Trajectory Waypoint Spacing for Spline-Based Flight Plans

Bryan C.H. Chu
PhD Candidate, GN&C
Arizona State University
Mesa, AZ, USA

Dr. James Keller
Associate Technical Fellow
Flying Qualities
Philadelphia, PA, USA

Dr. Spring Berman
Associate Professor
Arizona State University
Tempe, AZ, USA

ABSTRACT

A new guidance optimization scheme for spacing waypoints on spline trajectories is proposed. This scheme, the *bounded area minimization algorithm*, examines sequences of 3 waypoints that sample a given spline trajectory at its constituent knot locations and moves the interior waypoint to a location on the spline trajectory that minimizes the bounded area, computed using Green's theorem, between the trajectory and the straight-line paths (legs) that connect adjacent waypoints. For spline trajectories defined by more than 3 knots, the algorithm can be applied sequentially to cover the entire chain of knots. Five motion primitives were chosen to test the performance of the optimization scheme on piecewise cubic polynomial spline trajectories. Two of these motion primitives (sinusoid and exponential) as well as two real-world trajectories that have been flight-tested on an MD530F platform were then simulated in a full nonlinear rotorcraft flight dynamics simulator to quantify and compare the effects of optimized versus baseline waypoint spacing. The bounded area minimization algorithm was extremely effective at reducing cumulative cross track error when the waypoint spacing was large enough that the aircraft trajectories closely matched the straight-line waypoint legs used in the algorithm. A method of determining the smallest waypoint spacing at which the optimization algorithm is still beneficial was proposed based on varying the wavelength of the ADS-33 Slalom MTE. For densely sampled spline trajectories that do not have knot spacings large enough to realize the benefits of the optimization algorithm, knot removal can sometimes reduce the number of waypoints required to represent the trajectory while maintaining the cumulative cross track error. However, there is no guarantee that subsequent bounded area minimization results in better performance than the baseline waypoint spacing. Determination of waypoint spacing for densely sampled spline trajectories is a recommended area of future work. A practical benefit of the bounded area minimization algorithm is that it requires very little modification to in-service coupled waypoint guidance flight director and autopilot schemes on existing aircraft, which improves the likelihood of its adoption by reducing the effort required for formal qualification and certification.

I. INTRODUCTION

Humans have long used the concept of waypoints for navigation, with the earliest waypoints defined as distinctive natural features and landmarks. With the advent of modern GPS constellations and other navigational aids, waypoints associated with precise 3-dimensional points in space could be easily entered as a latitude, longitude, and altitude in a flight computer and given as guidance commands to an automatic flight control system. The earliest automatic flight plans consisted of sequences of waypoints separated by considerable distances so that a change in leg (line segment connecting two waypoints) would occur around once every 15-30 minutes. However, with the proliferation of autonomous aerial vehicles such as small drones and the push for urban air mobility, there is a need for autonomous aircraft to fly more complex trajectories for obstacle avoidance, traffic re-routing, and contingency management. One method of defining such trajectories is to sample them in space with discrete waypoints. However, due to the complex nature of these trajectories (an accurate approximation requires piecewise polynomials of degree 3 or higher between waypoints), the choice of waypoints and spacing between waypoints is non-trivial, entailing tradeoffs among the discrepancy between the desired trajectory and its approximation, the dynamic feasibility of the approximated trajectory, and its ease of implementation using aircraft guidance leg-switching logic.

To address this problem, this paper proposes the *bounded area minimization algorithm*, an optimal method of placing the interior waypoint of a sequence of three that minimizes the bounded area between the original trajectory and the straight-line path connecting the waypoint sequence. In addition, the optimization algorithm is applied to trajectories that have more than three waypoints using a chaining process. The algorithm is first tested on five motion primitives that are common components of rotorcraft flight trajectories, and then on more complex trajectories that can be used for real-time rotorcraft trajectory planning. Aircraft trajectory tracking error over optimized and non-optimized waypoint-based flight plans is compared in a full nonlinear rotorcraft flight

Presented at the Vertical Flight Society's 6th Decennial Aeromechanics Specialists' Conference, Santa Clara, CA, USA, Feb 6-8, 2024.

dynamics simulation in order to predict and measure the benefit of using this algorithm. Applications of the algorithm to in-service aircraft, conditions under which the algorithm improves or degrades tracking performance, and directions of future work are also discussed. We first define the type of flight trajectory that we consider in this work: a piecewise polynomial spline of degree 3 or higher, which is a typical output generated by high-level path planners [1] and has been used for rotorcraft trajectory following in [2], [3], [4], [5], [6], and [7]. Next, we discuss approaches to sampling an arbitrary spline or sequence of splines and flying such trajectories using waypoint switching algorithms. Finally, we describe previous work on solving the trajectory-tracking and path-following problems for aircraft and the advantages and shortcomings of these methods.

A. Trajectory Definition

A typical spline trajectory for flight planning purposes may be constructed using 150 to 200 knots. Splines are piecewise continuous polynomial-based trajectories. Each point on a spline is based on the value of a parameter, typically identified in notation as t; a set of polynomial basis functions; and a vector set of physical locations that permit transcription of real-valued polynomials evaluated at a specific parameter value into a location. In this context, knots are the set of discrete samples of the parameter that are used to shape the basis functions. The most fundamental basis function set is the power series. For this basis function set, a knot is defined as the value of the parameter that sets the point on the spline where two piecewise polynomials are joined together. When the parameter, t, equals a knot value, the corresponding point on the spline is denoted as a knot point. Because of this correspondence between knots (real-valued scalars) and the locations along the spline they produce, this paper will henceforth refer to knots as the points along the spline, otherwise known as the knot times. Between each pair of adjacent knots, the spatial coordinates $\boldsymbol{p}(t) = (x(t), y(t), z(t))$ of the spline are parametric polynomials in terms of the parameter t. For example, if knot n is located at position $\boldsymbol{p}_n = (x_n, y_n, z_n)$ in the global coordinate system and knot $n+1$ is located at $\boldsymbol{p}_{n+1} = (x_{n+1}, y_{n+1}, z_{n+1})$, then a cubic spline segment between knots n and $n+1$ can be described as follows:

$$\boldsymbol{p}(t) = \mathbf{a}_{n,0}(t-t_n)^3 + \mathbf{a}_{n,1}(t-t_n)^2 + \mathbf{a}_{n,2}(t-t_n) + \mathbf{a}_{n,3}, \quad t \in [t_n, t_{n+1}), \tag{1}$$

where $\mathbf{a}_{n,0}, \mathbf{a}_{n,1}, \mathbf{a}_{n,2}, \mathbf{a}_{n,3} \in \mathbb{R}^3$ are vectors of constant coefficients, t_n and t_{n+1} are the knot parameters at the start and end of the spline, and we adopt the nomenclature $\boldsymbol{p}(t_n) = \boldsymbol{p}_n$, $\boldsymbol{p}(t_{n+1}) = \boldsymbol{p}_{n+1}$. Note that when referring to individual elements of the vectors of constant coefficients, the following nomenclature is used: $\mathbf{a}_{n,0} = \left(C_{x_{n,0}}, C_{y_{n,0}}, C_{z_{n,0}}\right)$, $\mathbf{a}_{n,1} = \left(C_{x_{n,1}}, C_{y_{n,1}}, C_{z_{n,1}}\right)$, $\mathbf{a}_{n,2} = \left(C_{x_{n,2}}, C_{y_{n,2}}, C_{z_{n,2}}\right)$, and $\mathbf{a}_{n,3} = \left(C_{x_{n,3}}, C_{y_{n,3}}, C_{z_{n,3}}\right)$. In this paper, we focus only on 2-dimensional trajectories in the xy-plane, but our approach can easily be extended to trajectories in 3-dimensional space as well as trajectories that include the aircraft heading. Moreover, polynomials of degree higher than 3 may be used to define the spline, depending on the application. The works [8] and [9] provide examples of using spline trajectories for aircraft path planning and guidance.

There has been considerable work on developing time-based guidance algorithms to fly splines. For example, instantaneous velocity, acceleration, and jerk commands can be obtained from the spline trajectories simply by successive differentiation of the defining expressions. However, time-based guidance requires that the aircraft arrive at each checkpoint on the flight plan precisely in time for future guidance commands to be accurate and useful. Due to winds, external disturbances, or unexpected events, it is possible for the aircraft to fall behind or ahead in time from the desired position specified by the spline trajectory and thus any time-based guidance scheme that was based on these trajectory equations must then be corrected. To circumvent these challenges, we investigate waypoint-based guidance as a possible alternative, where the waypoints are defined by sampling spline trajectories in space. An aircraft can follow such a flight plan by implementing a waypoint switching algorithm, described in the next subsection.

B. Waypoint Switching Algorithms

Flying waypoints is not the only way to fly general trajectories in space. For example, [1] and [10] discuss methods of trajectory guidance based on proportional-navigation, a technique that is often used in missile homing guidance. The work [3] describes a plethora of methods for tracking nontraditional and curvilinear trajectories, one of which is waypoint guidance. However, all of the guidance algorithms except for waypoint guidance require significant changes to in-service autopilot software, which would incur considerable cost and necessitate laborious formal qualification efforts. We chose a spline-following waypoint guidance approach since it would require minimal changes to current software and processes, thus facilitating its adoption on aircraft.

Figure 1 illustrates a string of waypoints, denoted by the open circles labeled $n-1, \ldots, n+2$, that sample a spline trajectory shown in red. Note that the waypoints may or may not coincide with the points associated with the knots of the spline. The solid black lines connecting pairs of adjacent waypoints are the legs of the flight plan, and the dashed line represents the trajectory of an aircraft that follows the flight plan. The small filled circles at the intersections of the legs with the aircraft trajectory represent the points at which the vehicle changes to the next leg in its flight plan and/or intercepts the next leg. The waypoint labeled n is the waypoint that the aircraft is currently headed to, $n-1$ is the waypoint it has come from, and $n+1, n+2, \ldots$ are the successive waypoints in the flight plan after waypoint n. As the dashed line indicates, the aircraft does not necessarily

have to reach waypoint n before switching to $n+1$, and it is not constrained to stay on the legs for the entire flight.

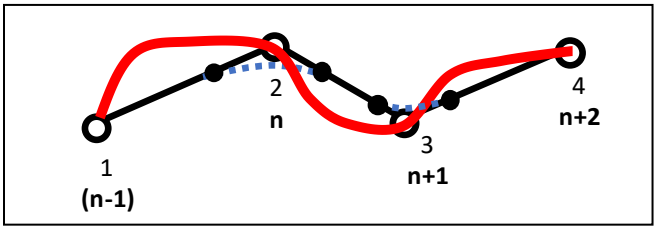

Figure 1: String of Waypoints Sampling a Spline Trajectory

The distance from the current TO waypoint at which the vehicle switches from the current leg to the next leg is known as the change distance and can be calculated using a variety of techniques. The techniques we introduce are based on coordinating banked turns at CRUISE and directed thrust vectoring in LOW SPEED (LS) flight. Since the boundary between CRUISE and LS flight is often an arbitrary speed cutoff, the LS regime will sometimes also be referred to more accurately as "Directed Thrust." Figure 2 shows the geometric setup of an aircraft approaching a leg switching point in (a) CRUISE and (b) LS flight. The change distance computation for CRUISE is based on assuming the aircraft travels at a constant bank angle during the change and thus inscribes a circle between the waypoint legs. The change distance computation for LS flight assumes a constant aircraft acceleration that changes its initial velocity v_i to a final velocity, v_f, whose direction is along the new leg. For both change distance calculations, there is an additional lead distance, Δx, that is up to the designer to specify and that varies between the CRUISE and LS cases. The total change distance, Δ_{ij}, and the distance d_{ij} between waypoints are specified in Figure 2 and later used in the constraints for the optimization problem formulation.

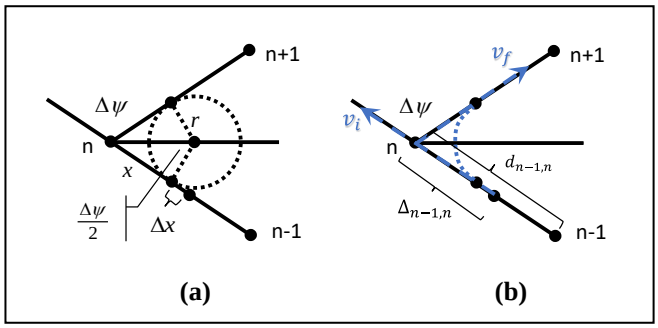

Figure 2: Waypoint Change Distance Formulations for (a) CRUISE and (b) LS Flight

Given a ground speed v which is assumed to be constant in magnitude throughout the transition $(v_f = v_i)$, ground track angle change $\Delta\psi$, maximum bank angle command limit ϕ_{max} in CRUISE, maximum planar acceleration command limit $\dot{v}_{max}$ in LOW SPEED (LS), and local acceleration of gravity g, the minimum change distance for CRUISE and LS conditions can be defined as:

$$\Delta_{CRUISE} = \frac{v^2}{g\,tan\phi_{max}} tan\left|\frac{\Delta\psi}{2}\right|, \quad \Delta_{LS} = \frac{v^2\sqrt{2(1-cos\Delta\psi)}}{2\dot{v}_{max}} \tag{2}$$

It is important to note that these change distance computations are minimums; they do not account for the distance to roll into a banked turn in the CRUISE condition or the distance to establish the altered rotor thrust necessary for a direction change in the LS condition. Thus, real aircraft dynamics, switching logic, and other unaccounted factors may require the waypoint separation to be increased, for the vehicle to fly the proposed trajectory accurately. Figure 3 plots an example spline trajectory sampled by 180 waypoints, defined as the knots of the spline, along with the simulated path of an aircraft that follows this flight plan and the geographic location where this spline trajectory was flight tested.

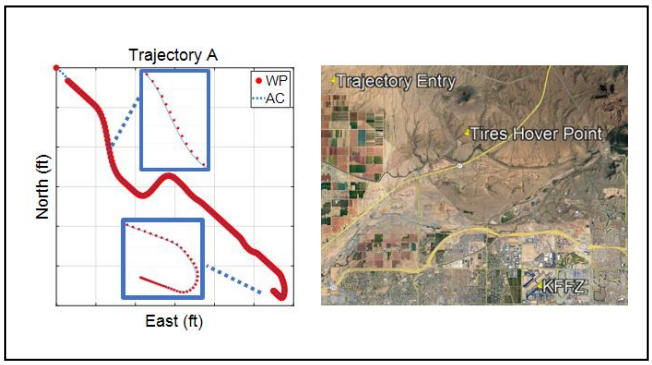

Figure 3: Example Waypoint-Sampled Spline Trajectory (WP = Waypoint, AC = Aircraft)

There is a noticeable discrepancy between the waypoint-based flight plan and the aircraft trajectory. In this study, we focus on reducing the spline trajectory tracking error that is due to the approximation of a given spline by a finite number of segments (legs); correcting for the tracking error that arises from the aircraft's flight dynamics is an avenue for future work.

C. Previous Work on Trajectory-Tracking and Path-Following Techniques for Aircraft

Most algorithms developed for aircraft trajectory tracking and path following attempt to guide the vehicle along a specified spline or through a sequence of discrete waypoints that are sampled from the spline. The survey [11] describes some commonly used path-following algorithms for fixed-wing UAVs, including carrot-chasing, nonlinear guidance law, line-of-sight (LOS) guidance, linear quadratic regulator (LQR) control, and vector field guidance. A unique ahead-time method for carrot-chasing guidance is discussed in [12]. The nonlinear guidance law approach and its derivation from the popular proportional-navigation missile homing guidance algorithm are described in [13], along with results on its performance in flight tests with several fixed-wing small-

scale aircraft. Various spline-based waypoint guidance algorithms have been developed and demonstrated in helicopter simulations and real aircraft, as discussed in [3], [4], [5], [6], [7], and [14]. Rotorcraft platforms that have in the past been suitable for developing these algorithms and could serve as a template for future autonomy development are described in [15] and [16]. However, none of these works present a formal methodology for sampling waypoints from a spline, such as selecting a set of waypoints that optimizes some property of the resultant flight path. The paper [17] describes the optimal transition trajectory between a series of three waypoints, but the waypoint coordinates are assumed to be fixed beforehand.

II. BOUNDED AREA MINMIZATION FORMULATION

We assume that the specified flight trajectory (or a segment of it), which we will refer to simply as the *trajectory*, is a smooth planar curve defined by a parametric representation $\boldsymbol{p}(t) = \big(x_p(t), y_p(t)\big)$ in terms of the parameter, t. We consider the problem of approximating the trajectory with two straight-line legs that connect a sequence of 3 waypoints, labeled $n-1$, n, and $n+1$, that are constrained on the trajectory and located at positions $\boldsymbol{q}_{n-1}$, $\boldsymbol{q}_n$, and $\boldsymbol{q}_{n+1}$ in the plane, respectively. We will refer to the path consisting of the straight-line legs as the *leg path* and denote it by $\boldsymbol{q}(t) = \big(x_q(t), y_q(t)\big)$. Note that an aircraft would not always fly exactly along the leg path, but rather could begin the transition from one leg to another before or even after a waypoint is reached, as shown in Figure 1. The initial and final parameter values are defined as t_{n-1} and t_{n+1}, respectively, so that $t \in [t_{n-1}, t_{n+1}]$ for both $\boldsymbol{p}(t)$ and $\boldsymbol{q}(t)$. The waypoints $n-1$ and $n+1$ are fixed at the endpoints of the trajectory, i.e., $\boldsymbol{q}(t_{n-1}) = \boldsymbol{q}_{n-1} \equiv \boldsymbol{p}(t_{n-1})$ and $\boldsymbol{q}(t_{n+1}) = \boldsymbol{q}_{n+1} \equiv \boldsymbol{p}(t_{n+1})$. The interior waypoint n is constrained to lie on the trajectory $\boldsymbol{p}$ and is initially set at a location $\boldsymbol{q}(t_n) = \boldsymbol{q}_n \equiv \boldsymbol{p}(t_n)$. Figure 4 illustrates the approximation of a spline trajectory by several leg paths with waypoints defined as the knots of the spline and the interior waypoint n located at different possible points (dashed circles) along the spline.

The optimization variable is the parameter t_* that defines the optimized location $\boldsymbol{q}(t_*)$ of the interior waypoint n. Given a trajectory $\boldsymbol{p}(t)$, $t \in [t_{n-1}, t_{n+1}]$, our optimization objective is to find the value of t_* for which the leg path $\boldsymbol{q}(t)$ most closely approximates $\boldsymbol{p}(t)$. One possible approach is to find t_* that minimizes the cumulative cross track error, defined in Appendix A, between $\boldsymbol{q}$ and $\boldsymbol{p}$. However, based on the authors' experience, various numerical challenges arise in the formulation and solution of an optimization problem with this quantity as the objective function. Instead, we define the objective function as the area $A(t)$ of the bounded region enclosed by $\boldsymbol{p}(t)$ and $\boldsymbol{q}(t)$, $t \in [t_{n-1}, t_{n+1}]$. In Figure 4, $A(t)$ is the sum of the blue-shaded regions when $\boldsymbol{q}(t)$ is defined as the solid black leg path. Since all parameters used to compute the area are fixed except for the optimization variable t_*, we write the area as $A(t_*)$. Appendix A demonstrates that minimizing the area $A(t_*)$ is equivalent to minimizing the cumulative cross track error between $\boldsymbol{q}$ and $\boldsymbol{p}$ for the case where $\boldsymbol{p}$ is defined as one of the two motion primitives described in Section III.

The optimization problem, which we refer to as the *Bounded Area Minimization Problem*, can now be formulated as follows:

$$\boldsymbol{minimize}: A(t_*)$$

$$\boldsymbol{subject\ to}: \boldsymbol{q}(t_*) = \boldsymbol{p}(t_*) \text{ for some } t_* \in [t_{n-1}, t_{n+1}]$$

$$\boldsymbol{q}(t_{n-1}) = \boldsymbol{p}(t_{n-1}),\ \boldsymbol{q}(t_{n+1}) = \boldsymbol{p}(t_{n+1})$$

$$d_{n-1,n} > \Delta_{n-1,n},\ d_{n,n+1} > \Delta_{n,n+1} \qquad \textbf{(3)}$$

Solving this problem can be visualized as sliding the interior waypoint n along the trajectory $\boldsymbol{p}$, as illustrated in Figure 4, until it is at a location $\boldsymbol{p}(t_*)$ for which the area $A(t_*)$ is minimized. In Figure 4, the area of the blue shaded region represents $A(t_*)$ for which $t_* = t_n$.

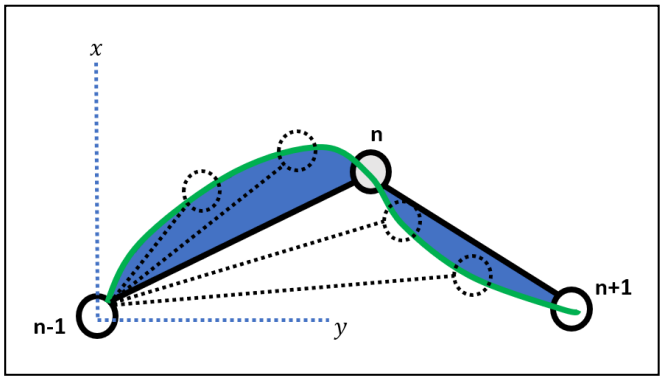

Figure 4. Sliding the Interior Waypoint along a Trajectory for Bounded Area Minimization

The bounded area $A(t_*)$ can be calculated by applying Green's theorem in the plane. In this paper, we only consider trajectories $\boldsymbol{p}(t)$ that intersect the leg path $\boldsymbol{q}(t)$ at three values of t: t_{n-1}, t_*, and t_{n+1}. Thus, area $A(t_*)$ is the sum of the areas of two bounded regions, each enclosed by $\boldsymbol{p}(t)$ and $\boldsymbol{q}(t)$, one for $t \in [t_{n-1}, t_*]$ and the other for $t \in [t_*, t_{n+1}]$. Green's theorem only applies to simply connected regions, so it must be applied separately to these two regions.

Bounded Area Minimization takes advantage of the fact that the spline trajectory equations are parametrized in time, $\boldsymbol{t}$, so it is easy to evaluate the path integral along the spline and back along the leg between two successive waypoints. Green's Theorem adapted to compute the area of a region D bounded by a closed curve C is:

$$A = \iint_D dA = \oint_C (Ldx + Mdy) = \frac{1}{2}\oint_C (-ydx + xdy)$$

$$\Rightarrow A = \frac{1}{2}\oint_{t_{n-1}}^{t_n} (-yx' + xy')\, dt \qquad \textbf{(4)}$$

When applying Green's Theorem in a right-handed coordinate system, the area A is positive if the curve C is traversed in the direction for which the region D is to the left of the curve when looking down on the x-y plane, as illustrated in Figure 5 by the clockwise orientation (signified by the arrows) of the curve that bounds region D.

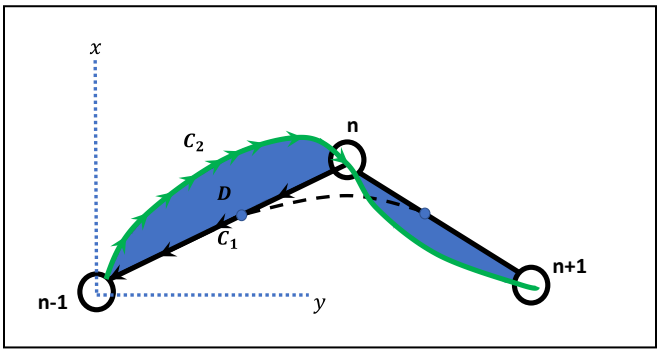

Figure 5. Application of Green's Theorem for Bounded Area Computation

The main advantage of using Green's Theorem is that the area enclosed by the trajectory and legs can be written as a function of the location of the interior waypoint parameter, $\boldsymbol{t}_*$. This allows the area formula to be minimized directly using gradient-based constrained nonlinear programming. Moreover, the formula can be applied to trajectories that can be written as closed-form expressions or as piecewise polynomial cubic spline approximations. Since this formula is based only on the geometry of the trajectory and leg path, the optimization problem does not require a model of the aircraft's flight dynamics.

Our optimization procedure finds the minimum area $A(t_*)$ by sliding the interior waypoint along the trajectory in two directions, shown in Figure 6: a "pull" toward waypoint $n-1$ to parameter value t_{*-}, and a "push" toward waypoint $n+1$ to parameter value t_{*+}. The figure also shows the corresponding areas $A(t_*)$ (shaded blue), which we denote as A_- for a pull operation (Figure 6-a) and A_+ for a push operation (Figure 6-b).

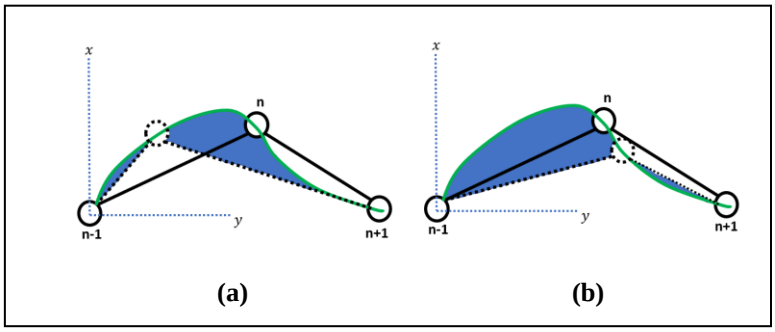

Figure 6: Bounded Area Minimization: Interior Waypoint (a) Pull and (b) Push Operations

As expected, sliding the interior waypoint along the spline trajectory changes the area $A(t_*)$. To further illustrate the impact of moving the interior waypoint, Figure 7 shows the effect of pulling or pushing this waypoint on the resultant aircraft trajectory (represented with a solid red line during the change from one leg to the next and a dashed red line otherwise.)

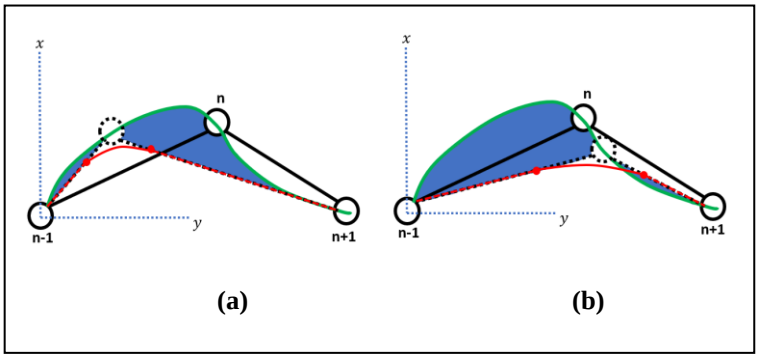

Figure 7: Effect of (a) Pulling and (b) Pushing the Interior Waypoint on the Aircraft Trajectory

The red aircraft trajectory closely follows the leg path except for the change segment, where it deviates as the aircraft switches from one leg to the next. In addition, sliding the interior waypoint changes the distance between the aircraft trajectory and the commanded green spline trajectory.

Table 1 outlines the main steps of an algorithm for solving the optimization problem. The algorithm can be applied to flight plans that consist of more than three waypoints by iterating through successive sequences of three waypoints and optimizing the location of the interior waypoint in the current sequence. This procedure is used for the complex trajectories discussed in Section IV.

Table 1: Bounded Area Minimization for A Chain of 3-Waypoint Sequences

Algorithm: Bounded Area Minimization Using Green's Theorem for Spline Trajectory Waypoint Spacing	
Input: Times t_{n-1}, t_{n+1} and the corresponding locations of waypoints n – 1, n + 1; trajectory $\boldsymbol{p}(t), t \in [t_{n-1}, t_{n+1}]$, or a cubic spline approximation of this trajectory with accompanying coefficients **Output:** Optimal time t_* and the corresponding optimal location of interior waypoint n **Select:** Initial guess for time $t_* = t_n$ and the corresponding location of waypoint n	
Compute area A_- as a function of t_{*-} using Green's Theorem for the case of pulling waypoint n forward (decreasing time)	
Compute area A_+ as a function of t_{*+} using Green's Theorem for the case of pushing waypoint n backward (increasing time)	
Minimize both area functions A_- and A_+ subject to the following constraints:	
Minimum distance between successive waypoints must be greater than the change distance, Δ_{ij}, computed between legs $n-1$ and n when pulling or legs n and $n+1$ when pushing.	
Decision variables must remain within the following bounds: $t_{n-1} < t_{*-} \leq t_n$ and $t_n \leq t_{*+} < t_{n+1}$	
Move waypoint n to location $\boldsymbol{p}(t_*)$, where $$t_* = \begin{cases} t_{*-}, & if\ t_{*-} < t_n \\ t_{*+}, & if\ t_{*+} > t_n \\ t_n, & if\ t_{*+} = t_{*-} = t_n \end{cases}$$ Proceed to the next sequence of 3 waypoints $(n+1, n+2, n+3)$	

In processing a longer chain of waypoints, it is recommended to skip 2 waypoints so that the area optimization algorithm always starts with a waypoint sequence that has not been already altered in a previous optimization run.

III. EVALUATION OF OPTIMIZATION ALGORITHM ON MOTION PRIMITIVES

The area minimization algorithm was tested on five motion primitives and cubic spline approximations of these trajectories: straight line, semicircle, sinusoid, decaying exponential, and logarithmic spiral. For all primitives, the optimal time t_* (yielding the optimal location $\boldsymbol{p}(t_*)$ of the interior waypoint) that minimizes the area $A(t_*)$ was computed analytically to ensure that the result from the optimization algorithm was reasonable. The analytical computation for the straight-line motion primitive is presented in this section; computations for the remaining primitives are given in Appendix B. Note that the trajectory arc length of the motion primitives must be selected with care. Although the optimization algorithm can be applied to trajectories of any length, the modeled aircraft physical capabilities and waypoint guidance switching logic constraints in the nonlinear flight dynamics simulator required the waypoint spacing to have some minimum value to demonstrate reduced cross-track error.

Primitive 1: Straight Line

The straight line is the simplest motion primitive. For a straight-line trajectory $\boldsymbol{p}$, the bounded area minimization problem does not have a unique solution t_* since $\boldsymbol{p}$ and $\boldsymbol{q}$ are identical, which implies that $A(t_*) = 0$ for all $t_* \in [t_{n-1}, t_{n+1}]$ and the cumulative cross-track error between $\boldsymbol{p}$ and $\boldsymbol{q}$ is zero. The solution can be arrived at from visual inspection or by writing the expression for $A(t_*)$ using Green's Theorem:

$$A(t_*) = \frac{1}{2}\left[\int_{t_n}^{t_{n-1}} \left(-y_q x'_q + x_q y'_q\right) dt + \int_{t_{n-1}}^{t_n} \left(-y_p x'_p + x_p y'_p\right) dt + \int_{t_{n+1}}^{t_n} \left(-y_q x'_q + x_q y'_q\right) dt + \int_{t_n}^{t_{n+1}} \left(-y_p x'_p + x_p y'_p\right) dt\right] \quad \textbf{(5)}$$

Substituting $x_q(t) = x_p(t)$ and $y_q(t) = y_p(t)$ into the previous formula results in:

$$A(t_*) = \frac{1}{2}\left[\int_{t_n}^{t_{n-1}} \left(-y_p x'_p + x_p y'_p\right) dt + \int_{t_{n-1}}^{t_n} \left(-y_p x'_p + x_p y'_p\right) dt + \int_{t_{n+1}}^{t_n} \left(-y_p x'_p + x_p y'_p\right) dt + \int_{t_n}^{t_{n+1}} \left(-y_p x'_p + x_p y'_p\right) dt\right] \quad \textbf{(6)}$$

Rearranging terms and inverting the polarity of the second and fourth integral terms when switching the limits of integration for them yields the expected result:

$$A(t_*) = \frac{1}{2}[0 + 0] = 0 \quad \textbf{(7)}$$

This implies that any $t_* \in [t_{n-1}, t_{n+1}]$ can be chosen for the location $\boldsymbol{p}(t_*)$ of the interior waypoint. In practice, this requires that a minimum change angle, $\Delta\psi$, be set under which the area minimization algorithm is not run, since round-off error and digit precision makes it very difficult to draw a completely straight line through three points in space stored in a computer.

Primitive 2: Semicircle

The semicircle is another commonly used motion primitive in flight plans. The leg path inscribes a triangle within the semicircle. The area $A(t_*)$ is the sum of the areas between the semicircle and the two segments comprising $\boldsymbol{q}$. For both the trajectory and its cubic spline approximation, the optimization algorithm computed the optimal time as $t_* = 7.5s$, which places the interior waypoint at a location $\boldsymbol{p}(t_*)$ that is equidistant and equally spaced in time from the other two waypoints.

Primitive 3: Sinusoid

The sinusoid was chosen because it can be used as a basis function in a Fourier series to represent any periodic signal. As for the semicircle, the leg path inscribes a triangle within the half-period of a sinusoid, and the area $A(t_*)$ is defined similarly. For both the trajectory and its cubic spline approximation, the optimization algorithm computed the optimal time as $t_* = 7.5s$, again placing the interior waypoint at a location equidistant and equally spaced in time from the other two waypoints.

Primitive 4: Decaying Exponential

The decaying exponential was chosen to represent those trajectories which involve asymptotic approaches to a desired state. As for the semicircle and sinusoid, three waypoints can be used to inscribe a triangular path within the decaying exponential. The optimal time was computed as $t_* = 4.8s$ for the exact trajectory and $t_* = 4.6s$ for its cubic spline approximation. This 0.2-second time discrepancy can be attributed to the error between the exponential function and its cubic spline approximation. In our flight simulation tests at 15 knots (see Figure 8-b), it did not produce a significant difference in the interior waypoint locations. Even at 60 knots, a 0.2-second delay results in a position error that is less than 27.5 ft, the diameter of the main rotor of the MD530F platform.

Primitive 5: Logarithmic Spiral

The logarithmic spiral was used as a motion primitive because it represents a trajectory with a continuously changing radius

of curvature. Figure 17 shows a plot of this trajectory with three waypoints spaced $1.5s$ apart (a total of $3.0s$ between waypoints $n-1$ and $n+1$), with the constants defined as $\alpha = 2\pi, b = 0.2$. The optimization algorithm computed the optimal time as $t_* = 1.68s$ for the exact trajectory and $t_* = 1.64s$ for the cubic spline approximation. This makes sense since the radius of curvature is constantly increasing, and therefore the interior waypoint must be pushed forward slightly from the optimal time for a semicircle (constant radius of curvature), which would be at the time midpoint $1.5s$. The small difference between the optimized solution for the exact trajectory and its cubic spline approximation was due to the discrepancy between these two curves and was deemed inconsequential (a 0.04-second delay corresponds to a position error of less than 5 feet at 60 knots, smaller than the error for the exponential trajectory).

The cubic spline approximations to the motion primitive trajectories were generated by specifying the time, position, and velocity boundary conditions at the three waypoints which sampled the original trajectory and then solving for the cubic coefficients that met these constraints. The trajectories for the five motion primitives were defined with the lengths and traversal times listed in the second column of Table 2, which were chosen based on an aircraft's typical ground speed. The third, fourth, and fifth columns of Table 2 list the times t_* that were computed from the optimization algorithm. The third column shows the optimization results for the exact trajectories, whereas the fourth and fifth columns show the results for the cubic spline approximations. There are two times t_* listed for the spline approximations because these approximations required different area expressions in the optimization algorithm (see Table 2), depending on whether the interior waypoint n was pulled forward from its original time t_n ($t_* < t_n$) or pushed backward ($t_* > t_n$).

Table 2: Optimal Times Computed for Motion Primitive Trajectories and their Cubic Spline Approximations

Motion Primitive	t_* (traj)	t_* pull (spl)	t_* push (spl)
Straight Line • 1000ft travel • $t_{n+1} = 15s$	$0.5t_{n+1}$	$0.5t_{n+1}$	$0.5t_{n+1}$
Semicircle • 500ft radius • $t_{n+1} = 15s$	$0.5t_{n+1}$	$0.5t_{n+1}$	$0.5t_{n+1}$
Sinusoid • 500ft amplitude • $t_{n+1} = 15s$	$0.5t_{n+1}$	$0.5t_{n+1}$	$0.5t_{n+1}$
Decaying Exponential • 1000ft initial point • $t_{n+1} = 15s$	$0.32t_{n+1}$	$0.30t_{n+1}$	$0.5t_{n+1}$
Logarithmic Spiral • 500ft initial radius • $t_{n+1} = 3s$	$0.56t_{n+1}$	$0.5t_{n+1}$	$0.55t_{n+1}$

Table 2 shows that the optimization algorithm converges to the same time t_* for the straight line, semicircle, and sinusoid motion primitives and their cubic spline approximations. The small discrepancies between the optimal times t_* computed for the exponential and logarithmic spiral primitives and their spline approximations are since cubic splines cannot be fit to these trajectories without error; thus, the optimal times t_* for the exact trajectories and their spline approximations are slightly different. As mentioned previously, these discrepancies produce differences in interior waypoint location that are all within a single rotor diameter of the MD530F platform when cruising at 60 knots.

IV. EVALUATION OF OPTIMIZATION ALGORITHM IN FLIGHT SIMULATIONS

In addition to the motion primitives, the optimization algorithm was tested on cubic spline approximations of complex real-world trajectories, labeled Trajectory A and Trajectory B, that lack closed-form parametric representations. We then flew the cubic spline approximations of the motion primitives and the complex trajectories with both optimized and non-optimized waypoint spacing in a nonlinear flight dynamics simulator representative of an MD530F airframe. For the motion primitives, the optimized solutions show a clear reduction in cross-track error compared to the non-optimized solutions, even in the presence of wind. However, for the complex trajectories, there are certain cases where the optimized waypoint spacing yields similar or increased cross-track error compared to non-optimized waypoint spacing.

Table 3 summarizes the cumulative and maximum cross-track errors for all the simulation cases that were run. All percentage increases and decreases are relative to the corresponding cross-track errors from simulations run for the same trajectory with equal-time waypoint spacing. For the motion primitives, the time of waypoint $n-1$ is $t_{n-1} = 0s$, so equal-time waypoint spacing corresponds to the case where waypoint n is at half the time of waypoint $n+1$, or $t_n = 0.5t_{n+1}$.

Table 3: Cross-Track Error Results from Flight Simulations for Primitives and Complex Trajectories in Calm Air

Primitive/ Trajectory	Cases Run	Cumulative Cross-Track Error Δ	Maximum Cross-Track Error Δ
Sinusoid (Cubic Spline) **60kt CRUISE**	$t_n = 0.25t_{n+1}$	+65.9%	+161.2%
	$t_n = 0.5t_{n+1}$ (Optimum Exact and Spline)	N/A	N/A
	$t_n = 0.75t_{n+1}$	+139.8%	+301.6%
Exponential (Cubic Spline) **15kt LS**	$t_n = 0.25t_{n+1}$	-9.0%	-6.0%
	$t_n = 0.5t_{n+1}$	N/A	N/A
	$t_n = 0.75t_{n+1}$	+98.0%	+81.3%
	$t_n = 0.32t_{n+1}$ (Optimum Exact)	-50.5%	-52.9%
	$t_n = 0.30t_{n+1}$ (Optimum Spline)	-52.2%	-57.6%
Trajectory A	Optimized Waypoint Spacing	+9.1%	+19.6%
Trajectory B	Optimized Waypoint Spacing	-3.1%	-11.7%

For the Exponential 15 kt Low-Speed (LS) primitive, note the closeness of the results for the cases where t_n is defined as the optimal time t_* for the exact trajectory (Optimum Exact) or for its cubic spline approximation (Optimum Spline). This result and the graphical results from Figure 8-b confirm that the small difference in t_* computed for the exact exponential trajectory and its cubic spline approximation (see Table 2) does not significantly impact flight performance.

Sinusoidal and Decaying Exponential Motion Primitives:

The sinusoid motion primitive was approximated by a leg path with three waypoints spaced approximately 1000 ft apart, flown with an entry ground speed of 60 knots CRUISE. Trajectories were simulated in calm air and with winds up to 15 knots out of 60 degrees. As Figure 8-a shows, the effect of the simulated wind did not significantly change the aircraft's trajectory.

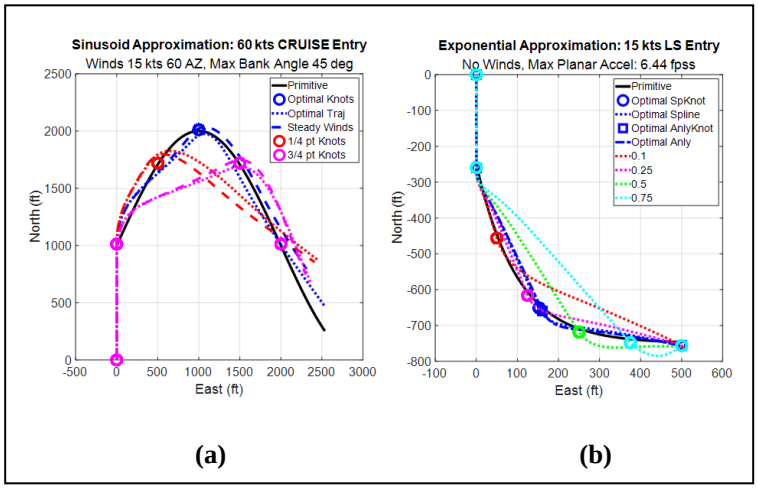

Figure 8: Simulated Aircraft Paths for the Sinusoidal and Decaying Exponential Trajectories

The exponential motion primitive was approximated by a leg path with three waypoints spaced approximately 500 ft apart, flown with an entry ground speed of 15 knots in LS mode. Trajectories were simulated in calm air with no wind. The time t_n of the interior waypoint was set to the fractions 0.1, 0.25, 0.32 (Optimum Exact), 0.30 (Optimum Spline), 0.5, and 0.75 of the total traversal time ($t_{n+1} - t_{n-1}$; $t_{n-1} = 0$). As Figure 8-b shows, the aircraft flight paths for the cases where $t_n = 0.32t_{n+1}$ and $t_n = 0.30t_{n+1}$ are very similar.

For both motion primitives, setting the time t_n of the interior waypoint to the optimal time considerably reduced the cross-track error values relative to cases with a non-optimized t_n, as shown in Table 3.

Trajectory A:

Trajectory A is composed of piecewise cubic polynomials and is based on flight-tested trajectories in Mesa, AZ on an MD530F airframe during the 2013 timeframe. It is based on a typical approach path that a helicopter might take while avoiding obstacles at high and low speeds, starting from 600ft AGL enroute to a hover landing. The waypoint spacing for these trajectories, $\Delta t = t_n - t_{n-1}$ for all pairs of sequential waypoints $n-1, n$, was approximately 1 second. We refer to this as the *baseline* waypoint spacing, for which all the waypoints and knot points coincide. For the purposes of this study, the trajectory was projected onto the x-y plane. The trajectory and the simulated helicopter flight paths for both the baseline and optimized waypoint spacings are shown in Figure 9. In the figure legends, "Spl" is the spline with knots labeled "Knots" that defines Trajectory A; "ACBase" is the helicopter's path when flying the waypoints with baseline spacing, labeled "ToBase;" and "ACOpt" is the helicopter's path when flying the waypoints with optimized spacing, labeled "ToOpt.

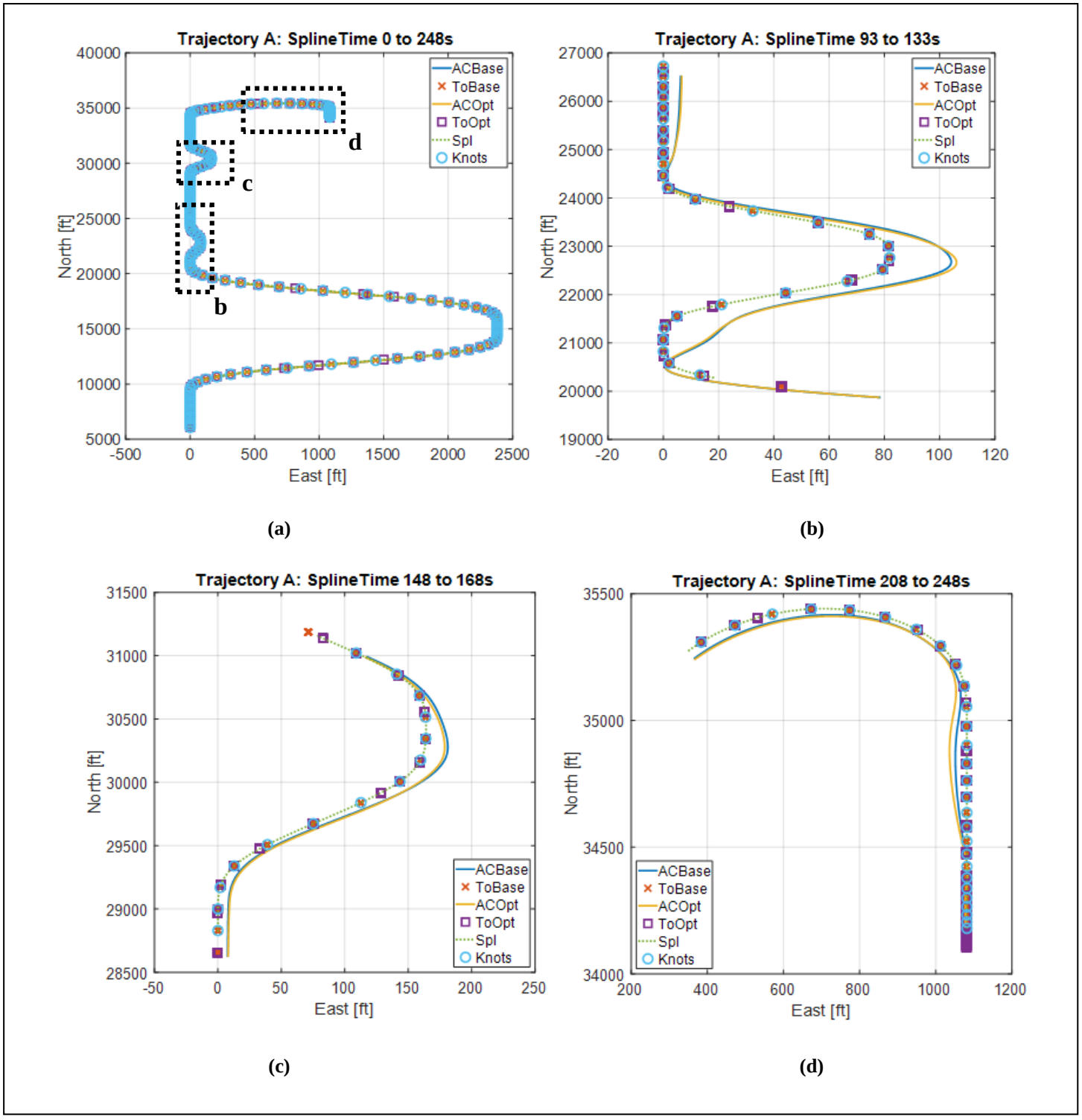

Figure 9: Trajectory A (with Zoomed-In Segments) and Simulated Flight Paths for Baseline and Optimized Waypoint Spacings

It is evident from Figure 9 that the optimization algorithm converges to a solution, as there are noticeable cases where the optimized waypoints (purple squares) are pulled backward or pushed forward with respect to the knots (cyan open circles.) However, since the knots of the spline are so close together (~100 – 200 ft apart), the optimized waypoints are as well, and the aircraft is unable to react fast enough to reach the next leg in the optimized flight plan before the guidance logic switches to the subsequent leg. Thus, the aircraft often cannot track the waypoints with either baseline or optimized spacing, as indicated in Figure 9 by the discrepancies between the simulated flight paths (ACBase, ACOpt) and the spline (Spl). In this case, the discrepancy between ACOpt and Spl exceeds the discrepancy between ACBase and Spl; Table 3 shows that the optimized waypoint spacing in fact yielded larger cross-track errors than the baseline waypoint spacing.

Trajectory B:

Trajectory B is similar to Trajectory A, except for differences in the aggressiveness of the simulated aircraft's turns and decelerations when flying waypoints along the trajectories. Table 4 lists the attributes of the two trajectories that were implemented in the simulations. As shown in Table 3, the optimized waypoint spacing for Trajectory B produced a slight reduction in cross-track error compared to the baseline waypoint spacing.

Table 4: Trajectory A vs. Trajectory B Comparison

Attribute	Trajectory A	Trajectory B
Starting Altitude AGL (ft)	600 ft	600 ft
Entry Speed (knots)	100 knots	100 knots
Number of Knots	180	180
Maximum Deceleration (ft/s^2)	-3.22 ft/s^2	-6.44 ft/s^2
Maximum Bank Angle (deg)	20 deg	30 deg
Maximum Roll Rate (deg/s)	15 deg/s	15 deg/s
Maximum Heading Rate (deg/s)	15 deg/s	15 deg/s
Average Knot Spacing (s)	1.439 s	0.958 s
Average Knot Separation (ft)	158.96 ft	150.50 ft

Trajectory C: Discretely Chirped Sinusoid ADS-33 Slalom MTE Based Trajectories

To investigate the detrimental impact of decreased waypoint spacing on the effectiveness of the optimization algorithm, a series of trajectories based on the ADS-33 slalom MTE [18] were flown in the simulator at both CRUISE and Directed Thrust/Low Speed (LS) conditions, starting with the recommended wavelengths in the ADS-33 requirement specification and slowly decreasing the wavelength if cross-track error specifications are met or increasing the wavelength if not. Note that the ADS-33 slalom MTE was originally designed as an extremely challenging maneuver to push the limits of human piloted handling qualities and was used to grade the acceptability of the agility of a platform. We now propose to use it as a method to determine useful waypoint spacing for autonomous VTOL platforms. It should be emphasized that the wavelength of the commanded sinusoid trajectory was not changed during the maneuver, as is typically done for a "chirped sinusoid" used for handling qualities assessment and system identification. Instead, the wavelength was changed after the maneuver was complete, and the simulation was run again with the reduced wavelength. After a sweep of maneuvers was run, each with a different wavelength, the results were examined in post-processing. An example ADS-33 Slalom MTE trajectory is illustrated in Figure 10.

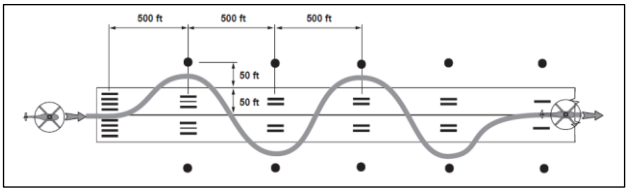

Figure 10: ADS-33 Slalom MTE from [18]

The simulated flight paths for the ADS-33 Slalom MTE trajectories at CRUISE and LS conditions are shown in Figure 11 and Figure 12, respectively. The figure legends indicate the wavelength in feet of each sinusoidal trajectory, and the circles mark the waypoints for each trajectory. The ground speeds were chosen from the performance specifications of the maneuver: GVE desired for CRUISE and DVE adequate for LS conditions. The simulations were run out of ground effect to avoid the risk of ground impact or the vehicle transitioning between altitude conditions that were in and out of ground effect. As shows for the CRUISE cases, there is a certain wavelength range at which the rotorcraft can no longer follow the commanded sinusoidal trajectory, can no longer preserve the shape of the sinusoid or follow the leg paths, and in many cases cannot meet the cross-track error requirements for the ADS-33 Slalom MTE (*turns shall be at least 50 ft from the centerline, with a maximum lateral error of 50 ft*). A similar trend can be seen in Figure 12 for the LS cases. For large wavelengths and waypoint spacings, the rotorcraft closely follows the leg paths between waypoints as expected. However, as the wavelength decreases to ~200 – 300 ft, the vehicle starts skipping waypoints, has difficulty tracking the commanded sinusoidal trajectory, and in extreme cases loses control and completely exits the course (e.g., at $\lambda = 200\ ft$). Figure 13 shows a reduced set of flight paths for both the CRUISE and LS conditions so that the transition between meeting and not meeting the minimum requirements of the maneuver can be more easily visualized.

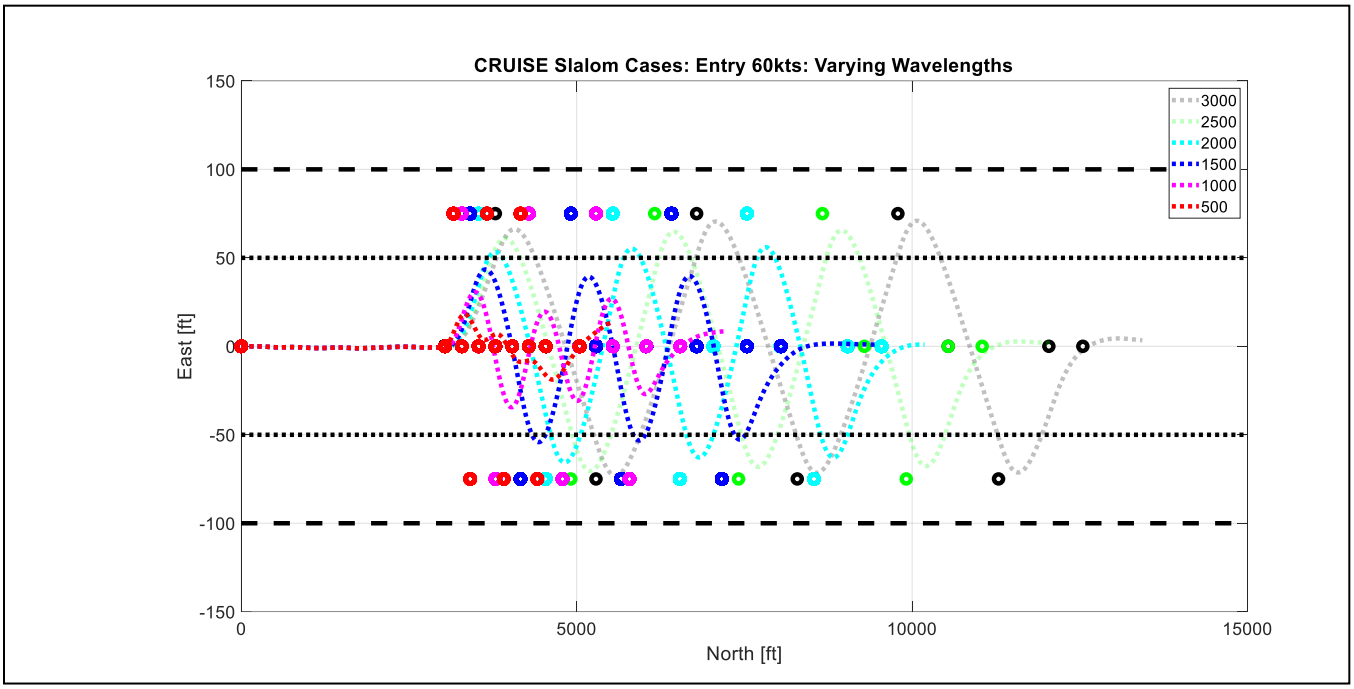

Figure 11: Simulated Flight Paths for Discretely Chirped ADS-33 Slalom MTE Trajectories at CRUISE

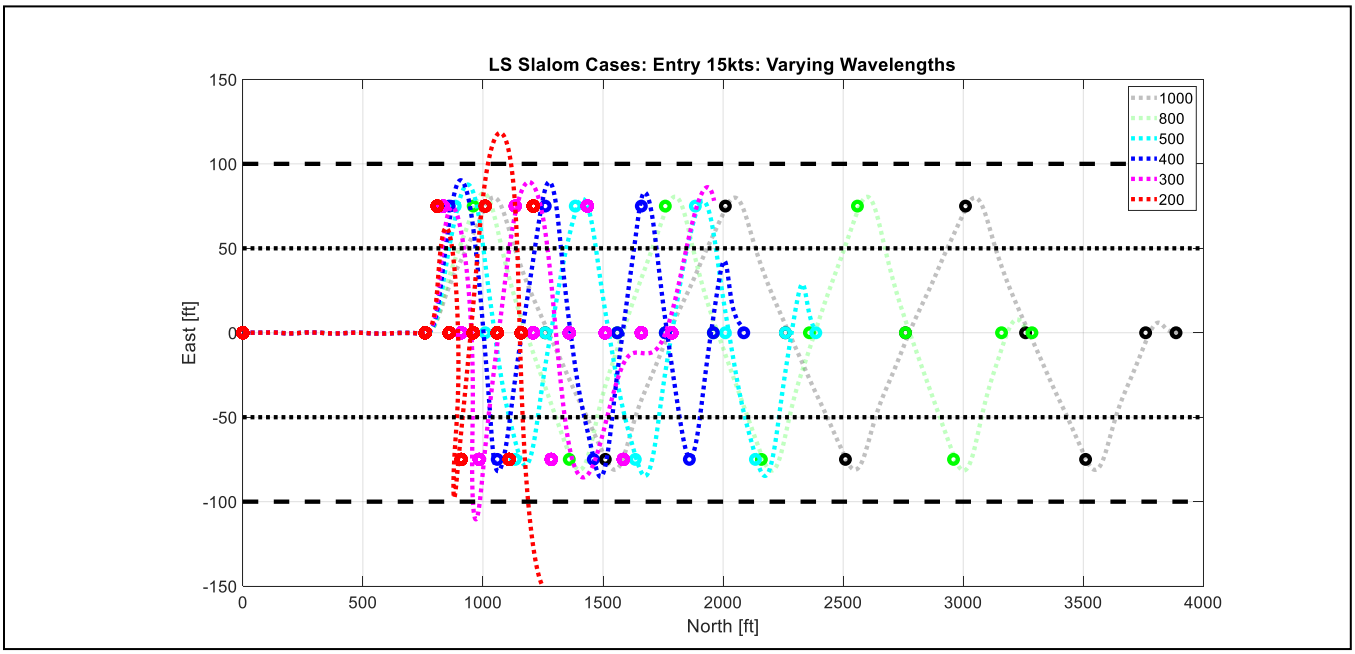

Figure 12: Simulated Flight Paths for Discretely Chirped ADS-33 Slalom MTE Trajectories at Directed Thrust/Low Speed (LS)

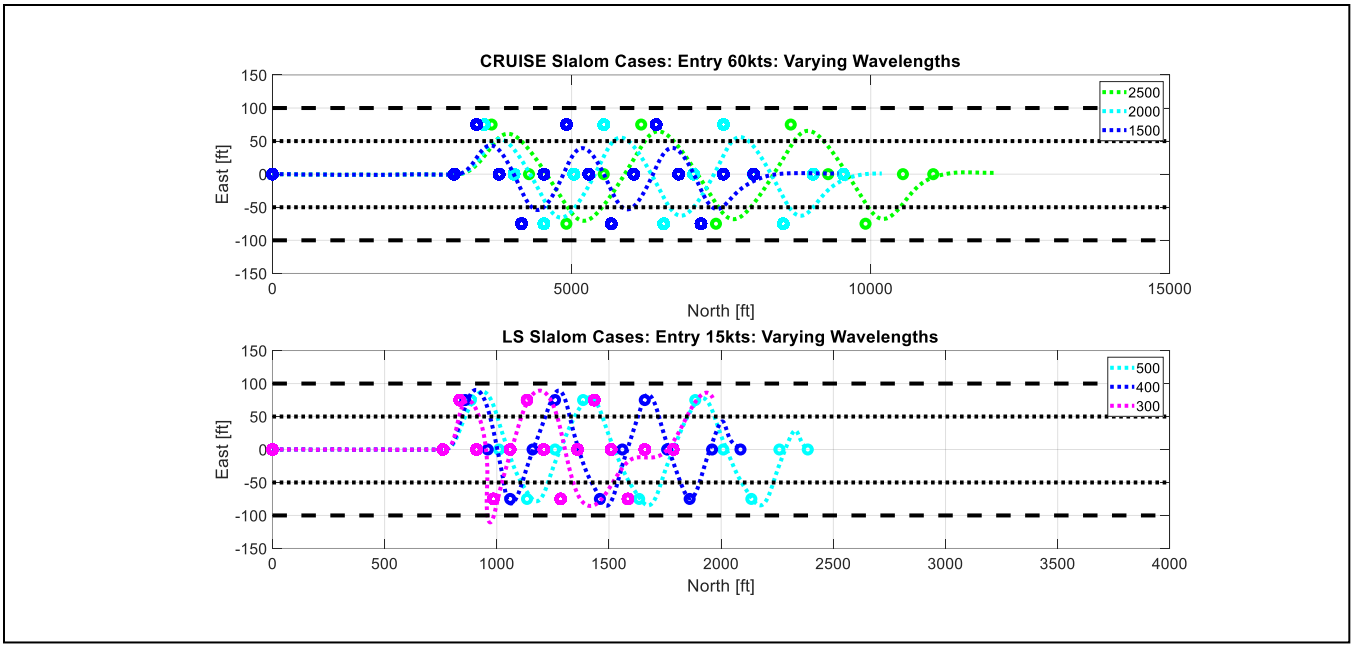

Figure 13: Subset of Simulated Flight Paths for Discretely Chirped ADS-33 Slalom MTE Trajectories

The wavelength λ at which the maneuver requirements are not met determines the minimum waypoint spacing, $\Delta_{sp} \approx \lambda/4$, for the optimization to be effective. Table 5 summarizes these lower bounds on λ and Δ_{sp} from the simulations. Note that for the CRUISE condition, the case λ = 1500 ft did not meet the cross-track error requirements but did preserve the shape of the commanded trajectory, and the case λ = 2000 ft both met the cross-track requirements and preserved the shape of the trajectory. For the LS condition, the case λ = 300 ft showed large deviations from the commanded trajectory but nearly met the cross-track error requirements, whereas the case λ = 400 ft both preserved the shape of the trajectory and met the cross-track error requirements. To more accurately determine the lower bounds on λ and Δ_{sp}, simulations may be run for trajectories with λ between 1500 and 2000 ft for the CRUISE condition and λ between 300 and 400 ft for the LS condition.

Table 5: Lower Bounds on Discretely Chirped Slalom Wavelength (λ) and Waypoint Spacing (Δ_{sp})

	LS (15 knots)	CRUISE (60 knots)
Preserves Shape of Trajectory	$\lambda > 300\ ft$ $\Delta_{sp} > 75\ ft$	$\lambda > 1500\ ft$ $\Delta_{sp} > 375\ ft$
Meets Cross-Track Error Requirements	$\lambda \geq 400\ ft$ $\Delta_{sp} \geq 100\ ft$	$\lambda \geq 2000\ ft$ $\Delta_{sp} \geq 500\ ft$

Note that the bounds in Table 5 were obtained for the particular simulation configuration, outer loop guidance implementation, and vehicle platform used in this study. The minimum recommended waypoint spacing will depend on the type of aircraft and guidance logic, and should be re-confirmed for any new simulation configuration or from flight tests. For cases where flight testing is not possible, this spacing can be obtained by running nonlinear flight simulations. However, if possible, flight testing of the discretely chirped slaloms would be the fastest way to determine the minimum recommended waypoint spacing under CRUISE and LS conditions for which the optimization algorithm reduces cross-track error.

Limitations of the Optimization Algorithm:

In general, the optimization algorithm produces the greatest improvement in cross-track error when the aircraft trajectory closely matches the leg path through the waypoints, which is often the case when the waypoints are spaced far apart. As the waypoint spacing is decreased, the aircraft dynamics and waypoint leg-switching logic tend to cause the aircraft to deviate from the prescribed leg path, so that the assumption of the Bounded Area Minimization Problem that the leg path closely approximates the aircraft trajectory is no longer valid and the legacy guidance controller starts having difficulty to accurately execute the flight plan. Under these circumstances, modifying the baseline waypoint spacing using the optimization algorithm is not always beneficial, and in some cases is detrimental. Future work involves clearly characterizing the conditions under which the optimization algorithm is and is not beneficial, and confirming the results with real aircraft flight test data.

Knot and Waypoint Reduction for Trajectories A and B:

Since the optimized waypoint spacing did not yield consistent improvement over the baseline waypoint spacing for Trajectories A and B, we investigated the effect on cross-track error of removing knots from these spline trajectories before

optimizing the waypoint spacing. The works [19] and [20] discuss systematic methods of knot reduction for B-splines, but these methods are typically very computationally expensive and involve considerable use of heuristics in determining error thresholds. Thus, they were not pursued. Simpler knot reduction schemes, such as removing waypoints based on a minimum threshold for the change angle between successive legs, were also tested. However, once enough waypoints were removed to meet the minimum spacing required for optimization to be effective (see Table 5), the cross-track error from flying the reduced-knot trajectory with baseline waypoint spacing was already considerably larger than this error for the original (denser) baseline spacing, so that optimizing the waypoint spacing was deemed to be moot. This is evidence that the method for optimally spacing waypoints on dense splines to reduce cross-track error is not based on bounded area minimization, and is a topic that should be researched in depth in the future.

V. PRACTICAL IMPLICATIONS

The proposed bounded area minimization algorithm provides a means of spacing waypoints on an arbitrary commanded spline trajectory in order to reduce tracking error. For aircraft with coupled waypoint guidance software already in-service, this algorithm allows for closer tracking of spline trajectories without the need to modify core GN&C algorithms on safety-critical flight computer systems and the ensuing formal qualification effort that would follow any safety-critical software change. It should be noted that there are many algorithms in the published literature that may be higher-performing in terms of tracking error, but all require a significant change to software and thus significant formal qualification testing. Figure 14 shows the approximate minimum waypoint spacing for several aircraft classes relative to the minimum change distance for CRUISE and LS conditions. Comparing the values from Table 5 of the lower bound on waypoint spacing for the CRUISE (375-500 ft) and LS (75-100 ft) conditions, we see that the minimum waypoint spacing for production vehicles either falls within these ranges or is much higher (e.g., the approximate minimum Cargo/Utility spacing is ~600 ft for both CRUISE and LS conditions). The Scout waypoint spacing of 100 – 150 ft was used as the baseline waypoint spacing for Trajectories A and B for demonstration purposes; it has not entered production vehicles to the authors' knowledge. This data shows why the optimization algorithm did not consistently improve tracking performance for Trajectories A and B, but also indicates that it could significantly improve tracking performance for other classes of production vehicles with more benign trajectories.

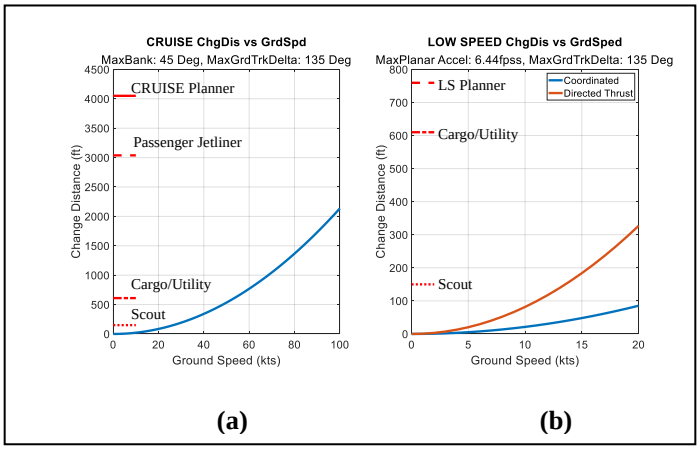

Figure 14: Minimum Change Distance vs. Approximate Minimum Waypoint Spacing for Production Aircraft

VI. CONCLUSIONS AND FUTURE WORK

The proposed optimization algorithm is a useful method for defining a sequence of waypoints that closely tracks a complex flight trajectory. It does not require any major software updates to existing aircraft and can take advantage of existing infrastructure and code. The algorithm is not applicable to cases where the baseline waypoint spacing is dense. The minimum waypoint spacing for which the algorithm produces useful solutions can be determined with test flights along discretely chirped sinusoidal slalom trajectories, either in a flight dynamics simulator or with real aircraft. Future work involves modifying the optimization algorithm to extend the range of scenarios for which it is beneficial, such as revising it to compute spacing for sequences of four waypoints instead of three (given that four knots can capture changes in curvature of a spline that three knots cannot) and relaxing the constraint that waypoints must be located on the spline trajectory.

Author contacts:

Bryan C. H. Chu, bcchu1@asu.edu
James Keller, james.f.keller@boeing.com
Spring Berman, spring.berman@asu.edu

APPENDIX A

EQUIVALENCE OF MINIMIZING AREA AND CROSS-TRACK ERROR

Figure 15 illustrates the quantities that are used to compute the cross-track error (y) and the cross-track velocity ($\dot{y}$) of an arbitrary point P, located at coordinates $(x_p(t), y_p(t))$ for some time t on a given trajectory, with respect to a leg between two waypoints n and $n+1$. The angles ψ, ψ_L, and ψ_{trk} denote the aircraft heading, leg heading, and ground track angle, respectively, all relative to North. The distances d_N and d_E denote the north and east components, respectively, of the distance of point P to waypoint $n+1$. The velocities v_N, v_E, and v_G represent the north, east, and total ground velocities of the vehicle, which are used for cross-track velocity and change distance computations.

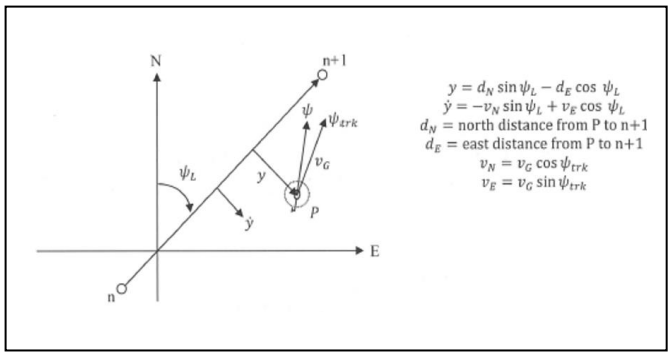

Figure 15: Cross-Track Error and Cross-Track Velocity Definitions

Using the geometry in Figure 15 (with $d_x = d_N$ and $d_y = d_E$), the cumulative cross-track error, $s_{cum}(t_*)$, between a trajectory and the first leg can be computed as:

$$s_{cum}(t_*) = \int_{t_{n-1}}^{t_*} \{(d_x \sin\psi_L) - (d_y \cos\psi_L)\} dt \qquad \textbf{(8)}$$

We will show that for trajectories defined as the semicircle and sinusoid motion primitives in Section III, the same value of time t_* minimizes both $s_{cum}(t_*)$ and the area $A(t_*)$ of the bounded region enclosed by the trajectory and its approximation by a leg path through a sequence of three waypoints. We do this by deriving the expressions for $s_{cum}(t_*)$ and $A(t_*)$, computing their derivatives with respect to t_*, setting them equal to zero, and then solving each equation for the resulting value of t_*, which minimizes the expressions. The optimal times t_* that minimize $A(t_*)$ are computed in Appendix B; here, we derive the expressions for $s_{cum}(t_*)$ for the semicircular and sinusoidal trajectories.

We first consider the semicircular trajectory, whose coordinates $(x_p(t), y_p(t))$ are defined by the parametric equations (15). If the origin of the coordinate system is placed at waypoint $n-1$, then the expressions for distances d_x and d_y can be written as follows for the leg between waypoints $n-1$ and n:

$$d_x = x_n - x_p(t) = C_{x_{n,3}} \left[1 - \sin\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t - t_{n-1})\right)\right]$$

$$d_y = y_n - y_p(t) = C_{y_{n,3}} \left[1 - 1 + \cos\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t - t_{n-1}) - \pi\right)\right] \qquad \textbf{(9)}$$

The expressions for d_x and d_y for the leg between waypoints n and $n+1$ are similar, with the coefficients redefined as $C_{x_{n+1,3}}, C_{y_{n+1,3}}$ and the leg heading ψ_L updated for this leg. We define $\psi_{L_{i\_j}}$ as the heading of the leg between waypoints i and j. Substituting the expressions for d_x and d_y into Eq. (9), the cumulative cross-track error can be evaluated as the sum of two integrals (one for each leg):

$$s_{cum}(t_*) = \int_{t_{n-1}}^{t_*} \left\{ \begin{array}{l} \left(C_{x_{n,3}} \left[1 - \sin\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1})\right)\right] \sin\psi_{L_{n-1\_n}}\right) \\ -\left(C_{y_{n,3}} \left[1 - 1 + \cos\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1}) - \pi\right)\right] \cos\psi_{L_{n-1\_n}}\right) \end{array} \right\} dt$$
$$+ \int_{t_*}^{t_n} \left\{ \begin{array}{l} \left(C_{x_{n+1,3}} \left[1 - \sin\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1})\right)\right] \sin\psi_{L_{n\_n+1}}\right) \\ -\left(C_{y_{n+1,3}} \left[1 - 1 + \cos\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1}) - \pi\right)\right] \cos\psi_{L_{n\_n+1}}\right) \end{array} \right\} dt \qquad \textbf{(10)}$$

Taking the derivative of the above expression with respect to the optimal time t_* for the interior waypoint n, applying the Fundamental Theorem of Calculus, and setting the resulting expression to zero yields:

$$\frac{d}{dt_*} s_{cum} = 0$$
$$= \left(C_{x_{n,3}} \left[1 - \sin\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1})\right)\right] \sin\psi_{L_{n-1\_n}}\right)$$
$$-\left(C_{y_{n,3}} \left[\cos\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1}) - \pi\right)\right] \cos\psi_{L_{n-1\_n}}\right)$$
$$+\left(C_{x_{n+1,3}} \left[1 - \sin\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1})\right)\right] \sin\psi_{L_{n\_n+1}}\right)$$
$$+\left(C_{y_{n+1,3}} \left[\cos\left(\frac{\pi}{t_{n+1} - t_{n-1}} \cdot (t_* - t_{n-1}) - \pi\right)\right] \cos\psi_{L_{n\_n+1}}\right) \qquad \textbf{(11)}$$

This equation is satisfied when $t_* = t_n$, since for this trajectory,

$$t_* = t_n \;\Rightarrow\; \frac{t_* - t_{n-1}}{t_{n+1} - t_{n-1}} = \frac{1}{2} \qquad \textbf{(12)}$$

As shown in Appendix B, the time $t_* = t_n$ also minimizes the area $A(t_*)$.

The argument for the sinusoidal trajectory is similar. The coordinates $x_p(t), y_p(t)$ of this trajectory are defined by the parametric equations (19). The derivative of the cumulative cross-track error expression for the sinusoid can be written as:

$$\frac{d}{dt_*}s_{cum} = 0$$

$$= \frac{d}{dt_*}\left[\int_{t_{n-1}}^{t_*} \begin{Bmatrix} \left(C_{x_{n,3}}\left[1 - sin\left(\frac{\pi}{t_{n+1}-t_{n-1}}\cdot(t-t_{n-1})\right)\right] sin\,\psi_{L_{n-1\_n}}\right) \\ -\left(C_{y_{n,3}}\left[1-2\cdot\frac{(t-t_{n-1})}{(t_{n+1}-t_{n-1})}\right] cos\,\psi_{L_{n-1\_n}}\right) \end{Bmatrix} dt \right.$$

$$\left. + \int_{t_*}^{t_n} \begin{Bmatrix} \left(C_{x_{n+1,3}}\left[1 - sin\left(\frac{\pi}{t_{n+1}-t_{n-1}}\cdot(t-t_{n-1})\right)\right] sin\,\psi_{L_{n\_n+1}}\right) \\ -\left(C_{y_{n+1,3}}\left[1-2\cdot\frac{(t-t_{n-1})}{(t_{n+1}-t_{n-1})}\right] cos\,\psi_{L_{n\_n+1}}\right) \end{Bmatrix} dt \right]$$

(13)

Taking the derivative of this expression with respect to t_*, applying the Fundamental Theorem of Calculus, and setting the resulting expression to zero gives:

$$\frac{d}{dt_*}s_{cum} = 0$$

$$= \begin{matrix} \left(C_{x_{n,3}}\left[1 - \sin\left(\frac{\pi}{t_{n+1}-t_{n-1}}\cdot(t_*-t_{n-1})\right)\right] \sin\psi_{L_{n-1_n}}\right) \\ -\left(C_{y_{n,3}}\left[1-2\cdot\frac{(t_*-t_{n-1})}{(t_{n+1}-t_{n-1})}\right] \cos\psi_{L_{n-1_n}}\right) \\ -\left(C_{x_{n+1,3}}\left[1 - \sin\left(\frac{\pi}{t_{n+1}-t_{n-1}}\cdot(t_*-t_{n-1})\right)\right] \sin\psi_{L_{n_n+1}}\right) \\ +\left(C_{y_{n+1,3}}\left[1-2\cdot\frac{(t_*-t_{n-1})}{(t_{n+1}-t_{n-1})}\right] \cos\psi_{L_{n\_n+1}}\right) \end{matrix}$$

(14)

This is true for $t_* = t_n$, which also minimizes the area $A(t_*)$ as shown in Appendix B.

APPENDIX B

ANALYTICAL SOLUTION OF OPTIMAL TIME FOR MOTION PRIMITIVES

Here, we derive the optimal time t_* that minimizes the area $A(t_*)$ of the bounded region enclosed by a trajectory $\boldsymbol{p}(t)$, defined as one of four motion primitives, and the corresponding leg path $\boldsymbol{q}(t)$, $t \in [t_{n-1}, t_{n+1}]$.

Primitive 2: Semicircle

The parametric equations for the coordinates of the semicircular trajectory are:

$$x_p(t) = A\sin\big(\omega(t - t_{n-1})\big) = C_{x_{n,3}} \cdot \sin\left(\frac{\pi}{(t_{n+1} - t_{n-1})} \cdot (t - t_{n-1})\right)$$

$$y_p(t) = A\,[1 + \cos(\omega(t - t_{n-1}) + \varphi)] = C_{y_{n,3}} \cdot \left[1 + \cos\left(\frac{\pi}{(t_{n+1} - t_{n-1})} \cdot (t - t_{n-1}) - \pi\right)\right]$$

(15)

The area $A(t_*)$ is the difference between the area bounded by the semicircle and the line between waypoints $n-1$ and $n+1$, which we denote by $A_{semicircle}$, and the area of the inscribed triangle with vertices at waypoints $n-1, n$, and $n+1$, denoted by $A_{triangle}$:

$$A(t_*) = A_{semicircle} - A_{triangle} = \frac{\pi r^2}{2} - \frac{1}{2}bh,$$

Using the parametric equations (15), this area can be written as:

$$A(t_*) = \frac{\pi r^2}{2} - \frac{1}{2}(2r)x_p(t_*) = \left[\frac{\pi r^2}{2} - rC_{x_{n,3}} \sin\left(\pi \cdot \frac{t_* - t_{n-1}}{t_{n+1} - t_{n-1}}\right)\right]$$

(16)

Taking the derivative of expression (16) with respect to t_* and setting the derivative to zero yields the value of t_* that minimizes $A(t_*)$, which determines the location $\boldsymbol{p}(t_n)$ of the interior waypoint along the semicircle trajectory $\boldsymbol{p}$:

$$\frac{dA}{dt_*} = 0 - r\pi \cdot \frac{1}{t_{n+1} - t_{n-1}} \cdot \cos\left(\pi \cdot \frac{t_* - t_{n-1}}{t_{n+1} - t_{n-1}}\right)$$

(17)

$$\Rightarrow\ t_* = t_n, \text{ since } \frac{t_n - t_{n-1}}{t_{n+1} - t_{n-1}} = \frac{1}{2}$$

(18)

Plugging this value for t_* into the equations (15) for the semicircle coordinates, we find that the interior waypoint should be placed at the point $(x_q(t_*) = C_{x_{n,3}}, y_q(t_*) = C_{y_{n,3}})$, which is the peak of the semicircle.

Primitive 3: Sinusoid

The parametric equations for the coordinates of the half-wave sinusoidal trajectory are:

$$x_p(t) = A\sin\big(\omega(t - t_{n-1})\big) = C_{x_{n,3}} \sin\left(\frac{\pi}{(t_{n+1} - t_{n-1})}(t - t_{n-1})\right)$$

$$y_p(t) = 2A\left[\frac{(t - t_{n-1})}{(t_{n+1} - t_{n-1})}\right] = 2C_{y_{n,3}} \cdot \left[\frac{(t - t_{n-1})}{(t_{n+1} - t_{n-1})}\right]$$

(19)

The area $A(t_*)$ is the difference between the area bounded by the half-wave sinusoid trajectory and the line between waypoints $n-1$ and $n+1$, denoted by $A_{sinusoid}$, and the area of the inscribed triangle, $A_{triangle}$:

$$A(t) = A_{sinusoid} - A_{triangle} = C_{x_{n,3}} \int_0^{\pi} \sin(t)\,dt - \frac{1}{2}bh$$

$$\Rightarrow A(t_*) = C_{x_{n,3}} \int_0^{\pi} \sin(t)\,dt - \frac{1}{2}\big(C_{x_{n+1,3}} - C_{x_{n-1,3}}\big)x_p(t_*)$$

$$\Rightarrow A(t_*) = C_{x_{n,3}}\left[2 - \frac{1}{2}\big(C_{x_{n+1,3}} - C_{x_{n-1,3}}\big)\sin\left(\pi \cdot \frac{t_* - t_{n-1}}{t_{n+1} - t_{n-1}}\right)\right]$$

(20)

where the expression for $x_p(t_*)$ is from the parametric equations (19). The derivative of $A(t_*)$ with respect to t_* is:

$$\frac{dA}{dt_*} = 0 - \frac{1}{2}\big(C_{x_{n+1,3}} - C_{x_{n-1,3}}\big) \cdot \frac{\pi C_{x_{n,3}}}{t_{n+1} - t_{n-1}} \cdot \cos\left(\pi \cdot \frac{t_* - t_{n-1}}{t_{n+1} - t_{n-1}}\right)$$

(21)

When set equal to zero, this derivative has the same solution for t_* as Eq. (17):

$$t_* = t_n$$

Plugging this value for t_* into the sinusoid coordinates (19), we find that the interior waypoint should be placed at $(x_q(t_*) = C_{x_{n,3}}, y_q(t_*) = C_{y_{n,3}})$, the peak of the sinusoid.

Primitive 4: Decaying Exponential

The parametric equations for the coordinates of the decaying exponential trajectory are:

$$x_p(t) = C_{x_{n-1,3}} \cdot e^{-y_p(t)}$$

$$y_p(t) = 2 \cdot C_{y_{n,3}} \cdot \left[\frac{t - t_{n-1}}{t_{n+1} - t_{n-1}}\right]$$

(22)

As illustrated in Figure 16, the area $A(t_*)$ can be written as the difference between the area $A_{triangle1}$ of the triangle (outlined in red) with vertices at the origin, waypoint $n-1$, and waypoint $n+1$, and the sum of the area $A_{exponential}$

under the exponential trajectory between times t_{n-1} and t_{n+1} and the area $A_{triangle2}$ of the triangle with vertices at the three waypoints $n-1$, n, and $n+1$:

$$A(t_*) = A_{triangle1} - (A_{exponential} + A_{triangle2})$$

(23)

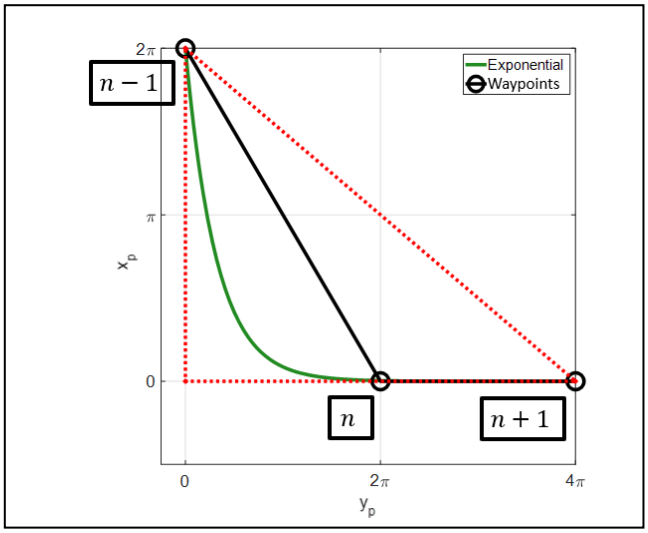

Figure 16: Bounded Area Associated with Decaying Exponential

We define $A_b \equiv A_{triangle1} - A_{exponential}$. Given the parametric equations (22), this area is computed as:

$$A_b = \frac{1}{2} C_{x_{n-1,3}} C_{y_{n+1,3}} - \int_{t_{n-1}}^{t_{n+1}} C_{x_{n-1,3}} \cdot e^{-y_p(t)}\, dt$$

(24)

Note that this area is not a function of t_*. The area $A_{triangle2}$ depends on the location of the interior waypoint n, and therefore is a function of t_*. This area is computed by multiplying the triangle's base, the distance between waypoints $n-1$ and $n+1$, by its height, the perpendicular distance between waypoint n and its base:

$$A_{triangle2}(t_*) = \left(\frac{\sqrt{\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)^2 + \left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)^2}}{2} \right) \cdot$$

$$\left(\frac{\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)\left(C_{y_{n+1,3}} - y_p(t_*)\right) - \left(C_{x_{n+1,3}} - x_p(t_*)\right)\left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)}{\sqrt{\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)^2 + \left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)^2}} \right)$$

(25)

By Eq. (23), taking the derivative of $A(t_*)$ with respect to t_* yields:

$$\frac{dA}{dt_*} = 0 + \frac{d}{dt_*}\left(\frac{\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)\left(C_{y_{n+1,3}} - y_p(t_*)\right)}{2} \right) - \frac{d}{dt_*}\left(\frac{\left(C_{x_{n+1,3}} - x_p(t_*)\right)\left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)}{2} \right)$$

(26)

When set equal to zero, the solution of this equation is the value of t_* that minimizes $A(t_*)$, which determines the location $\boldsymbol{p}(t_*)$ of the interior waypoint n along the exponential trajectory $\boldsymbol{p}$. Setting Eq. (26) equal to zero and substituting in the expressions for the trajectory coordinates (22), we obtain the following:

$$-\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)\left(\frac{d}{dt_*} y_p(t_*)\right) = -\left(\frac{d}{dt_*} x_p(t_*)\right)\left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)$$

$$\Rightarrow -\left(C_{x_{n+1,3}} - C_{x_{n-1,3}}\right)\left(2 \cdot C_{y_{n,3}} \cdot \left[\frac{1}{t_{n+1} - t_{n-1}}\right]\right) = 2 \cdot C_{y_{n,3}} \cdot \left[\frac{1}{t_{n+1} - t_{n-1}}\right] \cdot \left(C_{x_{n-1,3}} \cdot e^{-y_p(t_*)}\right)\left(C_{y_{n+1,3}} - C_{y_{n-1,3}}\right)$$

(27)

This equation can be solved for the x_p coordinate of the exponential at time t_*:

$$x_p(t_*) = C_{x_{n-1,3}} \cdot e^{-y_p(t_*)} = -\frac{C_{x_{n+1,3}} - C_{x_{n-1,3}}}{C_{y_{n+1,3}} - C_{y_{n-1,3}}}$$

(28)

The above equality was found from a graphical solution to be $t_* = 4.8s$ when $\left(C_{x_{n-1,3}} = 5, C_{y_{n,3}} = 2.5\right)$ which closely matches the solution computed using the optimization algorithm for the Bounded Area Minimization Problem, as described in Section III.

Primitive 5: Logarithmic Spiral

The parametric equations for the coordinates of the logarithmic spiral trajectory are given by:

$$y_p(t) = \alpha e^{bt} \sin(t)$$
$$x_p(t) = \alpha e^{bt} \cos(t)$$

(29)

As shown in Figure 17,the bounded area between the leg path and the spiral can be written as the difference between the area bounded by the spiral and the line between waypoints $n-1$ and $n+1$, which we denote by A_{spiral}, and the area of the inscribed triangle with vertices at waypoints $n-1, n$, and $n+1$, denoted by $A_{triangle}$.

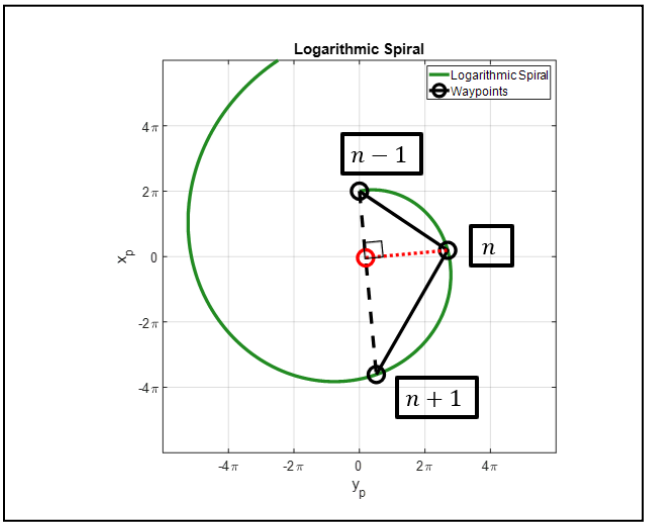

Figure 17: Bounded Area Associated with Logarithmic Spiral

The red circle represents the point on the line between the exterior waypoints $n-1$ and $n+1$ that is closest to the interior waypoint n. The equation of the line segment between waypoints $n-1$ and n+1 can be written in terms of the parameter t as follows:

$$y_q(t) = y_{n-1} + \frac{t - t_{n-1}}{t_{n+1} - t_{n-1}}(y_{n+1} - y_{n-1})$$

$$x_q(t) = x_{n-1} + \frac{t - t_{n-1}}{t_{n+1} - t_{n-1}}(x_{n+1} - x_{n-1})$$

(30)

Defining $y_p(t)$, $x_p(t)$ as in Eq. (30) and $y_q(t)$, $x_q(t)$ as in Eq. (31), the area A_{spiral} is obtained using Green's Theorem by integrating the curve along the spiral trajectory from $n-1$ to $n+1$ and back along the dotted line from $n+1$ to $n-1$:

$$A_{spiral} = \frac{1}{2}\left[\int_{t_{n-1}}^{t_{n+1}} (-y_p x'_p + x_p y'_p)dt + \int_{t_{n+1}}^{t_{n-1}} (-y_q x'_q + x_q y'_q)dt\right]$$

(31)

The area of the inscribed triangle, $A_{triangle}$, can be computed by defining the distance between $n-1$ and $n+1$ as the triangle's base, b, and the distance between the base and waypoint n (the length of the red dotted line in Figure 17) as the triangle's height, h. Since the location $\boldsymbol{p}(t_*)$ of waypoint n on the spiral determines this height, h and therefore $A_{triangle}$ can be written as functions of the decision variable t_*:

$$A_{triangle}(t_*) = \frac{bh(t_*)}{2}$$

(32)

The distance b is fixed and thus does not depend on t_*. The distance $h(t_*)$ can be computed based on the known equation for the distance between an arbitrary point and a line through two different points. The bounded area between the spiral and the leg path is then given by:

$$A(t_*) = A_{spiral} - A_{triangle}(t_*)$$

(33)

The minimum value of this area is obtained by taking the derivative of the expression $A(t_*)$ with respect to t_* and setting the resultant equal to zero. This operation yields:

$$\frac{d}{dt_*}A(t_*) = 0 \Rightarrow \frac{d}{dt_*}[y_p(t_*)] = \left[\frac{y_{n+1} - y_{n-1}}{x_{n+1} - x_{n-1}}\right] \cdot \frac{d}{dt_*}[x_p(t_*)]$$

(34)

Solving for t_* gives $1.69s$, which is also very close to the numerical solution of the optimization algorithm, as described in Section III. The constants for the parametric coordinates were $\alpha = 2\pi$ and b = 0.2.

ACKNOWLEDGMENTS

The authors would like to acknowledge the following parties for their influence in making this paper a reality.

- Don Caldwell for his excellent mentorship, wry sense of humor, practical experience flying helicopters, and in-depth knowledge designing and developing automatic flight control systems for them. But most importantly for always reminding us to ask “Who is John Galt?”
- Dino Cerchie for his vision recognizing and nurturing the MD530F platform as the ideal surrogate as an autonomy testbed decades before others caught on.
- Bill Pellerin for always reminding us that C-code won’t write itself, and that it’s 5-o’clock somewhere.
- Navid Dadkhah for his ability to manipulate splines and informal lessons on conversational Farsi.
- Graham Drozeski for reminding us that “good enough” is always the enemy of true progress.
- Denis Kosygin for his excellent introduction to the story of Queen Dido and derivations of the isoperimetric problem using Green’s Theorem in MAT 203 freshman year at Fine Hall, Princeton University.
- Jacob Walrath for reminding us all what is possible if one truly puts their heart and mind to a task.
- Chris Colosi for providing his perspectives from time spent in Philadelphia, PA, Mesa, AZ, Everett, WA, and abroad in the United Kingdom.
- Russell Enns for his introspective conversations on life, liberty, and the pursuit of happiness.
- Ram Janakiram for always asking the next question.

As always, there are many, many more people that contributed greatly to this paper than the above mentioned that may have been omitted so the authors wish to acknowledge all of them now.

REFERENCES

[1] S. M. Lavalle, Planning Algorithms, New York: Cambridge University Press, 2006.

[2] P. Petit, J. Wartmann, B. Fragniere and S. Greiser, "Waypoint based online trajectory generation and following control for the ACT/FHS," in *AIAA SciTech Forum*, San Diego, CA, 2019.

[3] J. Paduano, G. Drozeski, N. Dadkhah, S. Scherer, D. Cerchie, M. Hardesty, C. Cameron, J. Graham, B. Chu and et al, "TALOS: An Unmanned Cargo Delivery System for Rotorcraft Landing to Unprepared Sites," in *American Helicopter Society 71st Annual Forum*, Virginia Beach, VA, May 5, 2015.

[4] S. Choudhury, S. Arora and S. Scherer, "The Planner Ensemble and Trajectory Executive: A High Performance Motion Planning System with Guaranteed Safety," in *AHS 70th Annual Forum*, Montreal, 2014.

[5] Takahashi, Fujizawa and Lusardi, "Comparison of Autonomous Flight Control Performance Between Partial- and Full-Authority Helicopters," *Journal of Guidance, Control, and Dynamics*, vol. 45, no. 5, pp. 885-901, February 2022.

[6] Takahashi, Fujizawa, Lusardi and Whalley, "Autonomous Guidance and Flight Control on a Partial-Authority Black Hawk Helicopter," *Journal of Aerospace Information Systems*, vol. 18, no. 10, 2021.

[7] M. Whalley, M. D. Takahashi and H. Mansur, ""Flight Test Results for a New Mission-Adaptive Autonomy System on the RASCAL JUH-60A Black Hawk"," in *American Helicopter Society 72nd Annual Forum*, West Palm Beach, Florida, May 17-19, 2016.

[8] S. Scherer, L. Chamberlain and S. Singh, "Autonomous Landing at Unprepared Sites by a Full-Scale Helicopter," *Robotics and Autonomous Systems*, vol. 60, no. 12, pp. 1542-1562, 2012.

[9] J. Keller et al, "Coordinated Path Planning for Fixed-Wing UAS Conducting Persistent Surveillance Missions," in *IEEE Transactions on Automated Science and Engineering*, Vol. 14, No. 1, pp. 17 - 24, Jan 2017.

[10] F. Nawaz, P. Tekade and A. Ratnoo, "On Obstacle Avoidance Charactersitics of Proportional Navigation Guidance," in *AIAA SciTech Forum*, Orlando, FL, 2020.

[11] P. B. Sujit, S. Saripalli and J. B. Sousa, "Unmanned Aerial Vehicle Path Following: A Survey and Analysis of Algorithms for Fixed-Wing Unmanned Aerial Vehicles," in *IEEE Control Systems Magazine*, 2014.

[12] S. H. Lee, S. W. Hur, Y. Y. Kwak, Y. H. Nam and C. J. Kim, "Ahead-time Approach to Carrot-chasing Guidance Law for an Accurate Trajectory-

tracking Control," *International Journal of Control, Automation and Systems,* 2021.

[13] S. Park, J. Deyst and J. How, "A New Nonlinear Guidance Logic for Trajectory Tracking," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004.

[14] O. Halbe and M. Hajek, "Online Waypoint Trajectoy Generation Using State-Dependent Riccati Equation," *Journal of Guidance, Control, and Dynamics,* Vol. 42, No. 12, pp. 2687-2693 December 2019.

[15] D. Cerchie, D. Caldwell and B. Chu, "ULB Guidance, Navigation, and Control Laws and Evaluation of their Performance During Shipboard Operations," in *AHS International Specialist's Meeting on Unmanned Rotorcraft*, Mesa, AZ, 2012.

[16] A. Gubbels et al, "The NRC Bell 412 Advanced Systems Research Aircraft - Facility Description and Results of Initial In-Flight Evaluation," in *American Helicopter Society 58th Annual Forum*, Montreal, 2002.

[17] S. Hota and D. Ghose, "Optimal Transition Trajectory for Waypoint Following," in *IEEE International Conference on Control Applications*, Hyderabad, India, 2013.

[18] "ADS-33E-PRF, Aeronautical Design Standard Perforamance Specification Handling Qualities Requirements for Military Rotorcraft," US Army Aviation and Missile Command, Aviation Engineering Directorate, Redstone Arsenal, Alabama, 2000.

[19] T. Lyche and K. Morken, "A Data-Reduction Strategy for Splines with Applications to the Approximation of Functions and Data," *IMA Journal of Numerical Analysis,* vol. 8, pp. 185-208, 1988.

[20] T. Lyche and K. Morken, "Knot removal for parametric B-spline curves and surfaces," *Computer Aided Geometric Design,* vol. 4, pp. 217-230, 1987.