JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 31, No. 5, September–October 2008

Engineering Notes

ENGINEERING NOTES are short manuscripts describing new developments or important results of a preliminary nature. These Notes should not exceed 2500 words (where a figure or table counts as 200 words). Following informal review by the Editors, they may be published within a few months of the date of receipt. Style requirements are the same as for regular contributions (see inside back cover).

Trajectory-Shaping Guidance for Interception of Ballistic Missiles During the Boost Phase

John A. Lukacs IV*
Defense Technical Information Center, Ft. Belvoir, Virginia 22060-6218
and
Oleg A. Yakimenko†
Naval Postgraduate School, Monterey, California 93943-5107

DOI: 10.2514/1.32262

I. Introduction

FOR many years now, the U.S. Department of Defense has expended great effort to develop an integrated intercontinental ballistic missile (ICBM) defense system through a layered, defense in depth strategy. An ICBM's speed and altitude leave little room for error by the defender, and any strategy must include systems capable of defeating a ballistic missile (BM) at each of its three distinct phases (boost, midcourse, and terminal), which the Missile Defense Agency labels the engagement sequence groups (ESG) [1].

It is the portion of the effort directed toward the boost phase of the Ballistic Missile Defense Programs that is the focus of this Note. The boost-phase ESG is concerned with developing methods and technologies to conduct boost-phase intercept (BPI). Intercepting a missile in its boost phase is the ideal solution for a ballistic missile defense, because the missile is very vulnerable during this phase of its flight. The missile is relatively slow while struggling to overcome gravity, has a very visible exhaust plume, and cannot deploy countermeasures. Yet, the challenges that need to be overcome are immense: countering the large and changing acceleration rates, reliable scanning and tracking, and very short reaction time are among the most daunting. A variety of weapon systems are under development for conducting BPI, including airborne lasers, space-based intercept missiles, and ground-based intercept missiles. None of these systems is totally operational, though several look promising [1–3].

A missile's guidance law is one of the largest single factors affecting its ability to intercept a target. Yet, when discussing mission success in the BPI ESG, intercepting the target is only one factor; another major consideration is the ability to kill the target, using the available kinetic energy as effectively as possible. This suggests the need to control the geometry of the interception [4]. The discussion of the existing approaches as well as the generic models of ICBM and interceptor used in this study can be found in [5]. One of the findings of that paper is the constraint on the intercept altitude. As is well known, an endoatmospheric missile has a hard time maneuvering at high altitudes due to the low density of the surrounding atmosphere. Thus, the maximum allowable intercept value of 50–60 km is considered as the upper limit. The BM thrust model assuming the sharp drops at 130 and 240 s to represent the staging events suggests that 50–60-km altitude will be achieved between 130 and 140 s [5]. This is one of the most significant limitations on the BPI problem, because any station actively monitoring the launch area will still need 45–60 s to detect, track, analyze, and engage the target [3]. In the current study, simulations assume a 60-s delay in the interceptor launch, leaving about only 70–80 s for intercept per se.

This Note does not address the question of achieving operationability for various engagement scenarios, but rather offers an approach to rapidly compute intercept trajectories in case it is possible in principle. To this end, Sec. II develops and describes the essence of a new direct-method-based guidance law continuously calculated onboard the missile as a complete solution of a two-point boundary-value problem (TPBVP), and Sec. III presents some simulation results and discusses the feasibility of employing the proposed guidance law in the real-world conditions.

II. Essence of a New Guidance Algorithm

The guidance law suggested in this section determines the near-optimal flight path from the interceptor current position to a predicted BM position and then derives the set of control commands necessary to execute that flight. First, we formulate an optimization problem for the BM interception as follows.

A. Problem Formulation and Proposed Control Scheme

Among all admissible trajectories

$$\begin{gathered}\mathbf{z}(t) = [x_1(t), x_2(t), x_3(t), V(t), \gamma(t), \Psi(t)]^T \in S \\ S = \{\mathbf{z}(t) Z^6 \subset E^6\}, \qquad t \in [t_0, t_f]\end{gathered} \tag{1}$$

that satisfy

1) the system of differential equations (dynamic constraints)

$$\begin{aligned}\dot{x}_1 &= V\cos\gamma\cos\psi & \dot{V} &= g(n_x - \sin\gamma) \\ \dot{x}_2 &= V\cos\gamma\sin\psi & \dot{\gamma} &= gV^{-1}(n_z - \cos\gamma) \\ \dot{x}_3 &= -V\sin\gamma & \dot{\Psi} &= gn_y V^{-1}\cos^{-1}\gamma\end{aligned} \tag{2}$$

where x_1, x_2, and x_3 are the north–east–down coordinates of the interceptor's center of gravity; V, γ, and ψ are the interceptor's speed, path angle, and heading; n_x is the axial load factor; n_y and n_z (the other two components of the load factor) constitute the vector of controls $\mathbf{u} \in U^2 \subset E^2$ (we have no direct control over n_x),

Presented as Paper 6538 at the AIAA Guidance, Navigation, and Control Conference and Exhibit, Hilton Head, SC, 20–23 August 2007; received 26 May 2007; revision received 26 April 2008; accepted for publication 27 April 2008. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0731-5090/08 $10.00 in correspondence with the CCC.

*Lieutenant, U.S. Navy; john.lukacs.iv@gmail.com. Student Member AIAA.

†Research Associate Professor, Department of Mechanical and Astronautical Engineering, Code MAE/Yk; oayakime@nps.edu. Associate Fellow AIAA.

2) the (current) initial conditions

$$x_1(t_0) = x_{10}, \qquad x_2(t_0) = x_{20}, \qquad x_3(t_0) = x_{30}$$
$$V(t_0) = V_0, \qquad \gamma(t_0) = \gamma_0, \qquad \Psi(t_0) = \Psi_0 \tag{3}$$
$$n_y(t_0) = n_{y0}, \qquad n_z(t_0) = n_{z0}$$

that is,

$$x_i(t_0) = x_{i0}, \qquad \dot{x}_i(t_0) = \dot{x}_{i0}, \qquad \ddot{x}_i(t_0) = \ddot{x}_{i0}, \qquad i = 1, 2, 3 \tag{4}$$

3) the final conditions

$$x_1(t_f) = \hat{x}_{1f}^{\mathrm{BM}}, \qquad x_2(t_f) = \hat{x}_{2f}^{\mathrm{BM}}, \qquad x_3(t_f) = \hat{x}_{3f}^{\mathrm{BM}}$$
$$V(t_f) = V_f, \qquad \gamma(t_f) = \gamma_f, \qquad \Psi(t_f) = \Psi_f \tag{5}$$
$$n_y(t_f) = 0, \qquad n_z(t_f) = \cos\gamma_f$$

that is,

$$x_i(t_f) = x_{if}, \qquad \dot{x}_i(t_f) = \dot{x}_{if}, \qquad \ddot{x}_i(t_f) = \ddot{x}_{if}, \qquad i = 1, 2, 3 \tag{6}$$

based on the prediction of where the BM should be at time t_f,

$$\mathbf{x}_f^{\mathrm{BM}} = \hat{\mathbf{x}}_0^{\mathrm{BM}} + \hat{\mathbf{V}}_0^{\mathrm{BM}} \hat{t}_{\mathrm{go}} + \tfrac{1}{2}\hat{\mathbf{a}}_0^{\mathrm{BM}} \hat{t}_{\mathrm{go}}^2 \tag{7}$$

where $\hat{\mathbf{x}}_0^{\mathrm{BM}}$, $\hat{\mathbf{V}}_0^{\mathrm{BM}}$, and $\hat{\mathbf{a}}_0^{\mathrm{BM}}$ are the estimates of the BM's position, speed, and acceleration vectors at time $t = t_0$, and $\hat{t}_{\mathrm{go}} = t_f - t_0$ is an estimate of time-to-go,

4) the constraints, imposed on the intercept altitude

$$x_3(t_f) \le x_{3\max} \tag{8}$$

where $x_{3\max} = 60$ km,

5) controls

$$\xi(\mathbf{u}) = \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} n_{\max}(q) - \begin{bmatrix} |n_y(t)| \\ |n_z(t)| \end{bmatrix} \right\} \ge \mathbf{0} \tag{9}$$

which happen to be dependent on the dynamic pressure $q = 0.7pM^2$ (p is the static density determined by the altitude x_3 and Mach number M),

6) and the control derivatives

$$\eta(\mathbf{u}) = \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} \dot{n}_{\max} - \begin{bmatrix} |\dot{n}_y(t)| \\ |\dot{n}_z(t)| \end{bmatrix} \right\} \ge \mathbf{0} \tag{10}$$

find the optimal trajectory $\mathbf{z}^*(t)$ that minimizes some performance index (PI) J and the corresponding optimal control $\mathbf{u}^*(t)$.

The choice of PI for this specific application will be addressed later in this section, but first let us make three important comments. First, although the initial conditions for the interceptor are simply its current conditions and are therefore fixed, its final conditions are in fact variable. We can vary a position of the impact point by assigning different t_f [see Eq. (7)]. Also, the direction of the interceptor's velocity vector at the impact point is not defined (Ψ_f and γ_f are variable), and we have no direct control over the magnitude of the interceptor's final velocity V_f (because the thrust profile is fixed). We can only affect V_f indirectly by minimizing the interceptor's control actions and t_{go} to assure that V_f does not fall below a certain limit.

Second, as opposed to the standard formulation, the constraints on the control derivatives (10) are added to account for the actuator's dynamics (deflection of the control surfaces δ_y and δ_z). Being accounted for at the stage of reference trajectory generation implicitly assures more accurate tracking of this trajectory later on.

Finally, it should be noted that all four variables appearing on the right-hand side in Eq. (7) are not known for sure. The BM's current position, velocity, and acceleration, provided by the ground station or onboard alpha–beta–gamma filter, are only known with certain accuracy [4]. Moreover, for the boost phase, it is very difficult to rely on the estimate of acceleration, even if it is known with the sufficient accuracy (because of the staging events). Time-to-go is not known at all and is only assessed during optimization itself. Generally speaking, it means that there is no guarantee at the final point that the equality $\mathbf{x}(t_f) = \mathbf{x}^{\mathrm{BM}}(t_f)$ holds. However, when the interceptor approaches the BM, the estimates of the BM's position, velocity, and acceleration can be made more and more accurate, and time-to-go, determined at the previous iteration (solution of the TPBVP), becomes more precise too. Hence, to assure the high probability of kill [enforce the equality $\mathbf{x}(t_f) = \mathbf{x}^{\mathrm{BM}}(t_f)$], we would need to be able to solve the optimization problem during the intercept several times, as often as possible. The block diagram of the control system that accounts for such periodical updates is shown in Fig. 1. The inner loop represents the common feedback control for tracking the reference trajectory and the outer loop provides periodical updates of this reference trajectory.

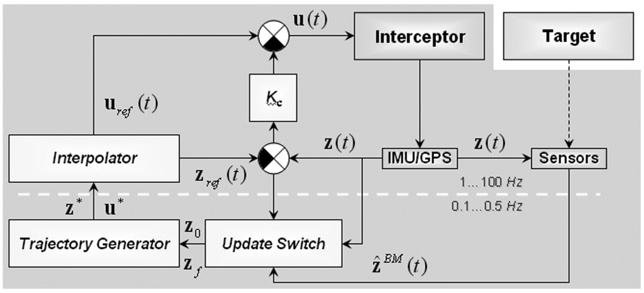

Fig. 1 Suggested double-loop feedback control.

As discussed in [5], neither of the indirect methods of calculus of variations can solve the preceding TPBVP with a varied right end in real time. Moreover, not many direct methods can handle such a problem either, especially in this situation, in which this trajectory has to be refined every other second or so to assure a continuity of the controls [meaning the requirement to satisfy up to the second-order derivative of the interceptor coordinates in the initial point for every cycle of the optimization as required by Eq. (4)] [6]. The direct method of calculus of variations presented earlier in [7], however, was specifically developed for similar problems and proved to be able to compute feasible trajectories in fractions of a second. The usage of this method assures the following:

1) The boundary conditions including high-order derivatives [Eqs. (4) and (6)] are satisfied a priori.

2) The control commands are smooth and physically realizable [Eqs. (9) and (10) hold].

3) The method is very robust and is not sensitive to small variations in the input parameters.

4) Only a few variable parameters are used, thus ensuring that the iterative process during optimization converges well and that the continuous update of the solution allows reliable path following even with no standard feedback ($K_c = 0$ in Fig. 1 can be zero).

Another important feature of this method is that it allows handling any complex PI. Therefore, we are not limited to simple standard PIs such as time, fuel expenditure, etc. (We will take advantage of this feature when introducing the compound PI later on.)

This direct method, developed more than a decade ago and then applied for a manned aircraft, has recently been employed in different real-time applications in academia and proved to be a reliable tool, easily tuned to a specific problem [8]. One of its latest applications also includes planning and conducting time-critical missions of multiple unmanned vehicles [9]. The differences between the preceding formulated problem with that of [7] is that the intercept problem implies fluid rather than soft final boundary conditions and that there is no authority over the thrust profile, leaving only two controls as opposed to three controls for the conventional aircraft.

B. Reference Trajectory

Applying the aforementioned direct method to the intercept problem leads to the following computational routine. It starts from assuming the intercept trajectory x_i $(i = 1, 2, 3)$ to be represented as a higher-order polynomial in some parameter τ, referred to as the virtual arc:

$$
\begin{aligned}
x_i'''(\tau) &= P_{x'''i}(\tau) = a_{i3} + a_{i4}\tau + a_{i5}\tau^2 + a_{i6}\tau^3 + a_{i7}\tau^4 \\
x_i''(\tau) &= P_{x''i}(\tau) = a_{i2} + a_{i3}\tau + \tfrac{1}{2}a_{i4}\tau^2 + \tfrac{1}{3}a_{i5}\tau^3 + \tfrac{1}{4}a_{i6}\tau^4 + \tfrac{1}{5}a_{i7}\tau^5 \\
x_i'(\tau) &= P_{x'i}(\tau) = a_{i1} + a_{i2}\tau + \tfrac{1}{2}a_{i3}\tau^2 + \tfrac{1}{6}a_{i4}\tau^3 + \tfrac{1}{12}a_{i5}\tau^4 + \tfrac{1}{20}a_{i6}\tau^5 + \tfrac{1}{30}a_{i7}\tau^6 \\
x_i(\tau) &= P_{xi}(\tau) = a_{i0} + a_{i1}\tau + \tfrac{1}{2}a_{i2}\tau^2 + \tfrac{1}{6}a_{i3}\tau^3 + \tfrac{1}{24}a_{i4}\tau^4 + \tfrac{1}{60}a_{i5}\tau^5 + \tfrac{1}{120}a_{i6}\tau^6 + \tfrac{1}{210}a_{i7}\tau^7
\end{aligned}
\tag{11}
$$

As shown later, the virtual arc τ [10] is a scaled arc-length parameter that serves as an independent variable for the polynomial functions (11) and allows decoupled optimization of the spatial trajectory and speed profile.

The coefficients a_{ik} ($i = 1, 2, 3$ and $k = 0, \ldots, 7$) can be determined by solving the following system of linear algebraic equations:

$$
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{6}\tau_f^3 & \frac{1}{24}\tau_f^4 & \frac{1}{60}\tau_f^5 & \frac{1}{120}\tau_f^6 & \frac{1}{210}\tau_f^7 \\
0 & 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{6}\tau_f^3 & \frac{1}{12}\tau_f^4 & \frac{1}{20}\tau_f^5 & \frac{1}{30}\tau_f^6 \\
0 & 0 & 1 & \tau_f & \frac{1}{2}\tau_f^2 & \frac{1}{3}\tau_f^3 & \frac{1}{4}\tau_f^4 & \frac{1}{5}\tau_f^5 \\
0 & 0 & 0 & 1 & \tau_f & \tau_f^2 & \tau_f^3 & \tau_f^4
\end{pmatrix}
\begin{pmatrix} a_{i0} \\ a_{i1} \\ a_{i2} \\ a_{i3} \\ a_{i4} \\ a_{i5} \\ a_{i6} \\ a_{i7} \end{pmatrix}
=
\begin{pmatrix} x_{i0} \\ x'_{i0} \\ x''_{i0} \\ x'''_{if} \\ x_{if} \\ x'_{if} \\ x''_{if} \\ x'''_{if} \end{pmatrix}
\tag{12}
$$

Equations (12) resolved for coefficients a_{ik} ($i = 1, 2, 3$ and $k = 0, \ldots, 7$) yield

$$
\begin{aligned}
&a_{i0} = x_{i0}, \quad a_{i1} = x'_{i0}, \quad a_{i2} = x''_{i0}, \quad a_{i3} = x'''_{i0} \\
&a_{i4} = \frac{-4x'''_{if} + 16x'''_{i0}}{\tau_f} + \frac{60x''_{if} - 120x''_{i0}}{\tau_f^2} + \frac{-360x'_{if} - 480x'_{i0}}{\tau_f^3} \\
&\quad + \frac{840x_{if} - 840x_{i0}}{\tau_f^4} \\
&a_{i5} = \frac{30x'''_{if} + 60x'''_{i0}}{\tau_f^2} + \frac{-420x''_{if} + 600x''_{i0}}{\tau_f^3} + \frac{2340x'_{if} - 2700x'_{i0}}{\tau_f^4} \\
&\quad + \frac{-5040x_{if} + 5040x_{i0}}{\tau_f^5} \\
&a_{i6} = \frac{-60x'''_{if} - 80x'''_{i0}}{\tau_f^3} + \frac{780x''_{if} - 900x''_{i0}}{\tau_f^4} + \frac{-4080x'_{if} - 4320x'_{i0}}{\tau_f^5} \\
&\quad + \frac{8400x_{if} - 8400x_{i0}}{\tau_f^6} \\
&a_{i7} = \frac{35x'''_{if} + 35x'''_{i0}}{\tau_f^4} + \frac{-420x''_{if} + 420x''_{i0}}{\tau_f^5} + \frac{2100x'_{if} + 2100x'_{i0}}{\tau_f^6} \\
&\quad + \frac{-4200x_{if} + 4200x_{i0}}{\tau_f^7}
\end{aligned}
\tag{13}
$$

and the final arc τ_f becomes the first optimization (varied) parameter.

It is worth noting that almost any intercept trajectory obtained as a result of the classical and modern guidance laws can be approximated with a lower-order polynomial, meaning that all of them are obviously a subset of the variety of trajectories that can be produced with the reference functions (11). Another point is that by having the Cartesian coordinates parameterized using the reference functions (11), we escape the necessity to integrate the kinematic (the first three) equations of system (2).

The reason for choosing an argument for the reference functions (11) to be some abstract parameter τ rather than time or path length is to allow optimizing (varying) the spatial trajectory without interfering with the speed profile [7,8], which is determined in our case by the axial load factor [given by the thrust profile $T(t)$]. A mapping between the virtual arc and time domains is defined using the virtual speed λ as

$$
\lambda(\tau) = \frac{d\tau}{dt} \tag{14}
$$

Next, we define the right-hand side vector in Eq. (12): the boundary conditions in the virtual arc domain. The initial conditions (4) are all determined by the current interceptor state (3) up to the second-order derivative in the time domain as follows:

$$
\dot{\mathbf{x}}_0 = \begin{bmatrix} \dot{x}_{10} \\ \dot{x}_{20} \\ \dot{x}_{30} \end{bmatrix} = \begin{bmatrix} V_0 \cos\gamma_0 \cos\psi_0 \\ V_0 \cos\gamma_0 \sin\psi_0 \\ -V_0 \sin\gamma_0 \end{bmatrix}
$$

$$
\ddot{\mathbf{x}}_0 = \begin{bmatrix} \dot{V}_0 \cos\gamma_0 \cos\Psi_0 - \dot{\gamma}_0 V_0 \sin\gamma_0 \cos\Psi_0 - \dot{\Psi}_0 V_0 \cos\gamma_0 \sin\Psi_0 \\ \dot{V}_0 \cos\gamma_0 \sin\Psi_0 - \dot{\gamma}_0 V_0 \sin\gamma_0 \sin\Psi_0 + \dot{\Psi}_0 V_0 \cos\gamma_0 \cos\Psi_0 \\ -\dot{V}_0 \sin\gamma_0 - \dot{\gamma}_0 V_0 \cos\gamma_0 \end{bmatrix}
\tag{15}
$$

In Eq. (15), the derivatives $\dot{V}_0$, $\dot{\gamma}_0$, and $\dot{\Psi}_0$ are taken from the dynamic (the last three) equations of system (2). For more flexibility, we will also vary the components of the third-order derivative in the initial point $\mathbf{x}_0$. To be more specific, its magnitude in the virtual domain, $|\mathbf{x}'''_0|$, and two angles defining its direction will be used as the second, third, and fourth varied parameters.

Regarding the final conditions (6), we can do the same, assuming γ_f and Ψ_f to be fifth and sixth optimization parameters and guessing on the value of V_f (to be adjusted during optimization):

$$
\dot{\mathbf{x}}_f = \begin{bmatrix} \dot{x}_{1f} \\ \dot{x}_{2f} \\ \dot{x}_{3f} \end{bmatrix} = \begin{bmatrix} V_f \cos\gamma_f \cos\psi_f \\ V_f \cos\gamma_f \sin\psi_f \\ -V_f \sin\gamma_f \end{bmatrix}
$$

$$
\ddot{\mathbf{x}}_f = \begin{bmatrix} \dot{V}_f \cos\gamma_f \cos\Psi_f - \dot{\gamma}_f V_f \sin\gamma_f \cos\Psi_f - \dot{\Psi}_f V_f \cos\gamma_f \sin\Psi_f \\ \dot{V}_f \cos\gamma_f \sin\Psi_f - \dot{\gamma}_f V_f \sin\gamma_f \sin\Psi_f + \dot{\Psi}_f V_f \cos\gamma_f \cos\Psi_f \\ -\dot{V}_f \sin\gamma_f - \dot{\gamma}_f V_f \cos\gamma_f \end{bmatrix}
\tag{16}
$$

To ensure a smooth flight path at the end of the trajectory and avoid any flair maneuver, we additionally assume

$$
\ddot{\mathbf{x}}_f = \mathbf{0} \tag{17}
$$

The usage of the virtual speed (14) allows the recalculation of the initial and final boundary conditions, transforming them from the time domain to the virtual arc domain using the obvious relations:

$$\dot{\mathbf{x}} = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\tau}\frac{\mathrm{d}\tau}{\mathrm{d}t} = \mathbf{x}'\lambda, \qquad \ddot{\mathbf{x}} = \frac{\mathrm{d}(\mathbf{x}'\lambda)}{\mathrm{d}\tau}\frac{\mathrm{d}\tau}{\mathrm{d}t} = \mathbf{x}''\lambda^2 + \mathbf{x}'\lambda'\lambda$$
$$\dddot{\mathbf{x}} = \frac{\mathrm{d}(\mathbf{x}''\lambda^2 + \mathbf{x}'\lambda'\lambda)}{\mathrm{d}\tau}\frac{\mathrm{d}\tau}{\mathrm{d}t} = \mathbf{x}'''\lambda^3 + 3\mathbf{x}''\lambda'\lambda^2 + \mathbf{x}'(\lambda''\lambda^2 + \lambda'^2\lambda) \tag{18}$$

When rearranged, theses relations define the virtual arc domain derivatives of the interceptor coordinates as

$$\mathbf{x}' = \dot{\mathbf{x}}\lambda^{-1}, \qquad \mathbf{x}'' = (\ddot{\mathbf{x}} - \dot{\mathbf{x}}\lambda')\lambda^{-2}$$
$$\mathbf{x}''' = (\dddot{\mathbf{x}} - 3\ddot{\mathbf{x}}\lambda' + \dot{\mathbf{x}}(2\lambda'^2 - \lambda''\lambda))\lambda^{-3} \tag{19}$$

Specifically, in the boundary points

$$\mathbf{x}'_0 = \dot{\mathbf{x}}_0\lambda_0^{-1}, \qquad \mathbf{x}''_0 = (\ddot{\mathbf{x}}_0 - \dot{\mathbf{x}}_0\lambda'_0)\lambda_0^{-2}, \qquad \mathbf{x}'_f = \dot{\mathbf{x}}_f\lambda_f^{-1}$$
$$\mathbf{x}''_f = (\ddot{\mathbf{x}}_f - \dot{\mathbf{x}}_f\lambda'_f)\lambda_f^{-2}$$
$$\mathbf{x}'''_f = (\mathbf{x}_f - 3\ddot{\mathbf{x}}_f\lambda'_f + \dot{\mathbf{x}}_f(2\lambda_f'^2 - \lambda''_f\lambda_f))\lambda_f^{-3} \tag{20}$$

The values of $\mathbf{x}_0$, $\dot{\mathbf{x}}_0$, $\ddot{\mathbf{x}}_0$, $\mathbf{x}_f$, $\dot{\mathbf{x}}_f$, $\ddot{\mathbf{x}}_f$, and $\dddot{\mathbf{x}}_f$ [Eqs. (15–17)] converted to $\mathbf{x}_0$, $\mathbf{x}'_0$, $\mathbf{x}''_0$, $\mathbf{x}_f$, $\mathbf{x}'_f$, $\mathbf{x}''_f$, and $\mathbf{x}'''_f$ using Eq. (20) are now ready to be used in Eq. (13) to compute the coefficients of the reference functions (11). Note that we need to do this conversion only for two boundary points. To this end, for these two points at the initial step, we may choose $\lambda_0 = V_0$, $\lambda_f = V_f$ [10], $\lambda'_0 = \dot{V}_0V_0^{-1}$, $\lambda'_f = \dot{V}_fV_f^{-1}$, and $\mathbf{x}'''_f = \mathbf{0}$. The fact that at the initial step (with no prior knowledge about the value of τ_f) we allow virtual and actual speed magnitudes at the terminal points to be the same implies that the virtual arc τ_f should be of the order of the actual path length s_f (just to start the optimization process). The latter can be roughly estimated based on the engagement geometry, thus allowing making a qualified initial guess on τ_f. Later on, we will correct the values of $\mathbf{x}'_0$, $\mathbf{x}''_0$, $\mathbf{x}'_f$, $\mathbf{x}''_f$, and $\mathbf{x}'''_f$ using Eq. (20) at each iteration based on the actual virtual speed profile obtained at the previous iteration. (It should be noted though that scaling the virtual speed profile $\lambda(\tau)$ does not really matter; the higher values of λ will simply result in the larger τ_f, leaving other parameters in the time domain unaltered.)

Equations (11–13), explaining how the boundary conditions can be met a priori, constitute the spirit of the proposed approach. Changing τ_f does not affect the boundary conditions (they are unconditionally satisfied with any τ_f by construction), but do change the shape of a candidate trajectory, allowing the time histories of the components of the state $\mathbf{z}(t)$ and control vector $\mathbf{u}(t)$ as well as t_f (t_{go}) to vary. Therefore, we have gained an opportunity to optimize the performance index without jeopardizing the boundary-value problem. Moreover, for the problem at hand, we additionally vary $\mathbf{x}'''_0$, γ_f, and Ψ_f, providing extra degrees of freedom.

To end this section, we use the first of Eqs. (19) one more time to convert the differential equation for the speed [the fourth one in Eq. (2)] into the t domain as

$$V'(\tau) = \dot{V}\lambda^{-1} = g(n_x - \sin\gamma)\lambda^{-1} \tag{21}$$

Because of the preset thrust profile, which cannot be altered, we will need to numerically integrate this equation to obtain a speed history. Luckily, this is the only equation to be integrated because the two remaining dynamic equations in system (2) are determined via inverse dynamics to be discussed next.

C. Discretization and Inverse Dynamics

The optimization routine starts with guessing the values of varied parameters, which are the virtual arc length τ_f, components of initial jerk $\mathbf{x}'''_0$, and final angles γ_f and Ψ_f. Then the numerical solution proceeds with computation of all parameters along the reference trajectory $P_{xi}(\tau)$ over a fixed set of N points (for instance, $N = 100$) equidistantly placed along the virtual arc $[0; \tau_f]$ with the interval of

$$\Delta\tau = \frac{\tau_f}{N-1} \tag{22}$$

We take the very next value for the virtual arc

$$\tau_j = \tau_{j-1} + \Delta\tau, \qquad j = 2, \ldots, N \tag{23}$$

and compute the corresponding parameters of the trajectory from $P_{xi}(\tau)$, $P_{x'i}(\tau)$, and $P_{x''i}(\tau)$ [Eq. (11)]). Having these, we proceed with defining the flight-path angle and heading as follows:

$$\gamma_j = \operatorname{arctg}\frac{-x'_{3;j}}{\sqrt{x'^2_{1;j} + x'^2_{2;j}}}, \qquad \Psi_j = \operatorname{arctg}\frac{x'_{2;j}}{x'_{1;j}} \tag{24}$$

Their derivatives can be defined via direct differentiation as

$$\gamma'_j = \frac{x'_{3;j}\left(x''_{1;j} + x''_{2;j}\right) - x''_{3;j}\left(x'^2_{1;j} + x'^2_{2;j}\right)}{\left(x'^2_{1;j} + x'^2_{2;j}\right)^{3/2}}\cos^2\gamma_j$$
$$\Psi'_j = \frac{x''_{2;j}x'_{1;j} - x''_{1;j}x'_{2;j}}{x''^2_{1;j}}\cos^2\Psi_j \tag{25}$$

Next, we numerically integrate Eq. (21) to obtain

$$V_j = V_{j-1} + V'_{j-1}\Delta\tau = V_{j-1} + g(n_{x;j-1} - \sin\gamma_{j-1})\lambda_{j-1}^{-1}\Delta\tau \tag{26}$$

(here, $n_{x;j-1}$ is defined by the thrust and drag models [5], and $\lambda_1 \triangleq \lambda_0$).

The next step is to compute the corresponding time interval, the elapsed time, and virtual speed λ:

$$\Delta t_{j-1} = 2\frac{\sqrt{\sum_{i=1}^{3}(x_{i;j} - x_{i;j-1})^2}}{V_j + V_{j-1}}$$
$$t_j = t_{j-1} + \Delta t_{j-1} \quad (t_1 \triangleq t_0), \qquad \lambda_j = \frac{\Delta\tau}{\Delta t_{j-1}} \tag{27}$$

Finally, the controls n_y and n_z are found by rearranging terms in the two last equations of Eq. (2):

$$n_{y;j} = V_jg^{-1}\lambda_j\Psi'_j\cos\gamma_j, \qquad n_{z;j} = V_jg^{-1}\lambda_j\gamma'_j + \cos\gamma_j \tag{28}$$

The time lag represented, for example, by a first-order system with the time constant T can also be easily inverted as well:

$$n^c_{y;j} = n_{y;j} + \frac{n_{y;j} - n_{y;j-1}}{\Delta t_{j-1}}T, \qquad n^c_{z;j} = n_{z;j} + \frac{n_{z;j} - n_{z;j-1}}{\Delta t_{j-1}}T \tag{29}$$

Now that we know the virtual speed profile, we can correct the terminal values of λ, λ', and λ'' to be used at the next iteration in Eq. (20) as follows:

$$\lambda_0 = \lambda_2, \qquad \lambda_f = \lambda_N, \qquad \lambda'_0 = (\lambda_3 - \lambda_2)\Delta\tau^{-1}$$
$$\lambda'_f = (\lambda_N - \lambda_{N-1})\Delta\tau^{-1}, \qquad \lambda''_f = (\lambda_N - 2\lambda_{N-1} + \lambda_{N-2})\Delta\tau^{-2} \tag{30}$$

(we use the simplest numerical differentiation formulas to estimate the first- and second-order derivatives).

D. Penalty Function and Performance Index

When all parameters (states and controls) are computed in each of N points, we can compute the values of PI and penalty function (PF). According to Eqs. (8–10), the PF can be formed as follows:

$$\Delta = [w^a, 1, 1, w^d, w^d]\begin{bmatrix} \max\limits_j(0; |x_{3;j}| - x_{3\max})^2 \\ \max\limits_j(0; |n_{y;j}| - n_{\max}(q_j))^2 \\ \max\limits_j(0; |n_{z;j}| - n_{\max}(q_j))^2 \\ \max\limits_j(0; |\dot{n}_{y;j}| - \dot{n}_{\max})^2 \\ \max\limits_j(0; |\dot{n}_{z;j}| - \dot{n}_{\max})^2 \end{bmatrix} \tag{31}$$

where the penalties on violation of constraints on intercept altitude, controls, and their derivatives are scaled with the weighting factors $w^a \sim 0.01$ and $w^d \sim 10$. (If the constraints on the control's derivatives (10) were accounted for via Eq. (29), then $w^d = 0$.)

Now let us address the issue of choosing the most appropriate PI. First of all, it is worth mentioning that one of the advantages of the proposed direct method is that it can handle any complex (compound) PI. For instance, in one of the earlier similar applications, the probability of target kill (in a stochastic simulation) was considered as a PI explicitly [11]. Here, we are dealing with the deterministic approach, and hence we should choose some physical parameter(s) that are somehow related to the probability to kill. One commonly used PI is miss distance. In our particular case, because the method allows for satisfying any terminal conditions a priori (i.e., assures a zero miss distance), we should look at some other parameter. Another choice of a PI could be a time of intercept t_{go} or physical path length s_f. We would also aim at maximizing an impact velocity or minimizing V_f^{-1}. The proposed direct method allows combining all three criteria into a single weighted compound PI:

$$J = w_1^J k_1 \tau_f + w_2^J k_2 t_{\mathrm{go}} + w_3^J k_3 V_f^{-1} \tag{32}$$

where k_i ($i = 1, 2, 3$) are the scaling factors, and w_i ($i = 1, 2, 3$ and $\|[w_1^J; w_2^J; w_3^J]\| = 1$) are the weighing coefficients to balance multiple objectives. Note that for the sake of computational simplicity, we use τ_f instead of s_f, as they are somewhat related to each other.

However, the preliminary analysis revealed that all three mentioned criteria are obviously related. The sooner the interceptor impacts the target and the shorter (less curved) path it flies, the larger impact velocity can be achieved. Moreover, because we have no direct control over the final interceptor's velocity, the last term in Eq. (32) can be excluded with absolutely no harm:

$$J = w_1^J k_1 \tau_f + w_2^J k_2 t_{\mathrm{go}} \tag{33}$$

Not only do the remaining terms in Eq. (32) assure the fastest intercept with the maximum impact velocity, but they also ensure that the numerical solution is actually physically realizable. Without these two parameters, the system might continue to optimize the intercept well beyond the capabilities of the missile or even physical reality. One example is that in a purely mathematical sense, there is no problem with a negative velocity (yet in the physical world that makes no sense) and the algorithm might try a program that would intercept after the missile velocity has gone past zero and into negative numbers (of course, the missile would stop flying long before it even reaches zero). One might be tempted to include a myriad of parameters to cover all possible eventualities; however, including these two parameters sufficiently accounts for nearly every physical limitation and eliminates the need for having a long list of cost variables.

In summary, the preceding guidance algorithm with the PI of the form of Eq. (33) assures a zero miss distance by construction of Eq. (11), avoids sharp maneuvers right before the impact by satisfying Eq. (17), takes into account changing constraints on the controls [with the dynamic pressure q as shown in Eq. (9)] by zeroing PF (31), indirectly maximizes V_f and minimizes τ_f and t_{go} via PF (33), and allows for controlling the three-dimensional shape of the intercept trajectory directly by varying impact angles γ_f and Ψ_f in Eq. (15). None of the classical guidance laws assures even one of the mentioned criteria against a maneuvering target, not to mention their combination.

Although the algorithm employing a PI in the form of Eq. (33) works perfectly well, the authors wanted to explore other objectives as well. To this end, it was suggested to add one more trajectory-shaping term, as shown next:

$$J = w_1^J k_1 \tau_f + w_2^J k_2 t_{\mathrm{go}} + w_3^J k_3 \frac{\left|\left(\mathbf{V}_f^{\mathrm{BM}}, \mathbf{V}_f\right)\right|}{\left\|\mathbf{V}_f^{\mathrm{BM}}\right\| \|\mathbf{V}_f\|} \tag{34}$$

The third term was chosen to maximize the angle of impact upon interception. This reflects the need to go beyond simply hitting the target and to instead maximize the kinetic energy to ensure that the interceptor disables the target. Therefore, the authors propose to pursue three objectives, which are to minimize the dimensions of the intercept maneuver (the first term), to minimize time-to-go (the second term), and to maximize the impact angle of the interception (i.e., hit the BM at 90-deg angle) (the third term).

In Eq. (34), using the factors k_1 and k_2 allows for scaling each term appropriately, so that they all are roughly equal to each other when optimized (e.g., the anticipated intercept time is counted in tens of seconds, whereas the cosine of the impact angle will vary from zero to one). Failure to weight them properly will obviously skew the PI. Additionally, through the weighting functions w_1^J, w_2^J, and w_3^J, a tradeoff analysis can be conducted and variables can be included or excluded as desired [for instance, making $w_3^J = 0$ brings PI back to the form of Eq. (33)].

The computational routine of the proposed algorithm looks as follows. With the estimates $\hat{\mathbf{x}}_0^{\mathrm{BM}}$, $\hat{\mathbf{V}}_0^{\mathrm{BM}}$, $\hat{\mathbf{a}}_0^{\mathrm{BM}}$, and $\hat{t}_{\mathrm{go}}$ and guesses of the values of varied parameters τ_f, $\mathbf{x}'''_0$, γ_f, and Ψ_f, we determine the initial and final boundary conditions [Eqs. (15–17)] and compute the candidate trajectory [Eqs. (11–13)]. Next, with a fixed thrust profile, we inverse the missile dynamics and determine the remaining states (24) and controls (28). Then we estimate PF (31) and PI (34). Now we can refine the estimate of $\hat{t}_{\mathrm{go}}$ and try to adjust some of the variable parameters to drive the PF to zero (within certain tolerance) and minimize the PI.

All of the computations were performed in the MATLAB/Simulink development environment employing the `fminsearch` or `fmincon` functions to carry out the optimization by varying six optimization parameters and evaluating the PI and PF together (`fminsearch`) or separately (`fmincon`). Once the minimum value of the PI has been found, the developed algorithm returns the required control time history to the missile guidance system, which can then execute the commands and fly the derived flight path. Because the missile system can be programmed with sufficient data to compensate for its control system time constant [Eq. (29)], the system lag can be effectively negated, thus eliminating a source of error. Updating the guidance system every several seconds (Fig. 1) results in an increasingly accurate final intercept position and further ensures the mission kill.

Ultimately, the authors would like to run a Monte Carlo simulation considering various engagement scenarios to determine a launch region (conditions) in which intercept does occur (is possible). In this case, divergence of the iteration process will indicate that the interceptor launched from a specific region with respect to the target cannot disable (reach) it. To this end, both functions `fminsearch` and `fmincon` have a limit on the maximum number of iterations (maximum number of function evaluations) allowed. Specifically, by default, `fminsearch` allows 200ω iterations, in which ω is the number of variable parameters, and `fmincon` is limited to 400 iterations. In this study, we limited the number of iterations to 100 and for several considered engagement scenarios, in which interception was possible, never reached this limit, meaning that solution always converged and converged very fast. The following discusses the results of one of such successful intercept simulations.

III. Simulation Results

This section presents the result of simulation of optimizing the intercept trajectory as explained in the previous section. During the simulation, the BM path data are called by the interceptor to mimic the missile's onboard sensors. The interceptor's launch point was chosen to be roughly 150 nm from the BM launch position. The interceptor is launched with a 60-s delay after the BM. The first portion of the interceptor flight is the vertical launch, which is assumed to last 6 s, followed by a 4-s period in which the missile arcs over toward the target. At 10 s, the state of the interceptor and the state of the BM are "seen" and input into the system under some assumed ground control station. At this point, the missile would not have an independent radar picture of the target, so instead would need to be fed targeting data, as is done by many surface-to-air missile systems. This can continue indefinitely until the interceptor has its own radar fix on the ballistic missile. Then the interceptor is able to estimate the location, velocity, and acceleration of the BM at the appropriate intervals on its own (in the current simulation, by simply coordinating the launch time of the interceptor with the launch time of the BM). At each control update cycle (according to the proposed control scheme of Fig. 1, every 2–10 s), the TPBVP is solved.

Figure 2 represents an example of a successful intercept. The dotted line depicts an unguided ascent of the BM, followed by the solid line representing a guided portion of the flight. A triangle on the BM trajectory designates this switching point: that is, the point when interceptor activates its guidance law (the point from which the state of the BM was extracted). The intercept trajectory on this figure actually consists of six pieces (i.e., six consecutive 10-s updates). Once the trajectory generator (Fig. 1) computes the intercept trajectory for the first time (when the estimates of the BM parameters become available) and the controller starts to track it, all subsequent updates of the intercept trajectory are based on the very good knowledge about the parameters defining this intercept trajectory, and therefore the runtime for the following optimization cycles is only a fraction of that needed for the first time. Therefore, the following only addresses solving the optimization problem at the very first instance, when the missile's computer has to find the intercept trajectory for the first time.

A first guess at time-to-go and flight distance is done by a simple iterative process that takes the known velocity profile of the interceptor and iterates an initial intercept time using a first-order approximation of the ballistic path [4] (i.e., the known velocity profile of the target [5]). Also, the program creates initial guesses for the final values of the flight-path angle ($\gamma_f = 0.5\pi + \hat{\gamma}_f^{\mathrm{BM}}$) and heading ($\Psi_f = \pi + \hat{\Psi}_f^{\mathrm{BM}}$). This serves only as an initial start point and must only be mildly accurate, because the program will optimize the time and flight path as it operates. The accuracy of the initial guesses only serves to decrease the necessary computation time.

As expected for initial optimization, about 20 different candidate paths were tested before arriving in the neighborhood of the final value, and then about 40 additional different flight paths were looked over before arriving at the final solution (Fig. 3). As seen in Fig. 3, the PF was indeed influencing the choice of trajectory. If the required control effort exceeded the maximum capabilities of the missile, the candidate intercept trajectory was rejected as infeasible. As seen from Fig. 3a at the 18th iteration, one of two PF components has reached its minimums already, but because the second component is not zero, the optimization goes on. At the 28th iteration, another component reaches zero but the second constraint is still violated, and so the optimization procedure continues. The final path has no penalty assigned to it.

Figure 4 proves the feasibility of the obtained solution in another way by explicitly showing that the required control inputs n_y^c and n_z^c are within the limitations of the system. The solid lines represent the maximum capability of the system at each instant of time based on the dynamic pressure of the missile, as prescribed by Eq. (9). (This snapshot only shows the capabilities for the chosen flight path, yet while the algorithm is running, those capabilities are individually recalculated each time and vary widely based on the specifics of the flight-path geometries being tested.) The constraints on the control derivatives (10) are also satisfied. Note that the optimal intercept trajectory uses the full capabilities of the missile, as it should. Also note that the required forces are near zero at the intercept point, demonstrating that no flair maneuver or high-gravity maneuver was conducted [which complies with Eq. (17)]. This suggests that the full kinematic energy of the missile is directed into the target, which was the original intent of the algorithm. To this end, Fig. 5 shows the impact angle versus iteration. Indeed, the impact angle goes to 90 deg, making its contribution to the PI (34) negligible.

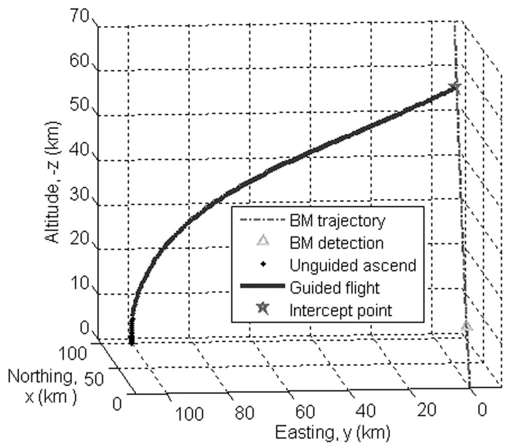

Fig. 2 Intercept trajectory.

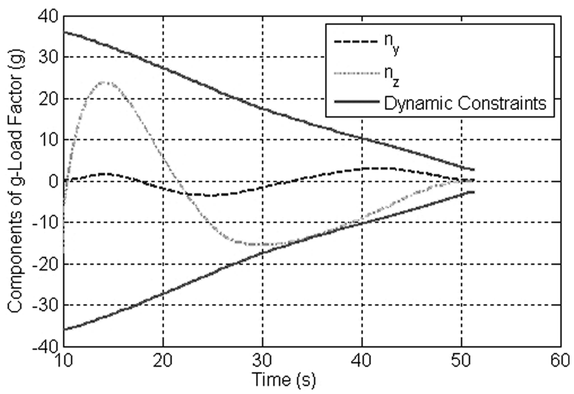

Fig. 4 Time histories of controls.

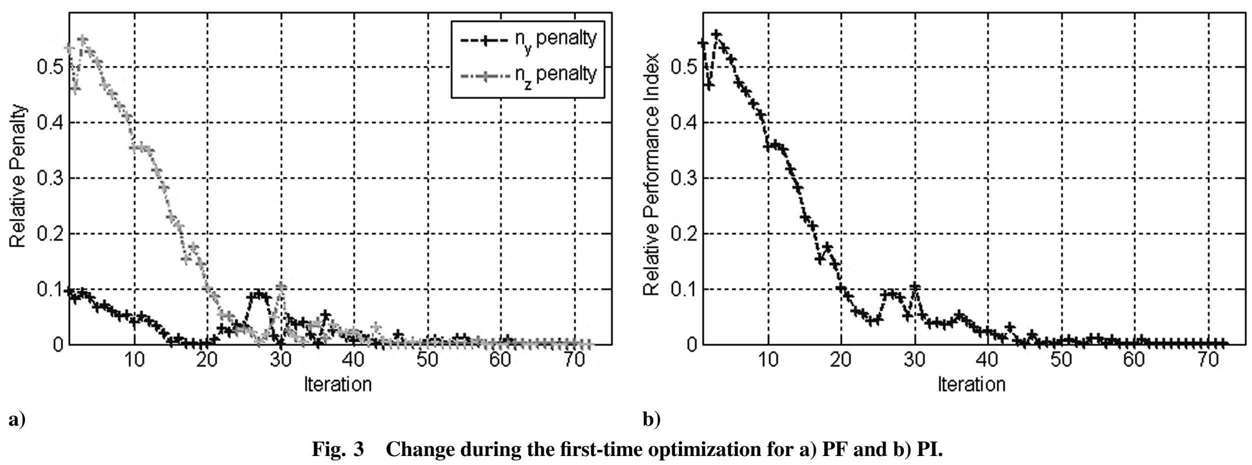

Fig. 3 Change during the first-time optimization for a) PF and b) PI.

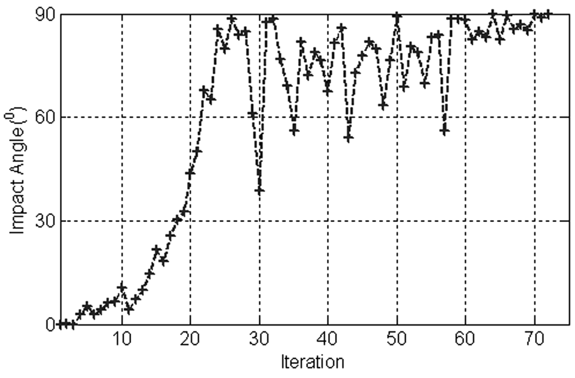

Fig. 5 Iterative values of the impact angle.

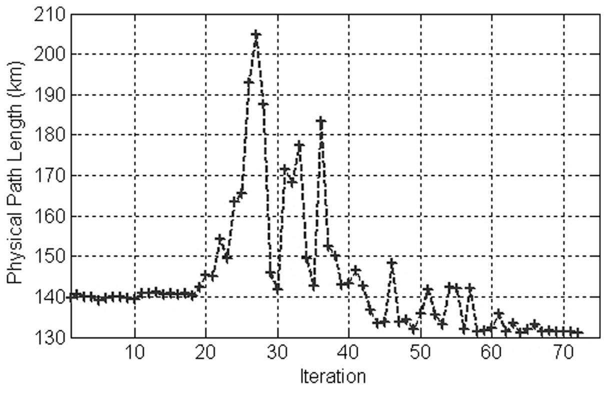

Fig. 7 Iterative values of the physical path length s_f.

Figure 6 shows the values of the two remaining terms in PI (34). Although neither t_{go} nor τ_f have a target value; it is seen that the algorithm has tried numerous values to converge to the best ones. It should be noted that the computed time-to-intercept t_{go} corresponds to the point on the ballistic flight path, and this is the major difference between the proposed algorithm and most predictive intercept guidance laws in which t_{go} is only an educated guess; the proposed algorithm has actually tested the proposed flight path and determined the precise t_{go}.

Another point to make is that the length of the virtual arc τ_f (Fig. 6a) is not proportional to the length of the physical length s_f, shown in Fig. 7. That is not only because of the varied speed profile along the trajectory, but also because other varied parameters ($\mathbf{x}'''_0$, γ_f, and Ψ_f) affect the overall shape and therefore the physical length of the trajectory too.

A chattering observed in Figs. 5 and 6 is not a bad thing, as it might be thought of, and is caused by three reasons. First, the final conditions are changing at every iteration because of the updated value of time-to-go [see Eq. (7)]. Second, as mentioned, the gradient-free variable-step method is used for optimization. Therefore, when the candidate solution violates the constraints, the algorithm performs a back step and then proceeds with the smaller step (in the same direction). Although this method might be considered to be not robust, in fact, compared with the gradient methods, it is more robust, finding the solution when it exists reliably and fast enough (even with a smaller number of performance index evaluations), not to mention that it might be quite difficult to assure smooth gradients for the aerodynamic lookup table data. Third, as also mentioned already, the first two objectives of PI (34) are conflicting with the third one, because, for example, minimization of the virtual arc is not necessarily consistent with achievement of the 90-deg impact angle objective (the latter tends to maximize arc length, particularly if the initial conditions are not “friendly” to such a crossing angle). Blending three objectives together using the weighting coefficients apparently causes a certain flatness of the performance index near the optimal solution. And, finally, it should be noted that a chattering occurs near the Pareto frontier (the set of solutions in which no further improvements can be made in all objectives simultaneously), meaning that we are in the vicinity of the solution already; the PI itself (Fig. 3b) does not demonstrate large chattering compared with those of Figs. 5 and 6. As seen from Figs. 5 and 6, we are almost done after 31 iterations and the remaining iterations are only needed to tune the final solution to meet specifications on the tolerance of varied parameters.

IV. Conclusions

To address the needs for the advanced guidance law overcoming fatal characteristics of modern guidance in a specific application, a new guidance algorithm has been developed and tested in simulations. This algorithm relies on the direct method of calculus of variations developed earlier and has four key elements: the optimal solution is searched for within the limited variations space (namely, high-order polynomials); the algorithm limits it even further by forcing unconditional satisfaction of the higher-order derivatives at endpoints and leaving only a few (six) varied parameters; it employs inverse dynamics, resulting in avoiding integration of all but one dynamic equations and therefore drastically speeds up the optimization routine; and all computations are carried out in the virtual domain, allowing space and time decoupling. The trajectory optimization problem is then converted into a nonlinear programming problem and solved numerically. The developed algorithm is extremely robust and requires only a moderate level of computational power, even in the considered application when the final boundary conditions are not fixed. It can therefore be easily incorporated into existing missile guidance systems. The future extension of this study will include addressing the following issues: performing a tradeoff analysis of PI’s weighting coefficients (to determine if the chosen terms of the compound PI are the most influential choices, if they are still redundant, or if other choices may provide better results); checking how inaccuracies in the estimation

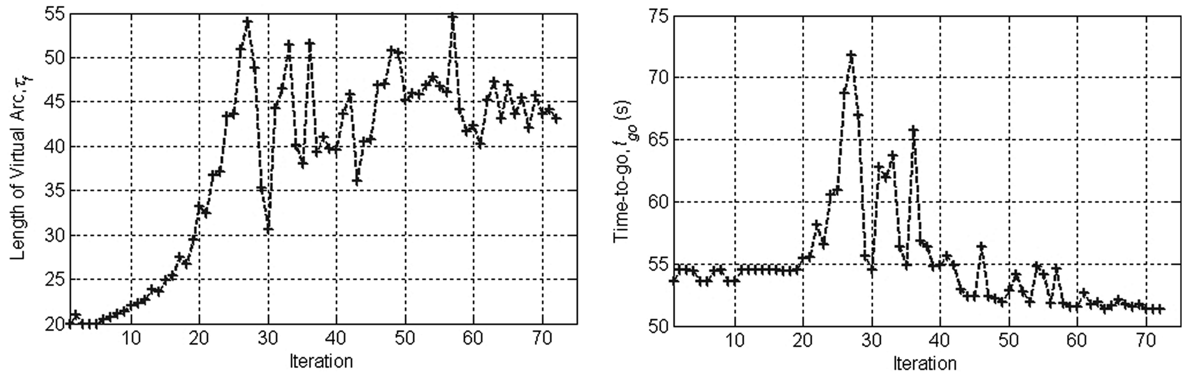

Fig. 6 Iterative values of the a) final arc τ_f and b) intercept time t_{go}.

of the BM parameters affect the probability of the interception; perform Monte Carlo simulation on launch-interception scenarios to determine possible limitations of the algorithm and reliable interception zones; and performing robustness analysis using the higher-fidelity models as well as autopilot models in the presence of disturbances.

References

[1] "Missile Defense—Worldwide," Missile Defense Agency, http://www.mda.mil/mdalink/pdf/bmdsbook.pdf [retrieved 26 June 2008].

[2] Cornell, C., "Shielding America: A National Missile Defense System," http://www.its.caltech.edu/~sciwrite/journal03/A-L2/Cornell.html [retrieved 26 June 2008].

[3] Barton, D. K., Falcone, R., Kleppner, D., Lamb, F. K., Lau, M. K., Lynch, H. L., Moncton, D., Montague, D., Mosher, D. E., Priedhorsky, W., Tigner, M., and Vaughan, D. R., "Report of the American Physical Society Study Group on Boost-Phase Intercept Systems for National Missile Defense: Scientific and Technical Issues," *Reviews of Modern Physics*, Vol. 76, No. S1, 2004.

[4] Zarchan, P., *Tactical and Strategic Missile Guidance*, 5th ed., Progress in Astronautics and Aeronautics, Vol. 219, AIAA, Reston, VA, 2007.

[5] Lukacs, J. A. IV., and Yakimenko, O. A., "Trajectory-Shape-Varying Missile Guidance for Interception of Ballistic Missiles During the Boost Phase," AIAA Guidance, Navigation and Control Conference and Exhibit, Hilton Head, SC, AIAA Paper 2007-6538, Aug. 2007.

[6] Yakimenko, O. A., Xu, Y., and Basset, G., "Computing Short-Time Aircraft Maneuvers Using Direct Methods," AIAA Guidance, Navigation, and Control Conference and Exhibit, Honolulu, AIAA Paper 2008-6632, Aug. 2008.

[7] Yakimenko, O. A., "Direct Methods for Rapid Prototyping of Near-Optimal Aircraft Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 5, 2000, pp. 865–875.

[8] Yakimenko, O., "Direct Method for Real-Time Prototyping of Optimal Control," Control 2006 Conference, United Kingdom Automatic Control Council Paper 190, 2006.

[9] Kaminer, I., Yakimenko, O., Dobrokhodov, V., Pascoal, A., Hovakimyan, N., Patel, V., Cao, C., and Young, A., "Coordinated Path Following for Time-Critical Missions of Multiple UAVs via L_1 Adaptive Output Feedback Controllers," AIAA Guidance, Navigation, and Control Conference and Exhibit, Hilton Head, SC, AIAA Paper 2007-6409, Aug. 2007.

[10] Taranenko, V. T., *Experience of Employing Ritz's, Poincaré's, and Lyapunov's Methods for Solving Flight Dynamics Problems*, Air Force Engineering Academy, Moscow, 1968 (in Russian).

[11] Dobrokhodov, V. N., and Yakimenko, O. A., "Synthesis of Trajectorial Control Algorithms at the Stage of Rendezvous of an Airplane with a Maneuvering Object," *Journal of Computer and Systems Sciences International*, Vol. 38, No. 2, 1999, pp. 262–277.