented Kalman filter (UKF) can be implemented.

When an estimator is implemented to solve the bearings-only target localization problem, the lack of range information results in *dynamic observability*: multiple measurements, collected over time, from varying vantage points, allow estimation of target state. Because of sensor noise, the geometry of observer positions and target position greatly affects the accuracy of the target state estimate. This leads to the subject of this research: what is the optimal trajectory which will localize a target with the smallest degree of uncertainty? Furthermore, how can this trajectory be computed in real-time on computing hardware likely to be available on an MAV or autonomous submunition?

Although the technology is general to many trajectory generation problems, the motivating mission is target state estimation for a MAV. A schematic of a sequential target localization task (where the sequence of targets to be visited is determined a priori) is shown in Fig. 1.

A human operator provides a sequence of targets and initial (possibly highly uncertain) estimates of target positions. Each target has an associated risk zone which must be avoided to reduce the likelihood of detection and possible loss of the observer vehicle. The vehicle has a limited field of view sensor, and must plan a sequence of trajectories to minimize the uncertainty in the final state estimate of each target.

B. Fast, Adaptive Trajectory Selection

A schematic of a system for target tracking using an MAV is shown in Fig. 2. It consists of five parts as follows: a) an aircraft, which is acted upon by external disturbances and has as input control commands; b) a flight control system which enables controlled flight and has as input a desired trajectory; c) a trajectory generator; d) an estimator which combines vehicle state information with measurements from a vision system to compute an estimate of target state; e) a camera, which provides bearing measurements to the target.

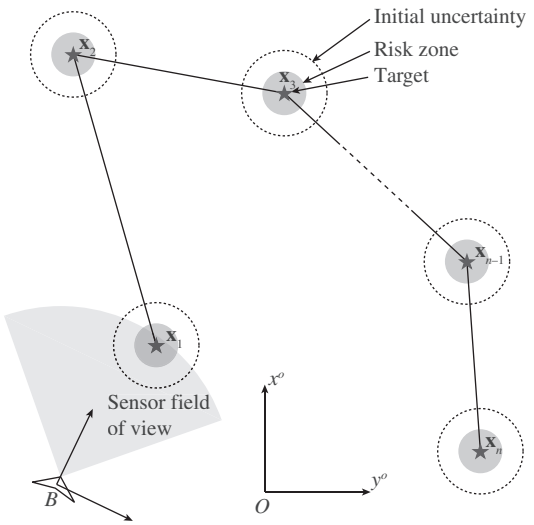


Fig. 1 Schematic of target localization task showing a sequence of targets to be tracked.

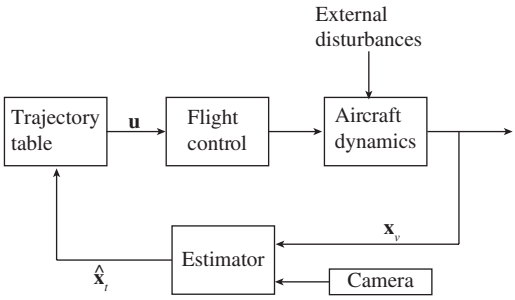


Fig. 2 Block diagram of the system.

The problem of state estimation for autonomous vehicles such as MAVs has been studied in great detail and seems to be well understood for these systems: indeed, several textbooks dealing with state estimation and target tracking have been published [1–3]. Low-dimensional state estimation problems (such as target tracking) can easily be implemented on relatively simple computers; planning trajectories which maximizes information gain is, however, still a difficult problem.

This research proposes a method for fast, adaptive trajectory selection based on a table of trajectories for a set of *nominal* target locations. Upon receipt of an initial (highly uncertain) target location, the trajectory corresponding to the nearest nominal target location is selected and the observer vehicle begins to fly the trajectory. Bearing measurements to the target are obtained and an estimator updates the target state estimate in real-time. As the target state estimate improves, a new trajectory can be selected from the table. To increase the utility of this approach, this paper describes a method for nondimensionalizing the table of trajectories based on the vehicle's speed, sensor range, and sensor update rate. This generalizes the table to different vehicle and sensor combinations. Two methods of trajectory parameterization are described: one based on turn rate commands and one based on a sequence of waypoints defined relative to the estimated target location.

The utility of this approach is demonstrated using Monte Carlo simulations of various target localization scenarios. Target localization performance using the trajectory table is almost equivalent to direct computation of optimal trajectories at vastly reduced online computational burden.

Portions of this work have appeared in [4,5]. This paper expands on previous work with additional numerical results. The remainder of this paper is organized as follows: Sec. II discusses related work; Sec. III describes the problem formulation; Sec. IV describes the trajectory table; Sec. V describes results of Monte Carlo simulations; finally, Sec. VI presents concluding remarks.

II. Related Work

The field of robot motion planning is very broad, with a long history of study. Two notable textbooks are Latombe [6] and LaValle [7].

Because of the dynamic observability caused by the bearings-only sensor, the trajectory followed by the observer vehicle has an enormous effect on the quality of state estimates [8], and optimal trajectory generation for target localization or tracking has become an active area of research [9–11].

Computing the optimal trajectory for a realistic vehicle model and realistic sensor models can become computationally prohibitive, and simplified models are generally used. For example, vehicle dynamics have been modeled as a point mass with velocity and acceleration constraints [11] and sensor models have been linearized [12]. Solution methods including dynamic programming [13] and direct collocation [14] have been used to generate the optimal trajectories. These techniques still require fairly powerful computers and depending on the complexity of the model (e.g., field of view constraints also increase complexity) may not be suitable for real-time operation on the processors likely to be available on an MAV.

An approach that has been used successfully is the use of motion primitives which are connected to form a path [15]. Optimal motion planning using motion primitives has also been addressed [16]. Parameterized maneuver classes [17] allow improved flexibility of the approach. The major difference between motion primitives (also called maneuver

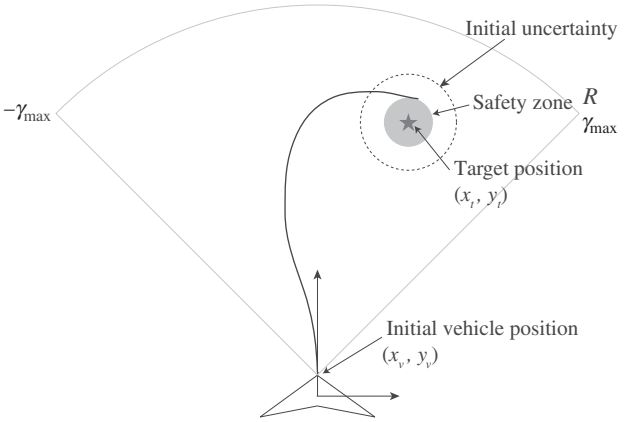


Fig. 3 Schematic of target localization task.

automatons) and the approach described in this paper is that rather than stitching together motion primitives, here a single trajectory is selected from a table of possible trajectories.

III. Problem Formulation

A schematic of a target localization task is shown in Fig. 3. Observer vehicle and target positions are denoted as $\mathbf{x}_v$ and $\mathbf{x}_t$, respectively, in an inertial frame O. A highly uncertain initial target position is assumed to be given and shown in the figure by the dashed line surrounding the target position. The target has a safety zone surrounding it, denoted in the figure by the gray circle surrounding the target position. The safety zone ensures the observer vehicle does not fly too close to the target, putting the observer vehicle at risk from target defense systems or collision. The vision system obtains a bearing γ to the target in the vehicle body frame B. The vision system has a limited field of view, defined by $\pm\gamma_{\max}$. The maximum range of the sensor system is shown in the figure by the top arc at range R. An estimation algorithm uses knowledge of vehicle position and the bearing measurements to compute an estimate of target position $\mathbf{x}_t = [x_t \ y_t]^T$.

For this problem, the initial uncertainty of the target position is assumed to be unbiased in any direction. A safety zone is included in this problem formulation as oftentimes targets for UAVs may be hostile or located in an environment dangerous for a UAV, e.g., close to trees or other structures. As such the safety zone defines the area with a large amount of risk for the UAV. In the case of an autonomous munition, the edge of the safety zone is where a terminal guidance algorithm would take over control of the vehicle.

The problem of target state estimation is clearly critical in the target tracking problem. Solutions to this problem have been well represented in the literature, and the focus here is on planning trajectories to maximize information gained about the target. For completeness we briefly define sensor and system models.

A. Sensor and Vehicle Models

1. Monocular Vision System Model

The vision system obtains a bearing to the target as follows:

$$\gamma = \arctan\left(\frac{y_t - y_v}{x_t - x_v}\right) - \psi_v + \nu, \tag{1}$$

where x_t, and y_t represent the location of the stationary target in the 2D plane; x_v, y_v, and ψ_v represent the vehicle position and heading; and ν is uncorrelated zero-mean Gaussian random noise with covariance Σ_v. Maximum sensor range is R and the sensor field of view is limited to $-\gamma_{\max} \leqslant \gamma \leqslant \gamma_{\max}$. The sensor sample period, which is the inverse of the frame rate for the vision system, is T_f.

2. Observer and Target Motion Model

The MAV is assumed to fly at constant altitude, and it is assumed to include an autopilot module which is able to follow heading and velocity commands. For the purpose of planning a kinematic model this is an adequate representation of vehicle dynamics, provided turn rate and acceleration constraints are accounted for in the trajectory. The velocity of the observer vehicle is assumed to be a constant v given by

$$\dot{x}_v = v \cos \psi_v \tag{2}$$

$$\dot{y}_v = v \sin \psi_v \tag{3}$$

$$\dot{\psi}_v = u \tag{4}$$

where u is the commanded turn rate (constrained by vehicle dynamics).

Since the target is assumed to be stationary, the target motion model is trivial as follows:

$$x_{t,k} = x_{t,k-1} \tag{5}$$

$$y_{t,k} = y_{t,k-1} \tag{6}$$

B. Target State Estimation

The purpose of the estimator is to compute an estimate $\hat{\mathbf{x}}_t$ of the state $\mathbf{x}_t$ and an estimate $\mathbf{P}$ of the covariance of the estimation error. The trajectory planning algorithm will find a path which minimizes the uncertainty $\mathbf{P}$ of the target state estimate.

The bearing model given in Eq. (1) results in a nonlinear estimation problem, and the algorithm for a Sigma Point Kalman Filter (i.e., a UKF) given in van der Merwe and Wan [18] is used to compute the target state estimate $\hat{\mathbf{x}}_t$.

C. The Fisher Information Matrix

The idea of information was developed first in the research of thermodynamics and exists as a way to measure the amount of "information" a known variable contains about a second unknown variable. In this research, the accuracy of the state estimate is measured using the tracking error $\mathbf{e} = \mathbf{x}_t - \hat{\mathbf{x}}_t$. Minimizing the uncertainty $\mathbf{P}$ in this error is equivalent to maximizing the information $\mathbf{Y}$ as the two are inverses of each other: $\mathbf{Y} = \mathbf{P}^{-1}$.

To illustrate the use of Fisher Information in a target tracking application, consider a discrete time system with trivial dynamics and a nonlinear measurement model

$$\mathbf{x}_{k+1} = \mathbf{x}_k \tag{7}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{8}$$

where $\mathbf{v}_k$ is uncorrelated zero-mean Gaussian random noise.

As shown by Ousingsawat and Campbell [11], the Fisher information matrix (FIM) for the estimation problem associated with this system can be computed recursively

$$\mathbf{Y}_k = \mathbf{Y}_{k-1} + \mathbf{H}_k^T \Sigma_v^{-1} \mathbf{H}_k \tag{9}$$

where $\mathbf{H}_k$ is the Jacobian of the measurement model evaluated at time k, i.e.,

$$\mathbf{H}_k = \frac{\delta}{\delta \mathbf{x}} h(\mathbf{x}_k) \tag{10}$$

For the vision model given by Eq. (1) the Jacobian of the sensor model with respect to the estimate of the target is

$$\mathbf{H}_k = \begin{bmatrix} -\dfrac{\sin \gamma_k}{r_k} & \dfrac{\cos \gamma_k}{r_k} \end{bmatrix} \tag{11}$$

where $r_k = \sqrt{(x_t - x_{v,k})^2 + (y_t - y_{v,k})^2}$.

For a single bearing measurement to a single target, $\Sigma_v = \sigma_v^2$. Expanding (9) gives

$$\mathbf{Y}_k = \mathbf{Y}_{k-1} + \frac{1}{r_k^2 \sigma_v^2} \begin{bmatrix} \sin^2 \gamma_k & -\sin \gamma_k \cos \gamma_k \\ -\sin \gamma_k \cos \gamma_k & \cos^2 \gamma_k \end{bmatrix} \tag{12}$$

Writing Eq. (12) as $\mathbf{Y}_k = \mathbf{Y}_{k-1} + \Delta \mathbf{Y}_k$, the information gained about a target over a trajectory can be expressed as

$$\mathbf{Y}_K = \mathbf{Y}_0 + \sum_{k=1}^{K} \Delta \mathbf{Y}_k \tag{13}$$

This presentation of the FIM is applicable when the target is stationary and the FIM itself is representative of the information gained about the position of the target.

The problem of computing a trajectory which maximizes information gained about a target can now be summarized as

$$\text{maximize} \quad c(\mathbf{Y}_K) \tag{14}$$

$$\text{subject to} \quad \text{trajectory constraints} \tag{15}$$

where $\mathbf{Y}_K$ is the FIM at the end of the trajectory and c is a function that maps $\mathbf{Y}_k$ to a scalar: often a determinant is used.

In simplified form, this problem can be solved analytically. However, when complications like safety zones, sensor field of view limits, and more complex vehicle models are considered, the problem must be solved using numerical optimization. With the limited computational power available on small UAVs, this size optimization problem generally cannot be solved in real-time.

IV. A Table of Optimal Trajectories

As stated earlier, real-time solution of the trajectory optimization problem is generally intractable on the computing hardware likely to be available on an MAV. To avoid this problem, a set of trajectories for representative target locations are precomputed and then stored in a lookup table. To make the resulting table of trajectories generally applicable to different sensors and vehicles, the problem is first nondimensionalized using sensor parameters and the observer vehicle speed.

A. NonDimensionalization of the Problem

The problem is nondimensionalized to make the solution (the generated lookup tables of trajectories) more general. By nondimensionalizing with respect to sensor parameters, two vehicles with different speeds or maximum turn rates can use the same lookup table given the same or similar sensor packages. Further, it allows direct comparison of different vehicle and sensor packages. Since the trajectory is parameterized as a sequence of waypoints, external influences like wind can be compensated for by the trajectory-following controller, allowing a single lookup table to be used in almost all conditions for a given sensor package.

1. Kinematics Model

To nondimensionalize the problem, vehicle kinematics are scaled with respect to sensor parameters. Distances are scaled by sensor range R and time is scaled by the sensor frame sample time T_f:

$$\dot{\tilde{x}}_v = \frac{T_f}{R} v \cos \psi_v \tag{16}$$

$$\dot{\tilde{y}}_v = \frac{T_f}{R} v \sin \psi_v \tag{17}$$

$$\dot{\tilde{\psi}}_v = T_f u \tag{18}$$

2. Discrete Time Kinematics Model

A second-order approximation is used to generate a discrete time model for vehicle kinematics with sample time T_s. The integration time is also scaled by the sensor frame sample time (i.e., $\tilde{\Delta}t = T_s/T_f$):

$$\tilde{\mathbf{x}}_{k+1} = \tilde{\mathbf{x}}_k + \frac{T_s}{T_f}\begin{bmatrix} \frac{T_f}{R} v\cos\psi \\ \frac{T_f}{R} v\sin\psi \\ T_f u \end{bmatrix} + \frac{1}{2}\frac{T_s^2}{T_f^2}\begin{bmatrix} -\frac{T_f^2}{R} vu\sin\psi \\ \frac{T_f^2}{R} vu\cos\psi \\ 0 \end{bmatrix} \tag{19}$$

The nondimensionalized discrete time vehicle kinematics are therefore

$$\tilde{\mathbf{x}}_{k+1} = \tilde{\mathbf{x}}_k + T_s\begin{bmatrix} \frac{v}{R}\cos\psi \\ \frac{v}{R}\sin\psi \\ u \end{bmatrix} + \frac{T_s^2}{2}\begin{bmatrix} -\frac{v}{R}u\sin\psi \\ \frac{v}{R}u\cos\psi \\ 0 \end{bmatrix} \tag{20}$$

3. Fisher Information Matrix

The FIM is based on the sensor measurement model, which also must be nondimensionalized. Nondimensionalizing the sensor model with respect to the sensor range gives

$$\tilde{\mathbf{H}}_k = \begin{bmatrix} -\frac{R\sin\gamma_k}{r_k} & \frac{R\cos\gamma_k}{r_k} \end{bmatrix} \tag{21}$$

The nondimensionalized information gained about the target from a single measurement can now be expressed as

$$\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \tilde{\mathbf{H}}_k^T \Sigma_v^{-1} \tilde{\mathbf{H}}_k \tag{22}$$

Hence, the nondimensional form of Eq. (12) is

$$\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \frac{R^2}{r_k^2\sigma_v^2}\begin{bmatrix} \sin^2\gamma_k & -\sin\gamma_k\cos\gamma_k \\ -\sin\gamma_k\cos\gamma_k & \cos^2\gamma_k \end{bmatrix} \tag{23}$$

Again, assuming a stationary target and writing Eq. (23) as $\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \Delta\tilde{\mathbf{Y}}_k$, the nondimensional information gained about a target over a trajectory can be expressed as

$$\tilde{\mathbf{Y}} = \tilde{\mathbf{Y}}_0 + \sum_{k=1}^{K} \Delta\tilde{\mathbf{Y}}_k \tag{24}$$

where $\tilde{\mathbf{Y}}_0 = R^2\mathbf{Y}_0$, the nondimensionalized initial target information. Note that σ_v, the bearing measurement uncertainty, is expressed in radians and is thus inherently nondimensional.

The nondimensional FIM will later be used in the cost function of the optimization problem.

B. Trajectory Parameterization

A common approach is to parameterize a trajectory based on a sequence of inputs. However, this has several drawbacks. First, the number of turn-rate commands for a given trajectory varies with the initial distance from the observer vehicle to the target, i.e., a shorter trajectory has fewer turn-rate commands. This requires either unused space to be reserved in the lookup table, or complicated memory access algorithms to be employed. On the other hand, a longer trajectory may have more than 200 turn-rate commands, which requires significant computational time even on a workstation-class computer. A waypoint parameterization allows for easier optimization and storage as every trajectory relies on a constant number of waypoints (here 10 waypoints are used). The waypoint trajectories are also easier to non-dimensionalize and allow for easier implementation of the optimization constraints. Finally, a trajectory derived from an input-based parameterization is typically followed in open loop, making it susceptible

to disturbances such as wind. Of course the input sequence can be transformed to a trajectory in space that can be followed using a trajectory-following controller, but this requires additional computation.

To generate a target localization table, the sensor field of view was uniformly discretized in the radial and angular directions: i.e., a 10×10 polar grid was defined over the sensor field of view and a nominal target location was defined at each grid point. This discretization is shown in Fig. 4. In this figure, there are N and M divisions in the radial and axial directions, respectively. The optimal trajectory is generated for a potential target located at the centroid of each cell. A schematic of the target localization tables showing nominal target locations and three sample trajectories associated with three nominal locations is shown in Fig. 5.

The trajectories exist in nondimensional space as a sequence of 10 waypoints, $\tilde{\mathbf{X}}_{\text{mn}} = [\tilde{\mathbf{x}}_{\text{mn},1}, \ldots, \tilde{\mathbf{x}}_{\text{mn},N}]$. Each waypoint $\tilde{\mathbf{x}}_{\text{mn},N}$ consists of an angle θ and a nondimensional distance r to the waypoint relative to the nominal target location. Thus, relative to the target location, each waypoint exists at Cartesian coordinates

$$\tilde{x}_n = r_n \cos\theta_n \tag{25}$$

$$\tilde{y}_n = r_n \sin\theta_n \tag{26}$$

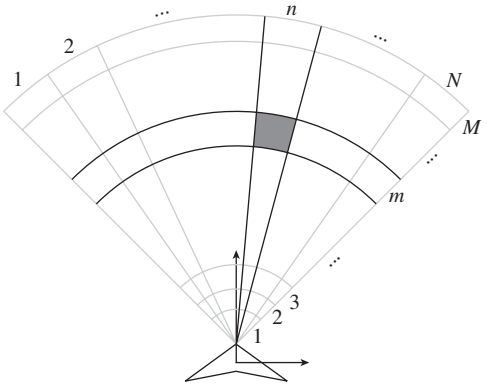


Fig. 4 Discretization of the sensor field of view.

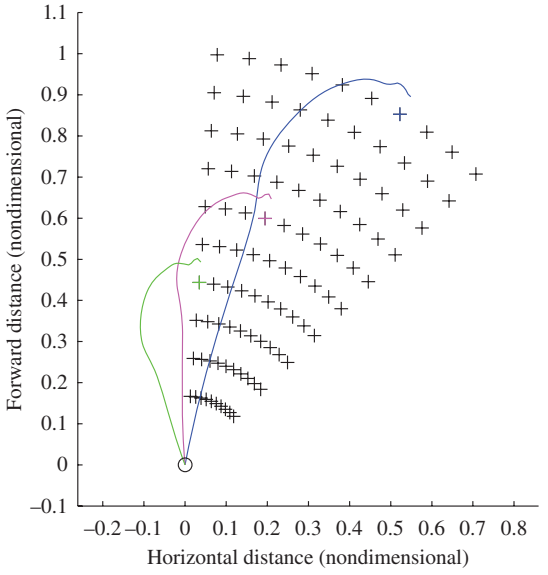


Fig. 5 Sample trajectory table target locations and sample trajectories.

where

$$\theta_n = [0 \ldots \pi] \tag{27}$$

$$r_n = r(\theta) \tag{28}$$

Each waypoint exists at a fixed angle from the target location and the distance of the waypoint from the target is allowed to vary, subject to constraints. To prevent the observer from flying a path that forces the observer into the risk zone after the path is flown, the final two waypoints are fixed to provide a pseudotangency constraint at the end of the path. The final fixed waypoint ensures that the observer will not enter the risk zone at the end of the optimized path. This is shown in Fig. 6.

Each trajectory consists of a sequence of 10 waypoints in nondimensional space. To compute a path in physical space, the waypoints are first dimensionalized by multiplying by the sensor range

$$\mathbf{X}_{\text{mn}} = R\tilde{\mathbf{X}}_{\text{mn}} = [\mathbf{x_{mn,1}}\ \mathbf{x_{mn,2}}, \ldots, \mathbf{x_{mn,10}}] \tag{29}$$

Finally, a spline is used to compute the complete path (Fig. 7).

The trajectory generation problem for target (m, n) can be summarized as

$$\text{minimize} \quad J(\tilde{\mathbf{X}}_{\text{mn}}) \tag{30}$$

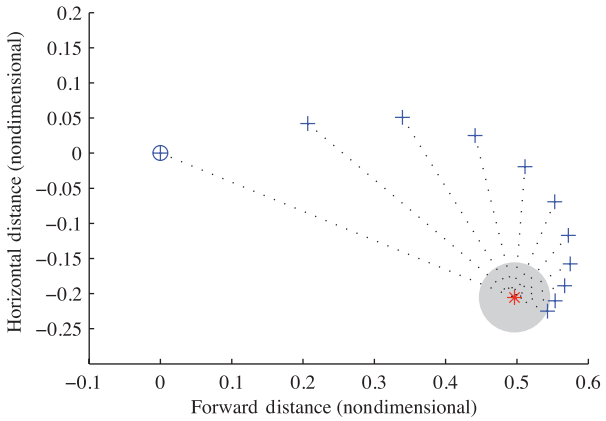


Fig. 6 Waypoints are defined by a distance *r* from the target location at fixed angles.

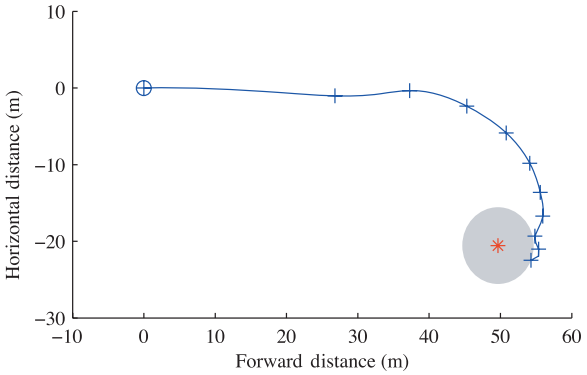


Fig. 7 Ten waypoints are interpolated and dimensionalized to form a complete path.

$$\text{subject to} \quad \tilde{\mathbf{x}}_k = f\left(\tilde{\mathbf{X}}_{\text{mn}}, T_s \frac{v}{R}, k\right) \tag{31}$$

$$\frac{R\dot{\psi}_{\min}}{v} \leqslant \tilde{\kappa}_k \leqslant \frac{R\dot{\psi}_{\max}}{v} \tag{32}$$

where, again, the cost function J is disclosed in Sec. C; the vehicle path is computed using the interpolating function f and $\tilde{\kappa}_k$ is the curvature of the path (nondimensionalized using the sensor range), constrained by vehicle turn-rate limits.

Recall that the final (i.e., 10th) waypoint is on the edge of the risk zone, and thus forms an implicit constraint on the trajectory. In addition, the 9th waypoint is constrained to ensure that the vehicle path is tangent to the risk zone at the 10th waypoint. This ensures that the vehicle does not blunder into the risk zone at the conclusion of the trajectory.

C. Cost Function

The cost function to be minimized is

$$J = (w_{\text{info}} + w_{\text{fov}}) J_{\text{info}}, \tag{33}$$

where w_{info} and w_{fov} are weights associated with information gain and keeping the target in the field of view.

1. *Information Cost*

The information cost is computed using the FIM. Note that the target is assumed to be stationary. A scalar value for information cost is then given by

$$J_{\text{info}} = \log\det \tilde{\mathbf{Y}}^{-1} \tag{34}$$

$$J_{\text{info}} = -\log\det \tilde{\mathbf{Y}} \tag{35}$$

It is important to note that minimizing $\tilde{\mathbf{Y}}^{-1}$ is mathematically equivalent to maximizing $\tilde{\mathbf{Y}}$, or information about the target. Alternatively, minimizing $\tilde{\mathbf{Y}}^{-1}$ is a way of minimizing the uncertainty in the target state estimate.

2. *Field of View Weight*

To keep the target in the field of view, a weight is computed based on the bearing to the target γ_k as follows:

$$w_{\text{fov}} = \left(\frac{\gamma_k}{\gamma_{\max}}\right)^4 \tag{36}$$

Although the field of view is also accounted for in the information cost, this term assists the optimization routine in finding a valid solution. Initial optimizations that did not include this weight resulted in poor, or no, convergence of the optimization.

D. Solution of Waypoint Parameterized Trajectories

To calculate solutions to the vector optimization problem defined by Eqs. (30–32), a minimization routine using a gradient-based line search method was implemented.

To improve the generated trajectories, a pseudotangency constraint is imposed by fixing the distance of the last two waypoints to the distance specified by a logarithmic spiral fitting the fixed final waypoint as well as the initial location of the observer vehicle. The distance of a waypoint from the target vehicle defined by the logarithmic spiral is given by

$$r_n = r_{\text{safe}} \exp(b\theta_n) \tag{37}$$

where

$$b = \frac{(\log r_k - \log r_{\text{safe}})}{\pi} \tag{38}$$

Because of the nature of the line search method, a feasible initial guess of a starting path is necessary to find a solution. To quickly generate valid paths, the optimizer is initialized with a path defined by a logarithmic spiral from

the initial observer position to the target position. The path which results from the optimization is a local optimum near the initial guess.

E. Table Implementation

After solving the optimization problem for each of the nominal target locations, a set of trajectories is created. These trajectories are then stored in a table to be used on a variety of vehicles. For ease of implementation, the storage of the trajectories for this research has been in a two-dimensional table. Each trajectory is indexed by both a nondimensional range and the bearing to each nominal target location. A trajectory can then be selected by matching as closely as possible the nondimensional range and bearing to the actual target to those available in the lookup table. This trajectory (i.e., sequence of nondimensionalized waypoints) is first dimensionalized into physical space and then the onboard waypoint-following controller follows the resulting path.

Only half of the sensor field of view is covered by the trajectory tables shown in Fig. 5. Early results from the optimization routines showed near-perfect reflection symmetry about the longitudinal axis of the vehicle. Thus, the size of the lookup table can be made 50% smaller by exploiting this reflection symmetry. Targets in the left half-plane of the observer vehicle are localized with a reflected trajectory from the lookup table.

F. Observer Vehicle and Sensor Variations

For the waypoint-based implementation, a trajectory-following controller allows vehicles of different speeds to use the single trajectory stored in the lookup table to best localize a target. Since the table was generated by nondimensionalizing the target localization problem using sensor range R, sensor update period T_f, and vehicle speed V, intuition suggests that trajectories stored in the lookup table are optimal for any vehicle and sensor package that has the same "observation number" as the generated trajectory table. This observation number relates the sensor range R and the sample time T_f to the observer vehicle speed v as follows:

$$N_{\text{obs}} = \frac{(R/T_f)}{v} \tag{39}$$

Intuitively, N_{obs} is a measure of the number of measurements of the target that can be obtained before the target is reached. Intuition also suggests, and simulations will later show, that two vehicles which share the same observation number can use the same lookup table for selecting optimal target localization trajectories. Simulations will also show that good, though suboptimal, trajectories can still be used for vehicles with differing observation numbers.

G. Target Localization

Target localization using the lookup table follows three steps:

1) An initial target location is passed to the table. If it is within the field of view, the trajectory associated with the closed cell centroid is selected. If the initial target location is outside the field of view, the vehicle is commanded to turn toward the initial given position of the target and fly until the target is seen.
2) The trajectory $\mathbf{u}_{\text{mn}}$ for the turn-rate-based implementation, or $\mathbf{X}_{\text{mn}}$ for the waypoint-based implementation, is followed in open loop over a control horizon T_c. A target state estimate is computed using a Sigma Point Kalman Filter [18,19]. The control horizon is dependent on the initial distance between the vehicle and the target at the time of the first camera measurement. When the control horizon is reached, a new trajectory is selected based on the current estimate of the target position.
3) The task is complete when the vehicle reaches the risk zone or the vehicle passes the target. This is defined to happen when the observer vehicle reaches the last waypoint or turn-rate command defining the trajectory. The next target in the sequence is selected, and the process repeats for all given targets.

In most target localization algorithms (e.g., [11,12]) the FIM is computed using the current estimate of target position. Thus, the computed FIM is an estimate of the upper bound on the information that can be gained about the target. Here, optimal paths are generated for a set of known nominal target locations, and the nominal target location is used to compute the information cost. The information gain associated with a trajectory pulled from the table is thus an approximation of the information which can be gained about an actual target, and the trajectory pulled from the table is an approximation of the optimal trajectory which would be computed, given the exact knowledge of target position.

V. Simulation Results

Simulation results demonstrate the utility of the proposed approach. First, a simple target localization scenario is used to demonstrate the use of the table. Second, the effect of vehicle and sensor variations on target localization is examined by varying N_{obs}. Third, the effect of adaptation (i.e., selecting new trajectories as the target state estimate is refined) is demonstrated. Finally, a Monte Carlo simulation is used to compare the cost and performance of table-derived trajectories with those computed using direct optimization.

A. Using the Table

A sequence of images showing localization of a single target is given in Fig. 8. In the figure, true target location is shown with *, estimated target location with $+$ and associated error ellipse. The green line shows the current path selected from the trajectory table; the blue line shows the path flown. The results show the target is initially within the field of view of the sensor, and so a trajectory can be selected immediately. The vehicle follows the trajectory while estimating target state for a control horizon $T_c = 2$ s and the target localization task is concluded as the vehicle passes the target. Sensor noise is assumed to be Gaussian with $\sigma_\nu = 0.0175$ rad (i.e., 1°). Using this parameterization

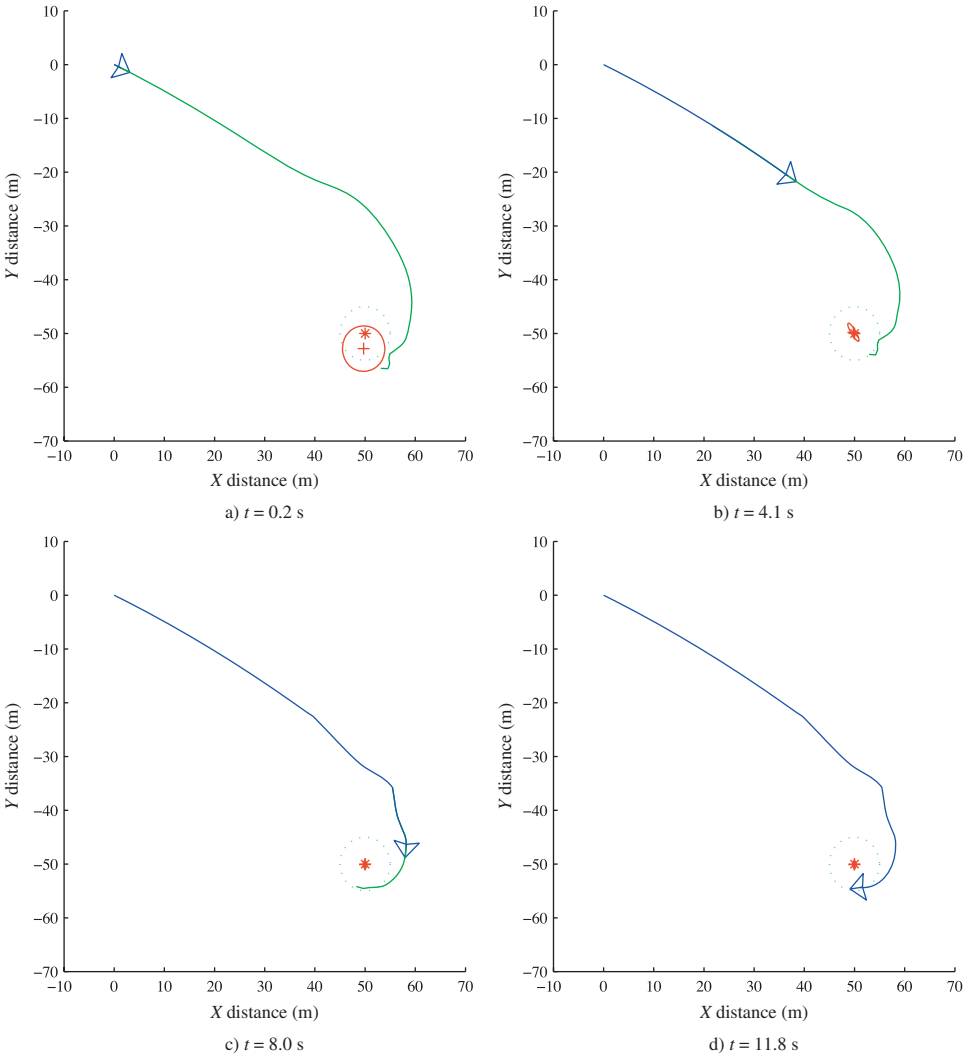


Fig. 8 Snapshots of a single target localization run using the waypoint-parameterized trajectory table.

and allowing for adaptation, the target was localized to an accuracy of 0.075 m using 0.13 s of CPU time. Direct optimization resulted in a localization accuracy of 0.064 m using 2.2 s of CPU time. CPU times for this comparison were measured on a desktop workstation with an Intel 2.4 GHz processor.

B. Observer Vehicle and Sensor Variations

Figure 9 shows that the trajectory created by dimensionalizing a set of 10 waypoints stored in the lookup table and then interpolating generates essentially the same trajectory as a direct optimization in dimensional space. For practical purposes, the trajectories are the same because the small variation in paths near the target occur when the target is outside the field of view of the observer vehicle's sensor package and, thus, is not providing any information gain.

Because the trajectory is stored as a series of waypoints, a trajectory-following controller allows vehicles of different speeds to use the single trajectory stored in the lookup table to best localize a target. Figure 10 shows that different observer vehicle and sensor combinations can still fly near-optimal paths using the same table generated for a single characteristic number N_{obs}. Flying at a slower airspeed and with a short-range sensor, the vehicle in Fig. 10a has the same N_{obs} as the vehicle in Fig. 10b that is flying at a faster airspeed but also has a longer-range sensor. Both vehicles can directly use the lookup table generated using the waypoint parameterization for the specific N_{obs}.

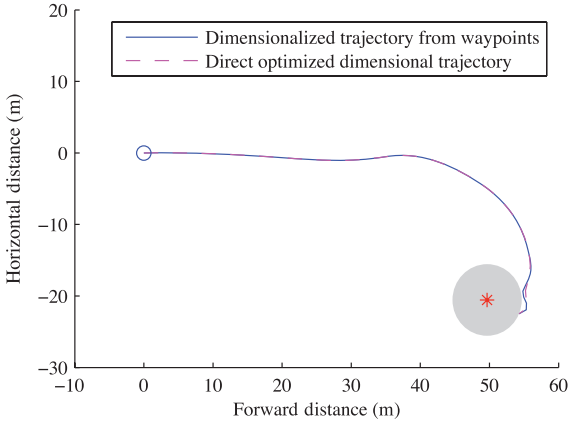
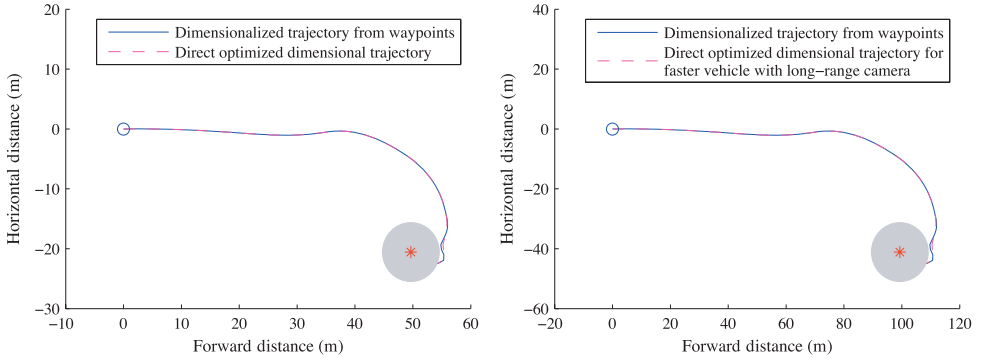


Fig. 9 Direct optimization in dimensional space and use of waypoints from a lookup table yield essentially the same path.



a) $v = 9\,\text{m/s}, R = 100\,\text{m}$ b) $v = 18\,\text{m/s}, R = 200\,\text{m}$

Fig. 10 Different observer vehicle and sensor combinations yield observers with the same observation number and can use the generated table directly.

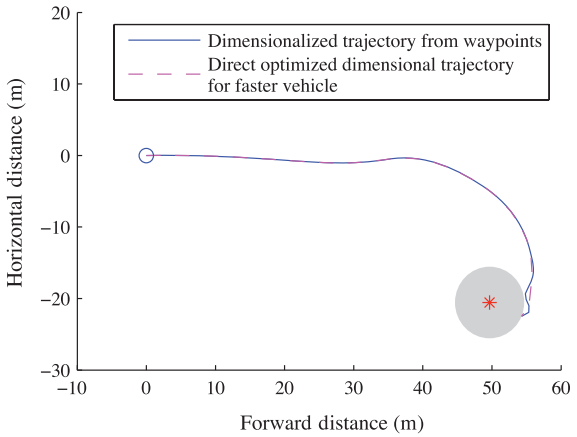


Fig. 11 Comparison of a directly optimized path vs the trajectory table for different observation numbers.

Simulations have further shown that observer setups having different N_{obs} from that of the table (e.g., due to a difference in observer vehicle speed) can still perform accurate target localization using the trajectory from the lookup table. Because the trajectories are stored as nondimensional waypoints, an observer vehicle setup with a sensor of any range can dimensionalize the trajectories to a valid real space path. Figure 11 shows a path that was optimized within dimensional space for a vehicle and sensor package with N_{obs} twice as large as the N_{obs} of the trajectory table compared with the path from the trajectory table. Simulation results have shown that the difference in cost is less than 0.5% when a path (i.e., sequence of waypoints) obtained from the trajectory table is flown.

C. Monte Carlo Simulation: Table vs Direct Optimization

A series of simulations was conducted to evaluate the performance of the trajectory table vs a direct optimization. For the simulations, a target was placed at a random location within the field of view of the sensor. The trajectory corresponding to the nearest nominal target location was chosen and the observer aircraft flew and took measurements along the entire trajectory. In this study, no adaptation of the trajectory occurred as the target state estimate was refined. This was then compared to a trajectory optimized for the actual (random) target location. The ratio of optimization cost from the lookup table trajectory to the optimization cost from a direct optimized trajectory vs target distance is shown in Fig. 12. On average, a trajectory from the lookup table results in 81.0% the optimization cost as the direct (true) optimized trajectory for a random target placed in the sensor field of view. It is important to note that the optimization cost J is typically a negative number. A larger magnitude negative number, as provided by direct optimization, in the optimization cost is thus preferable to a smaller magnitude negative number that is calculated from the table trajectory. The 81% average "return" on optimization cost provided by the table trajectories is thus suboptimal.

When comparing information gain alone, which is directly representative of the target localization performance of the trajectory, the table lookup method performs extremely well. For the 500 simulations, the lookup table method provides, on average, 90.0% the information gain as the true optimized path. This is because the field of view weight in the optimized cost function, although necessary for the optimizer to find a solution, does not affect the information gain in real life. As can be seen in Fig. 13, in some cases, the table lookup method performs significantly (greater than 20%) better than the directly optimized path. In these cases, small differences in the trajectory result in the observer vehicle obtaining a small increase in the number of measurement locations, where the target is within the field of view of the sensor. It is important to note in these results that the difference in performance is due mostly to difference in the actual target location and the nominal target location associated with the initial waypoint trajectory. Additionally, adaptation was not considered in these simulations, which likely would significantly improve the localization performance when using the waypoint-based lookup table trajectories.

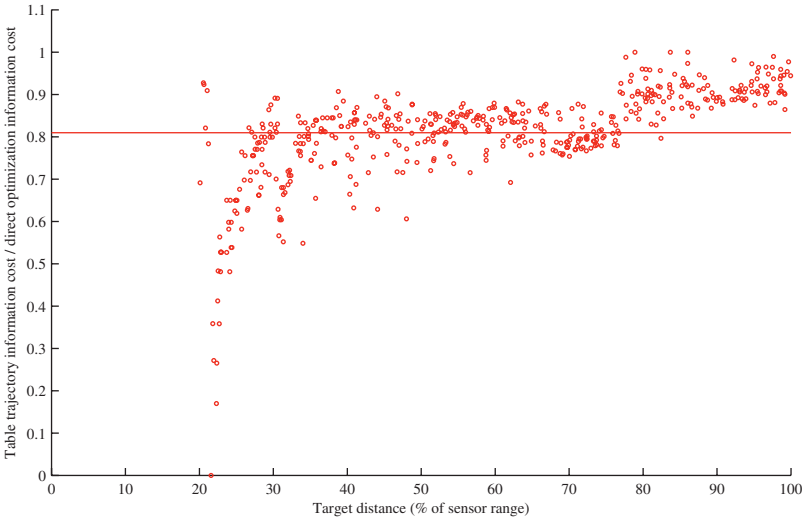


Fig. 12 Comparison of the cost for trajectories from the lookup table vs a direct optimized trajectory for 500 random target locations.

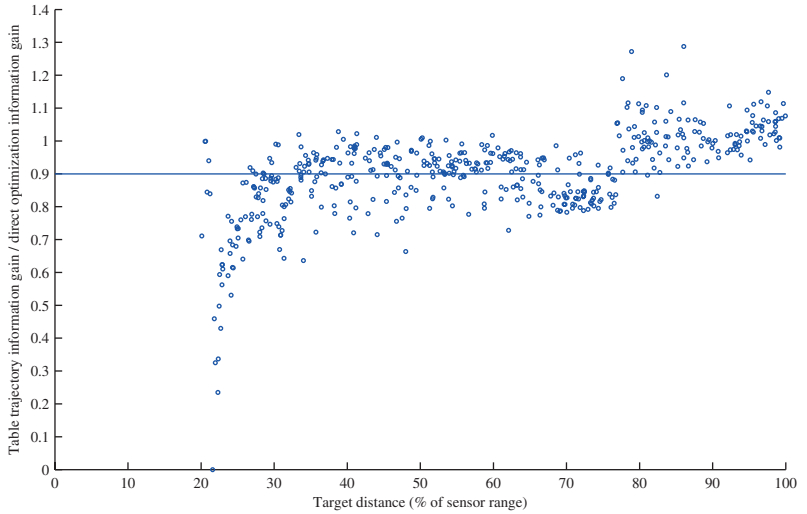


Fig. 13 Comparison of the information gain alone for trajectories from the lookup table vs a direct optimized trajectory for 500 random target locations.

The table trajectory can also be retrieved from memory and dimensionalized online much faster than an optimization can take place. Times for generating a sequence of 10 waypoints in dimensional space for a waypoint-following controller are given in Table 1 and shows that, on average, the table lookup occurs more than 130 times faster than the online optimization. It should be noted that the median optimization time of 1.28 s represents flight over 10–15% of the sensor range for a nominal MAV. Thus, trajectories computed online are likely to be obsolete before they can be flown. Note also that the CPU times reported were measured using a workstation-class computer with an AMD Opteron processor clocked at 2.6 GHz. The computation time for online optimization will be significantly longer using a processor carried onboard an MAV.

Table 1 Comparison of CPU times for generating a dimensional trajectory from the lookup table vs online optimization using a 2.6 GHz AMD Opteron processor

	Lookup table (s)	Online optimization (s)
Minimum	0.0084	0.4822
Maximum	0.0403	2.5074
Median	0.0084	1.2779
Mean	0.0089	1.2364

D. Adaptation

As the vehicle follows a trajectory selection from the table, target state estimation occurs in real-time. By selecting a new trajectory from the table when the estimated target state has changed, the target localization trajectory can be made adaptive to changes. In this research, a particular trajectory is followed for a control horizon T_c and a new trajectory is selected at the end of the control horizon. The benefits of adaptation is shown in Fig. 14. Without adaptation, the lookup table method localizes the target within 0.16 m. With adaptation and a control horizon of 2 s, the lookup table method localizes the target to within 0.075 m, or better than double the localization accuracy.

E. Discussion

Monte Carlo simulations have shown that the table trajectories will provide 90% of the information gain about a target that the online optimization would. However, this result does not take into account the advantage of fast trajectory adaptation that is possible when using the table trajectories. A study of required computational power showed that the mean time required to compute a trajectory dropped from 1.24 s using true optimization to 0.0089 s using the table trajectories on an AMD Opteron processor. Finally, allowing adaptation when using the table trajectories doubles the accuracy of the localization task. Because of the minimal computation time required (0.0089 s mean) to calculate a trajectory using the table, allowing for adaptation would have no significant effect on other observer vehicle tasks.

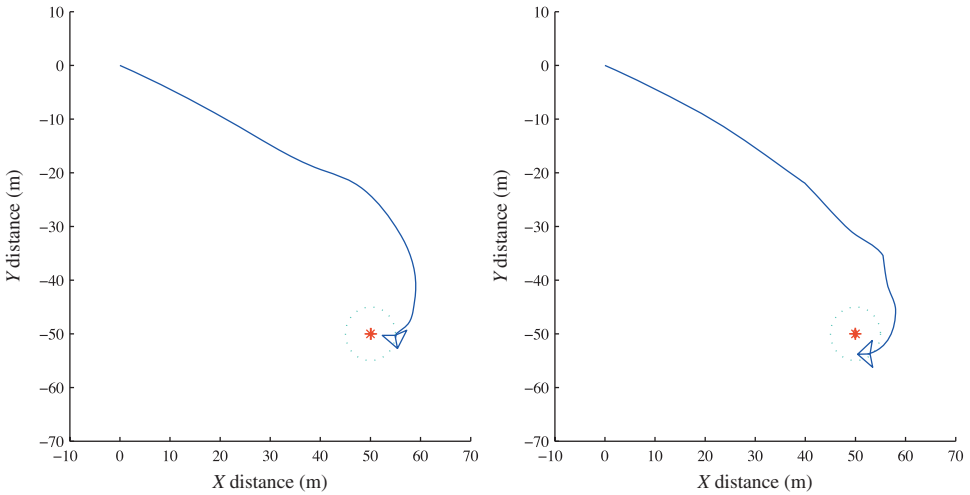


Fig. 14 Allowing for path adaptation as the target state estimate is refined improves the localization accuracy of the lookup table provided paths.

VI. Conclusion

This paper has described a method for fast adaptive trajectory generation for the problem of target localization using MAVs.

The system is based on a precomputed lookup table of trajectories. A set of nominal target locations is defined and trajectories which maximize information gain for each of the nominal target locations are computed offline. A trajectory is represented as a sequence of waypoints nondimensionalized with respect to sensor range. An observation number, which provides an approximation of the number of measurements of target state which can be obtained for a particular choice of vehicle speed, sensor range, and sensor update rate, is defined.

This parameterized approach to trajectory generation for optimal target localization significantly reduces the real-time computational load on a small- or micro- UAV's processor, freeing capacity for other tasks such as state estimation, navigation, or communication. Trajectories generated using the approach proposed here provided 90% of the information gain of the directly optimized trajectory while running 137 times faster than the direct optimization.

The initial results of the waypoint parameterization also show good localization performance. Because the trajectories are stored as a finite set of 10 waypoints, the memory requirements for this parameterization are minimal, with the entire trajectory table and associated information taking up only 10 kb of memory storage space. The waypoint-based trajectories were then dimensionalized and compared with trajectories that were optimized directly in real-space. The results of this comparison show that the two trajectories are nearly identical in real-space and the small differences have little or no effect on the localization as the path differences occur when the target is outside of the field of view of the sensor system. The waypoint-based trajectory was then dimensionalized for an observer vehicle with a significantly different observation number and compared to a trajectory created through direct optimization. Again, the two paths are nearly identical, with only small differences in some turns.

A Monte Carlo simulation consisting of 500 runs was performed using the waypoint-based trajectory table. Localization results were then compared with those obtained by optimizing a trajectory for the random target location. The dimensionalized waypoint-based table trajectories provide, on average, 81% of the information cost and 90% of the information gain, when compared to the direct optimized trajectory for a random target location. A few factors influence the localization performance and were detailed in the initial presentation of the results.

Finally, a comparison of computation times required for the trajectory design was done. This comparison shows that direct optimization of a trajectory takes 1.28 s on average, or about 10–15% of the sensor range of the observer vehicle. However, retrieving a nondimensional trajectory from the lookup table and then dimensionalizing the waypoints occurs 137 times faster, or approximately 0.0084 s. This trajectory design time is constant for trajectories of any size, from 25 to 100% of the sensor range.

References

[1] Kailath, T., Sayed, A. H., and Hassibi, B., *Linear Estimation*, 1st ed., Prentice-Hall, Upper Saddle River, NJ, March 2000.

[2] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T., *Estimation with Applications to Tracking and Navigiation*, Wiley Interscience, New York, NY, 2001.
doi: 10.1002/0471221279

[3] Simon, D., *Optimal State Estimation*, Wiley Interscience, Hoboken, New Jersey, NJ, 2008.

[4] Corbets, J. B., and Langelaan, J. W., "Parameterized Optimal Trajectory Generation for Target Localization," *AIAA Guidance, Navigation and Control Conference*, AIAA Reston, Virginia, Aug. 2007; also AIAA Paper 2007-6750.

[5] Corbets, J. B., and Langelaan, J. W., "Parameterized Trajectories for Target Localization using Small and Micro Unmanned Aerial Vehicles," *American Control Conference*, American Automatic Controls Council, Seattle, Washington, DC, June 2008.

[6] Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, Massachusetts, 1991.

[7] LaValle, S. M., *Planning Algorithms*, Cambridge Univ. Press, Cambridge, 2006.
doi: 10.1017/CBO9780511546877

[8] Huang, S., Kwok, N. M., Dissanayake, G., Ha, Q. P., and Fang, G., "Multi-Step Look-Ahead Trajectory Planning in SLAM: Possibility and Necessity," *IEEE International Conference on Robotics and Automation*, Institute of Electrical and Electronics Engineers, IEEE, Piscataway, New Jersey, NJ, April 2005.

[9] Frew, E. W., "Observer Trajectory Generation for Target-Motion Estimation Using Monocular Vision," Ph.D. Thesis, Stanford Univ., Stanford, California, CA, Aug. 2003.

[10] Grocholsky, B., Makarenko, A., and Durrant-Whyte, H., "Information Theoretic Coordinated Control of Multiple Sensor Platforms," *IEEE International Conference on Robotics and Automation*, Vol. 1, IEEE, Piscataway, New Jersey, NJ, Sept. 2003, pp. 1521–1526.

[11] Ousingsawat, J., and Campbell, M. E., "Optimal Cooperative Reconnaissance Using Multiple Vehicles," *Journal of Guidance, Control and Dynamics*, Vol. 30, No. 1, Jan. 2007, pp. 122–132.
doi: 10.2514/1.19147

[12] Sinclair, A. J., Prazenica, R. J., and Jeffcoat, D. E., "Simultaneous Localization and Planning for Cooperative Air Munitions," *International Conference on Cooperative Control*, Gainesville, FL, 2007.

[13] Nygards, J., Skoglar, P., Karlholm, J., Bjorstrom, R., and Ulvklo, M., "Towards Concurrent Sensor and Path Planning," TR, FOI, May 2005.

[14] Geiger, B. R., Horn, J. F., Delullo, A., Long, L. N., and Neissner, A. F., "Optimal Path Planning of UAVs Using Direct Collocation with Nonlinear Programming," *AIAA Guidance, Navigation and Control Conference*, Keystone, CO, 2006.

[15] Schouwenaars, T., Mettler, B., Legendre, P., and Dale, M. R. T., "Robust Motion Planning Using a Maneuver Automaton with Built-in Uncertainties," *Proceedings of the American Control Conference*, Denver, CO, 2003, pp. 2211–2216.

[16] Frazzoli, E., "Explicit Solutions for Optimal Maneuver-Based Motion Planning," *Proceedings of the 42nd Conference on Decision and Control*, IEEE, Maui, Hawaii, Dec. 2003, pp. 3372–3377.

[17] Dever, C., Mettler, B., Feron, E., Popovic, J., and McConley, M., "Nonlinear Trajectory Generation for Autonomous Vehicles via Parameterized Maneuver Classes," *Journal of Guidance, Control and Dynamics*, Vol. 29, No. 2, 2006, pp. 290–302.

[18] van der Merwe, R., and Wan, E., "The Square-Root Unscented Kalman Filter for State and Parameter-Estimation," *IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, Salt Lake City, UT, 2001.

[19] van der Merwe, R., Wan, E., and Julier, S., "Sigma Point Kalman Filters for Nonlinear Estimation and Sensor Fusion: Applications to Integrated Navigation," *AIAA Guidance, Navigation and Controls Conference*, AIAA, Providence, RI, Aug. 2004.

Michael G. Hinchey
Associate Editor