2013

# Bio-inspired, Varying Manifold Based Method With Enhanced Initial Guess Strategies For Single Vehicle's Optimal Trajectory Planning

Ni Li
*University of Central Florida*

# BIO-INSPIRED, VARYING MANIFOLD BASED METHOD WITH ENHANCED INITIAL GUESS STRATEGIES FOR SINGLE VEHICLE'S OPTIMAL TRAJECTORY PLANNING

by

NI LI
B.S. Northwestern Polytechnical University, 2008
M.S. University of Central Florida, 2011

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Mechanical & Aerospace Engineering
in the College of Engineering & Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2013

Major Professor: Yunjun Xu

# ABSTRACT

Trajectory planning is important in many applications involving unmanned aerial vehicles, underwater vehicles, spacecraft, and industrial manipulators. It is still a challenging task to rapidly find an optimal trajectory while taking into account dynamic and environmental constraints. In this dissertation, a unified, varying manifold based optimal trajectory planning method inspired by several predator-prey relationships is investigated to tackle this challenging problem.

Biological species, such as hoverflies, ants, and bats, have developed many efficient hunting strategies. It is hypothesized that these types of predators only move along paths in a carefully selected manifold based on the prey's motion in some of their hunting activities. Inspired by these studies, the predator-prey relationships are organized into a unified form and incorporated into the trajectory optimization formulation, which can reduce the computational cost in solving nonlinear constrained optimal trajectory planning problems. Specifically, three motion strategies are studied in this dissertation: motion camouflage, constant absolute target direction, and local pursuit.

Necessary conditions based on the speed and obstacle avoidance constraints are derived. Strategies to tune initial guesses are proposed based on these necessary conditions to enhance the convergence rate and reduce the computational cost of the motion camouflage inspired strategy.

The following simulations have been conducted to show the advantages of the proposed methods: a supersonic aircraft minimum-time-to-climb problem, a ground robot obstacle avoidance problem, and a micro air vehicle minimum time trajectory problem. The results show that the proposed methods can find the optimal solution with higher success rate and faster

convergent speed as compared with some other popular methods. Among these three motion strategies, the method based on the local pursuit strategy has a relatively higher success rate when compared to the other two.

In addition, the optimal trajectory planning method is embedded into a receding horizon framework with unknown parameters updated in each planning horizon using an Extended Kalman Filter.

# ACKNOWLEDGMENTS

First of all, I would like to give sincere appreciation to my advisor Dr. Yunjun Xu for his immense academic insight, patient attitude, and continuous support of my study and research. I could not have finished my dissertation without his knowledge and guidance.

Besides my advisor, I would like to thank the rest of my committee members: Dr. Kuo-chi Lin, Dr. Yuanli Bai, and Dr. Aman Behal for their corrections and help on my dissertation. I would also like to give special thanks to Prof. Suhada Jayasuriya for his insightful comments during my candidacy exam and preparation of my dissertation, and Dr. Khanh D. Pham for his help on my conference and journal papers.

I also want to thank my former and current lab-mates: Dr. Gareth Basset, Robert Sivilli, Charles Remeikas, Jacob Belli, Brad Sease, Kenneth Thompson, etc. for their suggestions on my research and English writing. My thanks also go to my closest friends Dan Chen, Jinling Liu, Xueping Yang, Yaohan Chen, Pingting Huang, and Yan Shan for their encouragement.

I would like to give special thanks to He Shen, for his continuous love, support and encouragement over the past five years.

Finally, I dedicate this dissertation to my dear family, especially to my parents, for their immense love and encouragement throughout my life.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| BC | Boundary Conditions |
| BiCF | Bio-inspired Computational Framework |
| CATD | Constant Absolute Target Direction |
| CoV | Calculus of Variations |
| DC | Direct Collocation |
| DoF | Degree of Freedom |
| E. C. | Equality Constraint |
| I. E. C. | Inequality Constraint |
| KKT | Karush-Kuhn-Tucker |
| LGL | Legendre-Gauss-Lobatto |
| LP | Local Pursuit |
| MAV | Micro Air Vehicle |
| MC | Motion Camouflage |
| NLP | Nonlinear Programming |
| PCP | Path Control Parameter |
| PMP | Pontryagin's Minimum Principle |
| PS | Pseudospectral |
| RHS | Right-hand Side |
| SCP | Speed Control Parameter |
| UAV | Unmanned Air Vehicle |
| ULA | Unified Linear Algebraic |

# CHAPTER ONE: INTRODUCTION

## Background of Trajectory Planning Problems

Trajectory planning plays a very important role in many applications including spacecraft [1]-[2], unmanned/micro aerial vehicles [3]-[6], ground robots [8]-[7], autonomous underwater vehicles [9], robotic manipulators [10]-[12], and image-guided neurosurgery [13]. A challenge with trajectory planning involves rapidly finding an optimal trajectory, while taking dynamic and environmental constraints into consideration [14]. To date, many methods have been developed to tackle this challenging problem.

One class of methods [15]-[33] seeks global optimal solutions for nonlinear systems. In order to find the best solution among multiple local optima, a global search effort is required [29]. The current global search methods include the grid search method [32], successive approximation method [33], and heuristic or meta-heuristic methods [15]-[23]. These methods may not be efficient for many realistic complex trajectory optimization problems [29] due to the following two limitations: (i) they cannot always find converged solutions, and (ii) it is difficult to provide rigorous optimality proofs [29].

Another class of methods [34]-[59] is used to find local optima by either: (i) indirect methods [36]-[44] or (ii) direct methods [45]-[59]. Indirect methods solve for local optimal solutions based on the Pontryagin's Minimum Principle (PMP) [35]. These methods often cannot obtain a converged solution when there are severe inequality constraints (I.E.C.s). In addition, indirect methods are extremely sensitive to the initial guess of the co-state, which is hard to estimate [34], [43], [44]. Direct methods convert the nonlinear constrained trajectory

optimization problems to nonlinear programming (NLP) problems [34] by using different discretization methods. Examples of discretization methods include the Hermit-Simpson [57], Trapezoid [59], Runge-Kutta [58], and Pseudospectral approaches [49]-[50], [52]-[54]. Direct methods can easily incorporate equality constraints and inequality constraints, but the dimension of the achieved NLP problems is typically large which results in a high computational cost [14]. In [60], a method inspired by the motion camouflage (MC) phenomenon is investigated, in which the trajectory of a vehicle is optimized in a carefully selected or iteratively refined manifold defined by a virtual prey path and a reference point.

## Contribution of Dissertation

Inspired by biological phenomena observed in hoverflies, ants and bats, a Unified Linear Algebraic (ULA) equation is derived to capture three motion strategies: Motion Camouflage (MC), Local Pursuit (LP), and Constant Absolute Target Direction (CATD). The ULA equation is then used to formulate a varying subspace (or manifold) for a bio-inspired computational framework (BiCF), in which nonlinear constrained trajectory optimization problems can be rapidly solved.

As with other direct methods, the proposed bio-inspired, varying manifold trajectory optimization method also transforms the infinite dimension problem to a NLP problem with finite dimension. The dimension of this achieved NLP problem is smaller than those achieved by the other traditional direct methods, which significantly improves the optimal solution convergence rate and calculation speed.

Strategies to enhance the initial guess of the optimizable variables are derived based on

the necessary conditions coming from the obstacle avoidance and speed constraints. These guidelines can enhance the convergent rate and reduce the computational time of the achieved NLP problem.

The proposed trajectory optimization methods can generate an optimal trajectory rapidly. These methods are applied to several applications in this dissertation including a Micro Air Vehicle (MAV) with unknown aerodynamic parameters, a ground two-wheel drive robot and a supersonic aircraft. The simulations will show the capabilities of the proposed trajectory optimization method.

## Dissertation Outline

The organization of this dissertation is shown as follows. Chapter 2 will introduce preliminary knowledge including the definitions of trajectory optimization and nonlinear programming, and the motion strategies observed in three species. Chapter 3 will introduce the bio-inspired, varying manifold trajectory optimization method. Two non-trivial examples, a robot collision avoidance problem and a hypersonic aircraft minimum-time-to-climb trajectory planning problem will be conducted to show the capability of the proposed trajectory optimization method. The initial guess techniques for bio-inspired trajectory method based on the MC strategy will be discussed in Chapter 4. A Monte Carlo simulation of the MAV minimum time trajectory planning problem considering many obstacles shows the efficiency of proposed initial guess strategies. In Chapter 5, the proposed trajectory optimization method will be applied to a MAV 3D trajectory planning problem with unknown dynamic parameters. To deal with the uncertainties and noise, the trajectory planning algorithm is embedded into a

receding horizon framework. An extended Kalman filter is applied to update the unknown aerodynamic coefficients for the next planning horizon. Conclusions and future works are discussed in Chapter 6.

# CHAPTER TWO: PRELIMINARY KNOWLEDGE

Trajectory Optimization and Nonlinear Programming

A typical trajectory optimization problem is also an optimal control problem [37]. It is to find the states $\boldsymbol{x}$, controls $\boldsymbol{u}$, and final time $t_f$ (if it is free) that will minimize the cost function

$$J = \varphi[\boldsymbol{x}(t_f), t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x}, \boldsymbol{u}, t)dt \tag{1}$$

in which the inequality constraints

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}, t) \le 0, \ \boldsymbol{g} \in \Re^{p\times 1} \tag{2}$$

and equality constraints

$$\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, t) = 0, \ \boldsymbol{h} \in \Re^{q\times 1} \tag{3}$$

are considered. The boundary conditions

$$\boldsymbol{\psi}[\boldsymbol{x}(t_0), \boldsymbol{x}(t_f), t_0, t_f] = 0, \ \boldsymbol{\psi} \in \Re^{l\times 1} \tag{4}$$

and the equations of motion

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \ \boldsymbol{x} \in \Re^{n\times 1}, \ \boldsymbol{u} \in \Re^{m\times 1} \tag{5}$$

where $q = n + l$, are considered as the equality constraints.

Nonlinear programming (NLP) problem [61]-[63] requires finding the $n$ vector $\boldsymbol{x}$ to minimize

$$F(\boldsymbol{x}) \tag{6}$$

subject to the $m$ constraints

$$\boldsymbol{c}_l \le \boldsymbol{c}(\boldsymbol{x}) \le \boldsymbol{c}_u \tag{7}$$

and $n$ bounds

$$\boldsymbol{x}_l \le \boldsymbol{x} \le \boldsymbol{x}_u \tag{8}$$

The equality constraints are imposed when $\boldsymbol{c}_l = \boldsymbol{c}_u$.

By extending to an infinite number of variables, the necessary conditions of NLP problem (i.e. Karush-Kuhn-Tucker (KKT) equation)) will approach the optimal control problem [62]. More detailed interpretation and proof can be found in [14].

Converting an trajectory optimization problem to a NLP problem requires the time interval $\left[t_0, t_f\right]$ to be divided into a prescribed number of subintervals, $t_k, k = 1, 2, ..., N$, whose endpoints are called nodes. The parameters that need to solve in the NLP problem are the unknown controls and states at these nodes. Then, the cost function and the state/constraint equations in NLP problem can be represented by using these parameters and solved by any standard NLP software.

## Predator-Prey Relations in Bio-phenomenon

Through millions of years of evolution, countless biological species have adopted simple yet effective behavior rules to accomplish their necessary tasks, particularly with respect to pursuit/evasion motion behavior [68]. Three natural behaviors to be used in the dissertation are:

- Motion Camouflage (MC) observed in hoverflies: The predator flies on a path connecting the prey and a background reference point to conceal its motion from the view of the prey [69];
- Constant Absolute Target Direction (CATD) motion observed in bats and sometimes in

dragonflies: The relative position between the predator and the prey keeps parallel to the initial relative position [70]-[72];

- Local Pursuit (LP) motion observed in ants: The velocity of the ant is always toward the position of the ant ahead of it [73]-[74].

In order to formulate the above behaviors, let us first define $\boldsymbol{x}_a$ as the position vector of predator, and $\boldsymbol{x}_p$ as the position vector of prey.

Motion Camouflage

The motion relation of MC strategy is shown in Figure 1. The background reference point is defined as $\boldsymbol{x}_r$, and the motion between predator and prey can be derived as [64]

$$\boldsymbol{x}_a(t) = \boldsymbol{x}_r + v(t)\left[\boldsymbol{x}_p(t) - \boldsymbol{x}_r(t)\right] \tag{9}$$

where $v$ shows the ratio of the relative distance between the prey and the reference point and the relative distance between the predator and the reference point. $v > 0$ is defined as the path control parameter (PCP) since it controls the position of predator:

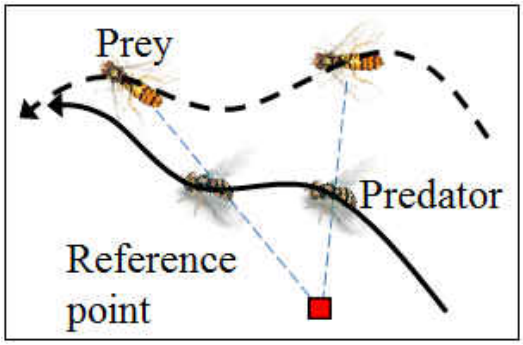



Figure 1 MC motion [64]

Constant Absolute Target Direction

As shown in Figure 2, to keep the relative position between the predator and the prey always parallel to the initial relative position, the formulation of the predator is governed by [64]

$$\boldsymbol{x}_a(t) = \boldsymbol{x}_p(t) + v(t)\left(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0}\right) \tag{10}$$

where the subscript 0 denotes the initial time. Note that to make the equation meaningful, $\boldsymbol{x}_{a,0} \neq \boldsymbol{x}_{p,0}$. As the path control parameter, $v(t)$ controls the distance between the prey and predator:

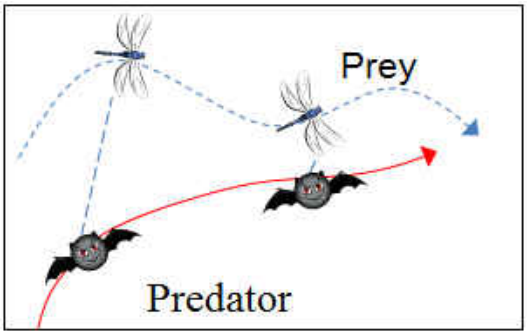



Figure 2 CATD motion [64]

Local Pursuit

The LP motion is shown in Figure 3, in which the direction of predator's velocity always points to the prey ahead. The formulation can be described as [64]

$$\dot{\boldsymbol{x}}_a(t) = v(t)\left[\boldsymbol{x}_p(t-\Delta) - \boldsymbol{x}_a(t)\right] \tag{11}$$

in which $\Delta$ represents the number of steps the prey ahead of the predator. Here, different from above two methods, $v(t)$ is a speed control parameter that determines the magnitude of the

predator's velocity. Given the position vectors of the prey and the predator, the speed of the predator becomes larger as $v(t)$ increases.

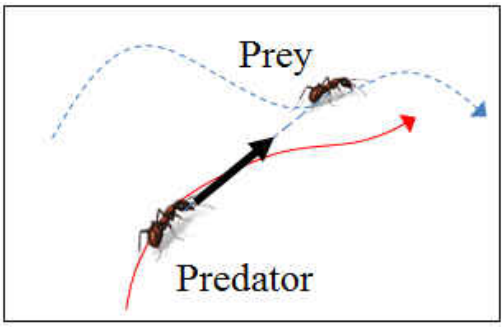



Figure 3 LP motion [64]

# CHAPTER THREE: BIO-INSPIRED TRAJECTORY OPTIMIZATION METHOD

The content of this chapter is modified from our journal paper [64]. As mentioned in the Chapter 1, some drawbacks existed in traditional direct trajectory optimization methods are: (1) the dimension of the converted Nonlinear Programming (NLP) problems are large; and (2) the computation complexity is high. Thus, it is necessary to find a method to reduce computational complexity and cost of the converted NLP problems.

Three kinds of motion strategies used by certain biological species have been illustrated and formulated in the Chapter 2. The common feature observed in these motion strategies is: "the predator only moves along paths in a certain subspace (or manifold)." Inspired by this observation, if consider the vehicle as a predator, the trajectory of the vehicle can be optimized in the subspace generated by a virtual prey [64]. The optimization time should be smaller than the most of the other existing methods which attempt to find a solution by searching all possible directions either locally or globally.

## Problem Definition

**Problem 1 (Nonlinear constrained optimal trajectory planning):** An trajectory optimization problem requires to find the states $\boldsymbol{x}$, controls $\boldsymbol{u}$, and final time $t_f$ (if it is free) to minimize a cost function $J = \varphi[\boldsymbol{x}(t_f), t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x}, \boldsymbol{u}, t)dt$, while the inequality constraints $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}, t) \le 0$ and equality constraints $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, t) = 0$ cannot be violated, as [54],

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{u},t} \; J = \varphi[\boldsymbol{x}(t_f), t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x},\boldsymbol{u},t)dt \\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x},\boldsymbol{u},t) \le 0 \\ &\qquad\qquad \boldsymbol{h}(\boldsymbol{x},\boldsymbol{u},t) = 0 \end{aligned} \tag{12}$$

As mentioned in the Chapter 2, the equality constraints include both dynamic equation and boundary conditions of the vehicle.

**Remark 1:** For a wide variety of dynamical models, such as mobile robots, MAV/ UAV, etc., the state vector $\boldsymbol{x} \in \Re^{n\times 1}$ includes two parts. The part of the state variables, which is relate to the "position" information of the system, is regarded as the "position" state vector as $\boldsymbol{x}_a \in \Re^{n_a \times 1}$, and the remaining part, if exists, is regarded as the "state rate" vector $\boldsymbol{x}_{sr} \in \Re^{(n-n_a)\times 1}$.

**Remark 2:** To solve **Problem 1**, the continuous states need to be discretisized, e.g. to $N$ nodes by using Legendre-Gauss-Lobatto (LGL) methods [54]. Thus, solve **Problem 1** is equivalent to solve NLP **Problem 2**, defined as,

$$\begin{aligned} &\min_{\boldsymbol{x}_k,\boldsymbol{u}_k,t_N} \; J = \varphi[\boldsymbol{x}_{(1)}, t_f] + \frac{t_f - t_0}{2} \sum_{k=0}^{N} L(\boldsymbol{x}_k, \boldsymbol{u}_k, t_k) w_k \\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, t_k) \le 0 \\ &\qquad\qquad \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, t_k) = 0 \end{aligned} \tag{13}$$

where $k = 1, ..., N$.

**Note 1:** The solution of **Problem 1** is equal to the limiting solution of **Problem 2** as the number of discretized nodes $N$ approaches $\infty$ [14].

**Remark 3**: There are three widely experienced boundary conditions (BC): (1) BC1: only initial and final $\boldsymbol{x}_a$ are known; (2) BC2: known initial $\boldsymbol{x}$ and final $\boldsymbol{x}_a$; (3) BC3: known initial and final $\boldsymbol{x}$.

Presentation of Prey Motion

During the optimization, we cannot just simply mimic the motion strategies of the animal, e.g. optimizing the path of predator in a fixed subspace generated by the virtual prey, because an optimal solution may not be generated in a fixed subspace. Thus, to obtain an optimal trajectory, the subspace formed by the virtual prey needs to be simultaneously optimized [65]. In this dissertation, the prey path is represented by B-spline method of which the shape can be affined by only a small number of control points [66].

For example, the $j^{th}$ direction of the virtual prey "position" at node $k$, $x_{p,j,k}, j=1,...,n_a$, $k=0,...,N$ is represented by a nonlinear rational B-spline curve of degree $d$ as [67]

$$x_{p,j,k}=\sum_{i=0}^{n_{cp}} B_{i,d}\left(t_k\right)P_{j,i},\ j=1,...,n_a,\ k=0,...,N \tag{14}$$

where $B_{i,d}\left(t\right), i=0,...,n_{cp}$ are the $d^{th}$ degree basis functions, $P_{j,i}, i=0,...,n_{cp}$ are the control points for the $j^{th}$ direction of the virtual prey "position", and $n_{cp}+1$ is the number of control points. The B-spline representation of the virtual prey motion Eq. (14) can also be written in the vector form as

$$\boldsymbol{\zeta}_{p,j}=B\boldsymbol{P}_j,\ j=1,...,n_a \tag{15}$$

in which $B=[B_{i,d}\left(t_k\right)]$, $i=0,...,n_{cp}$, $k=0,...,N$, and $\boldsymbol{P}_j^T=[P_{j,0},...P_{j,n_{cp}}]$ is the column vector of the control points.

**Remark 4**: The number of control points is not necessarily the same as the number of discretization nodes. Since the prey path is virtual, a much smaller number of control points are enough to define a flexible virtual prey motion, while a larger number of discretization nodes,

where the performance index and constraints are evaluated, are needed to define the actual vehicle's trajectory.

The $l^{th}$ derivative of $x_{p,j,k}$ can be shown either in the scalar form as [67]

$$\frac{d^l x_{p,j,k}}{dt^l} = \sum_{m=0}^{n_{cp}} B_{m,d}^l \left(t_k\right) P_{j,m},\ j = 1,...,n_a, k = 0,...,N \tag{16}$$

or in the matrix form as

$$\boldsymbol{\zeta}_{p,j}^{(l)} = B^{(l)} \boldsymbol{P}_j \quad ,\ j = 1,...,n_a \tag{17}$$

in which $B^{(l)} = [B_{m,d}^{(l)}\left(t_k\right)]$, $m = 0,...,n_{cp}, k = 0,...,N$ and the superscript $(l)$ represents the $l^{th}$ derivative.

Property 1: For a non-periodic knot vector, $B_{0,d}\left(t_0\right) = B_{n_{cp},d}\left(t_N\right) = 1$ and $B_{k,d}\left(t_N\right) = 0$, $k = 1,...,n_{cp} - 1$.

The other detail information of B-spline method can refer to [66]-[67].

## MC based Trajectory Optimization

### Problem Formulation

Assuming a vehicle (e.g. MAV, UAV, ground robot, etc.) as a predator, by using the MC strategy, its trajectory can be represented by a virtual prey path $\boldsymbol{x}_p$, a reference point $\boldsymbol{x}_r$, and PCPs $v$. According to MC motion rule (Eq.(9)), finding the higher order derivatives of $\boldsymbol{x}_a$ is easy. For example, the 1st and 2nd derivatives can be represented as

$$\begin{aligned}\dot{\boldsymbol{x}}_a(t) &= \dot{v}(t)(\boldsymbol{x}_p(t)-\boldsymbol{x}_r)+v(t)\dot{\boldsymbol{x}}_p(t)\\ \ddot{\boldsymbol{x}}_a(t) &= \ddot{v}(t)(\boldsymbol{x}_p(t)-\boldsymbol{x}_r)+2\dot{v}(t)\dot{\boldsymbol{x}}_p(t)+v(t)\ddot{\boldsymbol{x}}_p(t)\end{aligned} \tag{18}$$

After discretized by using LGL method, Eq.(9) and Eq. (18) become

$$\boldsymbol{\zeta}_{a,j} = \Xi\boldsymbol{\zeta}_{p,j} + \left(\boldsymbol{I}_{N+1}-\Xi\right)\boldsymbol{\xi}_j,\ j=1,...,n_a \tag{19}$$

and

$$\begin{aligned}\dot{\boldsymbol{\zeta}}_{a,j} &= \dot{\Xi}\left(\boldsymbol{\zeta}_{p,j}-\boldsymbol{\xi}_j\right)+\Xi\dot{\boldsymbol{\zeta}}_{p,j},\ j=1,...,n_a\\ \ddot{\boldsymbol{\zeta}}_{a,j} &= \ddot{\Xi}\left(\boldsymbol{\zeta}_{p,j}-\boldsymbol{\xi}_j\right)+2\dot{\Xi}\dot{\boldsymbol{\zeta}}_{p,j}+\Xi\ddot{\boldsymbol{\zeta}}_{p,j},\ j=1,...,n_a\end{aligned} \tag{20}$$

respectively, where $n_a$ is the position dimension of the vehicle, $\boldsymbol{\xi}_j = x_{r,j}\boldsymbol{1}$, $j=1,...,n_a$, of which $\boldsymbol{1}\in\Re^{(N+1)\times 1}$ is a vector and all of its elements equal to one, and $x_{r,j}$ is the $j^{th}$ component of the reference point. $\boldsymbol{\zeta}_{a,j}^T = \left[x_{a,j,0},...,x_{a,j,N}\right]$ and $\boldsymbol{\zeta}_{p,j}^T = \left[x_{p,j,0},...,x_{p,j,N}\right]$. The derivative of the PCP variable matrix $\Xi$ is

$$d^r\Xi / dt^r \triangleq diag\left\{d^r v_k / dt^r\right\}, k=0,...,N, r=1,2,... \tag{21}$$

As mentioned in the Chapter 2, after convert **Problem 2** to a NLP problem, the unknowns are the state and control variables at the nodes $k=0,...,N$. Thus, excludes $\boldsymbol{x}_a$, $\dot{\boldsymbol{x}}_a$ and $\dot{\boldsymbol{x}}_a$, the other state rates variables $\boldsymbol{x}_{sr}$ and control variables also need to be expressed in terms of $\boldsymbol{x}_r$, $v$, and $\boldsymbol{x}_p$. They can be calculated either by inverse dynamics of the vehicle or varying manifold based feedback linearization method. Here, only the varying manifold method will be introduced.

If the dynamic of a vehicle is written as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})+\boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} \tag{22}$$

by assuming that the output model is $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{x}_a$, and the relative degree is $l$, a manifold can be constructed as

$$\boldsymbol{s} = \frac{d^{l-1}\boldsymbol{y}}{dt^{l-1}} - \frac{d^{l-1}}{dt^{l-1}}\left[ v(\boldsymbol{x}_p - \boldsymbol{x}_r) + \boldsymbol{x}_r \right] \tag{23}$$

For a vehicle to stay on the varying manifold, the derivative of $\boldsymbol{s}$ should be zero, thus,

$$\dot{\boldsymbol{s}} = \frac{d^{l}\boldsymbol{y}}{dt^{l}} - \frac{d^{l}}{dt^{l}}\left[ v(\boldsymbol{x}_p - \boldsymbol{x}_r) + \boldsymbol{x}_r \right] = 0 \tag{24}$$

since the relative degree of y is $l$, it can be represented by Lie derivatives as $\boldsymbol{y}^{(l)} = L_f^l \boldsymbol{h}(\boldsymbol{x}) + L_g L_f^{l-1} \boldsymbol{h}(\boldsymbol{x})\boldsymbol{u}$. Also since the reference point in the varying manifold is constant, Eq. (24) can be used to solve for the control variables (i.e. the equivalent control) that can drive the vehicle on the manifold

$$\begin{aligned} \boldsymbol{u} &= \left[ L_g L_f^{l-1} \boldsymbol{h}(\boldsymbol{x}) \right]^{-1} \left\{ \frac{d^l}{dt^l}\left[ v(\boldsymbol{x}_p - \boldsymbol{x}_r) \right] - L_f^l \boldsymbol{h}(\boldsymbol{x}) \right\} \\ &= \left[ L_g L_f^{l-1} \boldsymbol{h}(\boldsymbol{x}) \right]^{-1} \left\{ \sum_{k=0}^{l} C_l^k \frac{d^k v}{dt^k}(\boldsymbol{x}_p - \boldsymbol{x}_r) + \sum_{k=0}^{l} C_l^{l-k} v \frac{d^{l-k}\boldsymbol{x}_p}{dt^{l-k}} - L_f^l \boldsymbol{h}(\boldsymbol{x}) \right\} \end{aligned} \tag{25}$$

All the states and control variables are now in terms of $\boldsymbol{x}_p$, $\boldsymbol{x}_r$ and $v$, equivalent with **Problem 1**, trajectory optimization problem is now to solve,

$$\begin{aligned} &\min_{\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t_f} \quad J = \varphi[\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t)dt \\ \textbf{(Problem 3)}\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t) \le 0 \\ &\qquad\qquad \boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t) = 0 \end{aligned} \tag{26}$$

Note that now equality constraints $\boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t) = 0$ only include boundary conditions, because the dynamic equation of the vehicle has been already considered when calculating states and control variables.

After discretized and represented virtual prey motion by B-spline method (Eq. (17)), solving Eq. (26) is equivalent to solving NLP **Problem 4**,

$$\begin{aligned} &\min_{\boldsymbol{P}_j, x_{r,j}, \boldsymbol{v}, t} J = \varphi[\boldsymbol{P}_N, x_{r,j}, \boldsymbol{v}_N, t_f] + \frac{t_f - t_0}{2} \sum_{k=0}^{N} L(\boldsymbol{P}_{j,k}, x_{r,j}, \boldsymbol{v}_k, t) w_k \\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}_j, x_{r,j}, \boldsymbol{v}, t) \le 0 \\ &\qquad\qquad \boldsymbol{h}'(\boldsymbol{P}_j, x_{r,j}, \boldsymbol{v}, t) = 0 \end{aligned} \tag{27}$$

where, $\boldsymbol{P}_j^T = [P_{j,0}, ... P_{j,n_{cp}}]$, and $\boldsymbol{v}^T = [v_0, ..., v_N]$.

Boundary Conditions Incorporation

The boundary conditions in the proposed method can be incorporated when calculating certain control points or PCPs. This section will talk about how to incorporate the three boundary conditions (illustrated in the **Remark 1**).

**Lemma 1:** For BC1, the initial and final PCPs can be selected as 1. The first and last control points of the virtual prey motion are $P_{j,0} = x_{a,j,0}$ and $P_{j,n_{cp}} = x_{a,j,N}$, respectively, where $j = 1, ..., n_a$.

**Proof**. In the MC strategy (Eq. (9)), the initial and final PCPs can be selected as 1. Then the initial and final positions of the vehicle and virtual prey satisfy $\boldsymbol{x}_{a,0} = \boldsymbol{x}_{p,0}$ and $\boldsymbol{x}_{a,N} = \boldsymbol{x}_{p,N}$.

Representing the virtual prey motion in B-splines, $x_{p,j,0} = \sum_{m=0}^{n_{cp}} B_{m,d}(t_0) P_{j,m}$ and

$x_{p,j,N} = \sum_{m=0}^{n_{cp}} B_{m,d}(t_N) P_{j,m}$, $j = 1, ..., n_a$. Utilizing Property 1, $P_{j,0} = x_{a,j,0}$ and

$P_{j,n_{cp}} = x_{a,j,N}, j = 1, ..., n_a$.

**Lemma 2:** For BC2, the initial and final PCPs can be selected as 1, and the first and last control points are $P_{j,0} = x_{a,j,0}$ and $P_{j,n_{cp}} = x_{a,j,N}$, $j = 1, ..., n_a$. $v_1$ should satisfy

$$v_1 = \left[ (x_{p,k,0} - x_{r,k}) D'_{01} \right]^{-1} \cdot \left\{ \dot{x}_{a,k,0} - (x_{p,k,0} - x_{r,k}) \sum_{j=0,\, j\neq 1}^{N} D'_{0j} v_j - v_0 \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \right\} \tag{28}$$

in which $k$ is an arbitrary selected direction of $\boldsymbol{x}_a$ but satisfies $x_{p,k,0} \neq x_{r,k}$. Here the subscript in $D'$ denotes the entry of the matrix. $D$ is the differentiation matrix in the LGL pseudospectral discretization, and $D' \triangleq \left[ 2/(t_f - t_0) \right] D$ [49], $P_{m,1}, m = 1,...,n_a, m \neq k$ satisfies

$$\begin{aligned} P_{m,1} = {} & \frac{1}{\dot{B}_{1,d}(t_0)} \left\{ \dot{x}_{a,m,0} - (x_{p,m,0} - x_{r,m}) \sum_{j=0,\, j\neq 1}^{N} D'_{0j} v_j - \sum_{l=0, l\neq 1}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{m,l} \right\} \\ & - \frac{\left[ (x_{p,m,0} - x_{r,m}) \right]}{\dot{B}_{1,d}(t_0) \left[ (x_{p,k,0} - x_{r,k}) \right]} \left\{ \dot{x}_{a,k,0} - (x_{p,k,0} - x_{r,k}) \sum_{j=0,\, j\neq 1}^{N} D'_{0j} v_j - \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \right\} \end{aligned} \tag{29}$$

**Proof**: Known the boundary "position states" of the vehicle, based on **Lemma 2**, $v_0 = v_N = 1$, and $P_{j,0} = x_{a,j,0}$ and $P_{j,n_{cp}} = x_{a,j,N}$.

Based on (Eq. (9)), the initial velocity of the vehicle is $\dot{\boldsymbol{x}}_{a,0} = \dot{v}_0 (\boldsymbol{x}_{p,0} - \boldsymbol{x}_r) + v_0 \dot{\boldsymbol{x}}_{p,0}$. Let's select the $k^{th}$ component of this equation $\dot{x}_{a,k,0} = \dot{v}_0 (x_{p,k,0} - x_{r,k}) + v_0 \dot{x}_{p,k,0}$ if $x_{p,k,0} \neq x_{r,k}$. Approximating the PCP vector using the pseudospectral collocation method and representing the virtual prey motion in B-spline as shown in Eq. (14), the following equation can be achieved

$$\dot{x}_{a,k,0} = (x_{p,k,0} - x_{r,k}) \sum_{j=0}^{N} D'_{0j} v_j + v_0 \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \tag{30}$$

Equation (30) can be rearranged to calculate $v_1$ as in Eq. (28).

If $n_a > 1$, $v_1$ calculated from different directions should be the same, i.e.,

$$\begin{aligned}&\left[(x_{p,k,0}-x_{r,k})\right]\cdot\left\{\dot{x}_{a,m,0}-(x_{p,m,0}-x_{r,m})\sum_{j=0,\,j\neq 1}^{N}D_{0j}^{'}v_j-v_0\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{m,l}\right\}\\&=\left[(x_{p,m,0}-x_{r,m})\right]\cdot\left\{\dot{x}_{a,k,0}-(x_{p,k,0}-x_{r,k})\sum_{j=0,\,j\neq 1}^{N}D_{0j}^{'}v_j-v_0\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}\right\}\end{aligned}\tag{31}$$

where, $m\neq k, m\in[1,...,n_a]$. Since $v_0=1$, $P_{m,1}, m\neq k, m\in[1,...,n_a]$ can be calculated from Eq.(29) instead of being optimized.

**Lemma 3:** For BC3, the initial and final PCPs can be selected as 1, $P_{j,0}=x_{a,j,0}$ and $P_{j,n_{cp}}=x_{a,j,N}$, $j=1,...,n_a$. $v_1$ and $v_{N-1}$ have to satisfy

$$\begin{aligned}\begin{bmatrix}v_1\\v_{N-1}\end{bmatrix}=&\begin{bmatrix}D_{01}^{'}(x_{p,k,0}-x_{r,k}) & D_{0(N-1)}^{'}(x_{p,k,0}-x_{r,k})\\D_{N1}^{'}(x_{p,k,N}-x_{r,k}) & D_{N(N-1)}^{'}(x_{p,k,N}-x_{r,k})\end{bmatrix}^{-1}\cdot\\&\begin{bmatrix}\dot{x}_{a,k,0}-\sum\limits_{\substack{j=0,\,j\neq 1\\ j\neq N-1}}^{N}D_{0j}^{'}(x_{p,k,0}-x_{r,k})v_j-v_0\sum\limits_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}\\\dot{x}_{a,k,N}-\sum\limits_{\substack{j=0,\,j\neq 1\\ j\neq N-1}}^{N}D_{Nj}^{'}(x_{p,k,N}-x_{r,k})v_j-v_N\sum\limits_{i=0}^{n_{cp}}\dot{B}_{l,d}\left(t_N\right)P_{k,l}\end{bmatrix}\end{aligned}\tag{32}$$

in which the $k^{th}$ direction is selected such that the matrix inversion in Eq. (32) exists. $P_{m,1}$ and $P_{m,n_{cp}-1}$, $m=1,...,n_a$, $m\neq k$ satisfy

$$
\begin{bmatrix} P_{m,1} \\ P_{m,n_{cp}-1} \end{bmatrix} =
$$

$$
\begin{bmatrix} \dot{B}_{1,d}(t_0) & \dot{B}_{n_{cp}-1,d}(t_0) \\ \dot{B}_{1,d}(t_N) & \dot{B}_{n_{cp}-1,d}(t_N) \end{bmatrix}^{-1} \cdot \begin{bmatrix} \dot{x}_{a,m,0} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{0j}(x_{p,m,0} - x_{r,m}) v_j - \sum_{\substack{l=0;l\neq 1 \\ l\neq n_{cp}-1}}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{m,l} \\ \dot{x}_{a,m,N} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{Nj}(x_{p,m,N} - x_{r,m}) v_j - \sum_{\substack{l=0;;l\neq 1 \\ l\neq n_{cp}-1}}^{n_{cp}} \dot{B}_{l,d}(t_N) P_{m,l} \end{bmatrix}
$$

$$
- \begin{bmatrix} \dot{B}_{1,d}(t_0) & \dot{B}_{n_{cp}-1,d}(t_0) \\ \dot{B}_{1,d}(t_N) & \dot{B}_{n_{cp}-1,d}(t_N) \end{bmatrix}^{-1} \cdot \begin{bmatrix} D'_{01}(x_{p,m,0} - x_{r,m}) & D'_{0(N-1)}(x_{p,m,0} - x_{r,m}) \\ D'_{N1}(x_{p,m,N} - x_{r,m}) & D'_{N(N-1)}(x_{p,m,N} - x_{r,m}) \end{bmatrix}
$$

$$
\cdot \begin{bmatrix} D'_{01}(x_{p,k,0} - x_{r,k}) & D'_{0(N-1)}(x_{p,k,0} - x_{r,k}) \\ D'_{N1}(x_{p,k,N} - x_{r,k}) & D'_{N(N-1)}(x_{p,k,N} - x_{r,k}) \end{bmatrix}^{-1}
$$

$$
\cdot \begin{bmatrix} \dot{x}_{a,k,0} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{0j}(x_{p,k,0} - x_{r,k}) v_j - \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \\ \dot{x}_{a,k,N} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{Nj}(x_{p,k,N} - x_{r,k}) v_j - \sum_{i=0}^{n_{cp}} \dot{B}_{l,d}(t_N) P_{k,l} \end{bmatrix} \tag{33}
$$

**Proof**: Based on **Lemma 3**, $v_0 = v_N = 1, P_{i,0} = x_{a,j,0}$, and $P_{j,n_{cp}} = x_{a,j,N}, j = 1,...,n_a$. The velocities of the vehicle are calculated by $\dot{\boldsymbol{x}}_{a,0} = \dot{v}_0(\boldsymbol{x}_{p,0} - \boldsymbol{x}_r) + v_0 \dot{\boldsymbol{x}}_{p,0}$ and $\dot{\boldsymbol{x}}_{a,N} = \dot{v}_N(\boldsymbol{x}_{p,N} - \boldsymbol{x}_r) + v_N \dot{\boldsymbol{x}}_{p,N}$, respectively. Let's select any $k^{th}$ component of these two equations. Similar to the proof of **Lemma 2**, when approximating the PCP vector in the pseudospectral discretization and the virtual prey motion in the B-spline curve, Eq. (32) can be achieved.

The selection of the $k^{th}$ component should guarantee that the matrix inversion in Eq. (32) exists. If $n_a > 1$, $v_1$ and $v_{N-1}$ calculated from different directions should be the same, following the same procedure used in proofing **Lemma 2**, Eq. (33) can be derived.

### Optimization Algorithm

Since all of the boundary conditions have been incorporated to solve the certain PCPs and control points, the equality constraints are eliminated. Thus, Eq. (27) is equivalent to solving,

$$\textbf{(Problem 5)} \quad \min_{P_j, x_{r,j}, \boldsymbol{v}, t} J = \varphi[\boldsymbol{P'}_N, x_{r,j}, \boldsymbol{v'}_N, t_f] + \frac{t_f - t_0}{2} \sum_{k=0}^{N} L(\boldsymbol{P'}_{j,k}, x_{r,j}, \boldsymbol{v'}_k, t) w_k, \; j = 1, ..., n_a \quad (34)$$
$$\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}', x_{r,j}, \boldsymbol{v'}, t) \le 0$$

where, $\boldsymbol{P'}$, $x_{r,j}$, $\boldsymbol{v'}$ and $t$ are considered as optimizable parameters and can be solved by nonlinear programming methods. Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ under three boundary conditions are summarized in **Table 1** .

Table 1 Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ in the MC Based Optimization Procedure

| BC | MC Motion Rule |
|---|---|
| 1 | $v_i, i = 1, ..., N-1,$ |
| | $P_{j,i}, i = 1, ..., n_{cp} - 1, \; j = 1, ..., n_a$ |
| 2 | $v_i, i = 2, ..., N-1,$ |
| | $P_{k,i}, i = 1, ..., n_{cp} - 1, k \in [1, n_a]$ |
| | $P_{m,i}, i = 2, ..., n_{cp} - 1, m = 1, ..., n_a, m \ne k$ |
| 3 | $v_i, i = 2, ..., N-2,$ |
| | $P_{k,i}, i = 1, ..., n_{cp} - 1, k \in [1, n_a]$ |
| | $P_{m,i}, i = 2, ..., n_{cp} - 2, m = 1, ..., n_a, m \ne k$ |

## CATD based Trajectory Optimization

### Problem Formulation

Assuming a vehicle (e.g. MAV, UAV, ground robot, etc.) as a predator, by using the CATD strategy, its trajectory will be described by the position of virtual prey $\boldsymbol{x}_p(t)$, initial

relative position between the prey and predator $\left(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right)$ and PCPs $v(t)$, as shown in Eq. (10) . Then, higher order derivatives of $\boldsymbol{x}_a$, such as the 1st and 2nd order derivatives, are represented as,

$$\begin{aligned}\dot{\boldsymbol{x}}_a(t)&=\dot{\boldsymbol{x}}_p(t)+\dot{v}(t)\left(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right)\\ \ddot{\boldsymbol{x}}_a(t)&=\ddot{\boldsymbol{x}}_p(t)+\ddot{v}(t)\left(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right)\end{aligned} \tag{35}$$

**Remark 5:** $\boldsymbol{x}_{a,0}\neq\boldsymbol{x}_{p,0}$ $P_{j,0}\neq x_{a,j,0}$ is required. This can be relaxed as an inequality constraint of $\boldsymbol{\varepsilon}-\left|\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right|\leq\mathbf{0}$ in which $\boldsymbol{\varepsilon}$ is a vector constitute by small positive numbers.

After discretization, Eq. (10) and Eq. (35) become

$$\boldsymbol{\zeta}_{a,j}=I_{N+1}\boldsymbol{\zeta}_{p,j}+\Xi\boldsymbol{\xi}_j,\qquad j=1,...,n_a \tag{36}$$

and

$$\begin{aligned}\dot{\boldsymbol{\zeta}}_{a,j}&=I_{N+1}\dot{\boldsymbol{\zeta}}_{p,j}+\dot{\Xi}\boldsymbol{\xi}_j,\qquad j=1,...,n_a\\ \ddot{\boldsymbol{\zeta}}_{a,j}&=I_{N+1}\ddot{\boldsymbol{\zeta}}_{p,j}+\ddot{\Xi}\boldsymbol{\xi}_j,\qquad j=1,...,n_a\end{aligned} \tag{37}$$

Here, $\boldsymbol{\xi}_j=\boldsymbol{\zeta}_{a,j,0}-\boldsymbol{\zeta}_{p,j,0}$, in which $\boldsymbol{\zeta}_{a,j,0}=x_{a,j}(t_0)\boldsymbol{1}$ and $\boldsymbol{\zeta}_{p,j,0}=x_{p,j}(t_0)\boldsymbol{1}$.

To describe $\boldsymbol{x}_{sr}$ and control variables in terms of $\left(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right)$, $v$, and $\boldsymbol{x}_p$, we can use either inverse dynamics or varying manifold based feedback linearization method, depends on whether the system is control affine. Similarly, only the varying manifold method will be introduced. When the dynamic of the vehicle is a control affine system (defined in Eq.(22)) and the output model is $\boldsymbol{y}=\boldsymbol{h}(\boldsymbol{x})=\boldsymbol{x}_a$, a manifold can be constructed as

$$\boldsymbol{s}=\frac{d^{l-1}\boldsymbol{y}}{dt^{l-1}}-\frac{d^{l-1}}{dt^{l-1}}\left[\boldsymbol{x}_p(t)+v(t)\left(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0}\right)\right] \tag{38}$$

where $l$ is the relative degree. To maintain a vehicle staying on the varying manifold, the derivative of $\boldsymbol{s}$ should be zero, thus,

$$\dot{\boldsymbol{s}} = \frac{d^l \boldsymbol{y}}{dt^l} - \frac{d^l}{dt^l}\left[\boldsymbol{x}_p(t) + v(t)\left(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0}\right)\right] = 0 \tag{39}$$

With the similar derivation in the MC rule, the control variables in LP rule can be calculated by

$$\begin{aligned} \boldsymbol{u} &= \left[L_g L_f^{l-1}\boldsymbol{h}(\boldsymbol{x})\right]^{-1}\left\{\frac{d^l}{dt^l}\left[\boldsymbol{x}_p(t) + v(t)\left(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0}\right)\right] - L_f^l \boldsymbol{h}(\boldsymbol{x})\right\} \\ &= \left[L_g L_f^{l-1}\boldsymbol{h}(\boldsymbol{x})\right]^{-1}\left\{\sum_{k=0}^{l} C_l^k \frac{d^k v}{dt^k}(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0}) + \sum_{k=0}^{l} C_l^{l-k} v \frac{d^{l-k}\boldsymbol{x}_p}{dt^{l-k}} - L_f^l \boldsymbol{h}(\boldsymbol{x})\right\} \end{aligned} \tag{40}$$

After all the states and control variables being represented in terms of $\boldsymbol{x}_p$, $\left(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0}\right)$ and $v$, equivalent with **Problem 2**, trajectory optimization problem is now to solve,

$$\begin{aligned} &\min_{\boldsymbol{x}_p, \boldsymbol{v}, t_f} \quad J = \varphi[\boldsymbol{x}_p, \boldsymbol{v}, t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x}_p, \boldsymbol{v}, t)dt \\ \textbf{(Problem 6)}\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x}_p, \boldsymbol{v}, t) \le 0 \\ &\qquad\qquad\ \ \boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{v}, t) = 0 \end{aligned} \tag{41}$$

Note that similarly as MC rule, now equality constraints $\boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t) = 0$ only include boundary conditions, because the dynamic equation of the vehicle has been already considered when calculating states and control variables. If combine with Eq. (17)), solving Eq. (41) will be equivalent to solving **Problem 7**,

$$\begin{aligned} &\min_{\boldsymbol{P}_j, \boldsymbol{v}, t} \ J = \varphi[\boldsymbol{P}_N, \boldsymbol{v}_N, t_f] + \frac{t_f - t_0}{2}\sum_{k=0}^{N} L(\boldsymbol{P}_{j,k}, \boldsymbol{v}_k, t)w_k \quad , j = 1, ..., n_a \\ &\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}_j, \boldsymbol{v}, t) \le 0 \\ &\qquad\qquad\ \ \boldsymbol{h}'(\boldsymbol{P}_j, \boldsymbol{v}, t) = 0 \end{aligned} \tag{42}$$

where, $\boldsymbol{P}_j^T = [P_{j,0}, ...P_{j,n_{cp}}]$, and $\boldsymbol{v}^T = [v_0, ..., v_N]$.

Boundary Conditions Incorporation

This section will talk about how to incorporate the three boundary conditions (illustrated in the **Remark 1**) by using CATD motion rule.

**Lemma 4:** $v_0$ in the CATD motion rule need to be 1.

**Proof:** According to Eq. (10), when $t=0$, $\boldsymbol{x}_{a,0}=\boldsymbol{x}_{p,0}+v_0(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0})$ valid only when $v_0=1$.

**Lemma 5:** For BC1, (1) $v_0=1$, $P_{j,0}$ could be guessed as any value but $P_{j,0}\neq x_{a,j,0}$, $j=1,...,n_a$;

(2) if $v_N$ is guessed, the last control point $P_{j,n_{cp}}=x_{a,j,N}-v_N(x_{a,j,0}-P_{j,0})$.

**Proof**: If $v_0=1$, $\boldsymbol{x}_{p,0}$ is overlapped with $\boldsymbol{x}_{a,0}$ ; The final position is $\boldsymbol{x}_{a,N}=\boldsymbol{x}_{p,N}+v_N(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0})$.

Thus if given $v_N$, $\boldsymbol{x}_{p,N}$ must satisfy $\boldsymbol{x}_{p,N}=\boldsymbol{x}_{a,N}-v_N(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0})$. Based on Property 1,

$P_{j,0}\neq x_{a,j,0}$ and $P_{j,n_{cp}}=x_{a,j,N}-v_N(x_{a,j,0}-P_{j,0})$, $j=1,...,n_a$.

**Lemma 6:** For BC2, $v_0=1$ and given any $v_N$, $P_{j,n_{cp}}=x_{a,j,N}-v_N(x_{a,j,0}-P_{j,0})$, $P_{j,0}\neq x_{a,j,0}$,

$i=1,...,n_a$. $v_1$ satisfies

$$v_1=\left[(x_{a,k,0}-x_{p,k,0})D_{01}^{'}\right]^{-1}\left\{\dot{x}_{a,k,0}-\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}-(x_{a,k,0}-x_{p,k,0})\sum_{\substack{j=0\\ j\neq 1}}^{N}D_{0j}^{'}v_j\right\} \tag{43}$$

in which the $k^{th}$ component must be selected such that $x_{a,k,0}\neq x_{p,k,0}$. $P_{m,1}$, where

$m=1,...,n_a, m\neq k$, must satisfies

$$P_{m,1}=\frac{1}{\dot{B}_{1,d}\left(t_0\right)}\left\{\dot{x}_{a,m,0}-(x_{a,m,0}-x_{p,m,0})\sum_{\substack{j=0\\ j\neq 1}}^{N}D_{0\,j}^{'}v_j-\sum_{l=0,l\neq 1}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{m,l}\right\}$$
$$-\frac{\left[(x_{a,m,0}-x_{p,m,0})D_{01}^{'}\right]}{\dot{B}_{1,d}\left(t_0\right)\left[(x_{a,k,0}-x_{p,k,0})D_{01}^{'}\right]}\cdot\left\{\dot{x}_{a,k,0}-(x_{a,k,0}-x_{p,k,0})\sum_{\substack{j=0\\ j\neq 1}}^{N}D_{0\,j}^{'}v_j-\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}\right\} \tag{44}$$

**Proof**: Based on **Lemma 5**, $v_0=1$ and given any $v_N$, $P_{j,0}\neq x_{a,j,0}$, and $P_{j,n_{cp}}=x_{a,j,N}-v_N(x_{a,j,0}-P_{j,0})$. The initial velocity of the predator can be simplified as $\dot{\boldsymbol{x}}_{a,0}=\dot{\boldsymbol{x}}_{p,0}+\dot{v}_0(\boldsymbol{x}_{a,0}-\boldsymbol{x}_{p,0})$. The $k^{th}$ component of this equation is select as $\dot{x}_{a,k,0}=\dot{x}_{p,k,0}+\dot{v}_0(x_{a,k,0}-x_{p,k,0})$. Approximating the PCP vector using the pseudospectral collocation and representing prey motion with a B-spline curve, the following equation can be achieved

$$\dot{x}_{a,k,0}=\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}+(x_{a,k,0}-x_{p,k,0})\sum_{j=0}^{N}D_{0\,j}^{'}v_j \tag{45}$$

Equation (45) can be rearranged to calculate $v_1$ as Eq. (44). If $n_a>1$, since $v_1$ calculated from different directions should be same, i.e. for $m\neq k, m\in[1,...,n_a]$,

$$\left[(x_{a,k,0}-x_{p,k,0})D_{01}^{'}\right]^{-1}\left\{\dot{x}_{a,k,0}-\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{k,l}-(x_{a,k,0}-x_{p,k,0})\sum_{\substack{j=0\\ j\neq 1}}^{N}D_{0\,j}^{'}v_j\right\}$$
$$=\left[(x_{a,m,0}-x_{p,m,0})D_{01}^{'}\right]^{-1}\left\{\dot{x}_{a,m,0}-\sum_{l=0}^{n_{cp}}\dot{B}_{l,d}\left(t_0\right)P_{m,l}-(x_{a,m,0}-x_{p,m,0})\sum_{\substack{j=0\\ j\neq 1}}^{N}D_{0\,j}^{'}v_j\right\} \tag{46}$$

Simplify Eq. (46), $P_{m,1}$, $m\in[1,...,n_a]$, $m\neq k$, can be calculated instead of being optimized as shown in Eq. (44).

**Lemma 7:** For BC3, $v_0 = 1$ and given any $v_N$, $P_{j,0} \neq x_{a,j,0}$, and $P_{j,n_{cp}} = x_{a,j,N} - v_N(x_{a,j,0} - P_{j,0})$. $v_1$ and $v_{N-1}$ should satisfy

$$\begin{bmatrix} v_1 \\ v_{N-1} \end{bmatrix} = \begin{bmatrix} D'_{01}(x_{a,k,0} - x_{p,k,0}) & D'_{0(N-1)}(x_{a,k,0} - x_{p,k,0}) \\ D'_{N1}(x_{a,k,0} - x_{p,k,0}) & D'_{N(N-1)}(x_{a,k,0} - x_{p,k,0}) \end{bmatrix}^{-1} \cdot \begin{bmatrix} \dot{x}_{a,k,0} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{0j}(x_{a,k,0} - x_{p,k,0}) v_j - \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \\ \dot{x}_{a,k,N} - \sum_{\substack{j=0,\, j\neq 1 \\ j\neq N-1}}^{N} D'_{Nj}(x_{a,k,0} - x_{p,k,0}) v_j - \sum_{i=0}^{n_{cp}} \dot{B}_{l,d}(t_N) P_{k,l} \end{bmatrix} \tag{47}$$

in which the $k^{th}$ component must be selected such that the matrix inversion in Eq. (47) exists. $P_{m,1}$ and $P_{m,N-1}$, $m = 1,...,n_a, m \neq k$, satisfy Eq. (48)

**Proof**: Based on **Lemma 5**, $v_0 = 1$ and given any $v_N$, $P_{j,0} \neq x_{a,j,0}$, and $P_{j,n_{cp}} = x_{a,j,N} - v_N(x_{a,j,0} - P_{j,0})$. The initial and final velocities of the vehicle are calculated by $\dot{\boldsymbol{x}}_{a,0} = \dot{\boldsymbol{x}}_{p,0} + \dot{v}_0(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0})$ and $\dot{\boldsymbol{x}}_{a,N} = \dot{\boldsymbol{x}}_{p,N} + \dot{v}_N(\boldsymbol{x}_{a,0} - \boldsymbol{x}_{p,0})$, respectively. Selecting the $k^{th}$ component of these two equations, and approximating the PCP vector in the pseudospectral collocation and the prey motion in the B-spline, Eq. (47) is achieved. The selection of the $k^{th}$ component should guarantee that the matrix inversion in Eq. (47) exists. If $n_a > 1$, $v_1$ and $v_{N-1}$ calculated from different directions should be same, therefore Eq. (48) can be obtained.

$$
\begin{bmatrix} P_{m,1} \\ P_{m,n_{cp}-1} \end{bmatrix} =
$$

$$
\begin{bmatrix} \dot{B}_{1,d}(t_0) & \dot{B}_{n_{cp}-1,d}(t_0) \\ \dot{B}_{1,d}(t_N) & \dot{B}_{n_{cp}-1,d}(t_N) \end{bmatrix}^{-1} \cdot \begin{bmatrix} \dot{x}_{a,m,0} - \sum_{\substack{j=0, j\neq 1 \\ j\neq N-1}}^{N} D'_{0j}(x_{a,m,0} - x_{p,m,0}) v_j - \sum_{\substack{l=0, l\neq 1 \\ l\neq N-1}}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{m,l} \\ \dot{x}_{a,m,N} - \sum_{\substack{j=0, j\neq 1 \\ j\neq N-1}}^{N} D'_{Nj}(x_{a,m,0} - x_{p,m,0}) v_j - \sum_{\substack{i=0, l\neq 1 \\ l\neq N-1}}^{n_{cp}} \dot{B}_{l,d}(t_N) P_{m,l} \end{bmatrix}
$$

$$
- \begin{bmatrix} \dot{B}_{1,d}(t_0) & \dot{B}_{n_{cp}-1,d} \\ \dot{B}_{1,d}(t_N) & \dot{B}_{n_{cp}-1,d}(t_N) \end{bmatrix}^{-1} \cdot \begin{bmatrix} D'_{01}(x_{a,m,0} - x_{p,m,0}) & D'_{0(N-1)}(x_{a,m,0} - x_{p,m,0}) \\ D'_{N1}(x_{a,m,0} - x_{p,m,0}) & D'_{N(N-1)}(x_{a,m,0} - x_{p,m,0}) \end{bmatrix}
$$

$$
\cdot \begin{bmatrix} D'_{01}(x_{a,k,0} - x_{p,k,0}) & D'_{0(N-1)}(x_{a,k,0} - x_{p,k,0}) \\ D'_{N1}(x_{a,k,N} - x_{p,k,0}) & D'_{N(N-1)}(x_{a,k,N} - x_{p,k,0}) \end{bmatrix}^{-1}
$$

$$
\cdot \begin{bmatrix} \dot{x}_{a,k,0} - \sum_{\substack{j=0, j\neq 1 \\ j\neq N-1}}^{N} D'_{0j}(x_{a,k,0} - x_{p,k,0}) v_j - \sum_{l=0}^{n_{cp}} \dot{B}_{l,d}(t_0) P_{k,l} \\ \dot{x}_{a,k,N} - \sum_{\substack{j=0, j\neq 1 \\ j\neq N-1}}^{N} D'_{Nj}(x_{a,k,0} - x_{p,k,0}) v_j - \sum_{i=0}^{n_{cp}} \dot{B}_{l,d}(t_N) P_{k,l} \end{bmatrix} \tag{48}
$$

Optimization Algorithm

After all of the boundary conditions have been incorporated to solve the certain PCPs and control points, all of the equality constraints are eliminated. Now, Eq. (42) is equivalent to solving,

$$
\textbf{(Problem 8)} \quad \min_{\boldsymbol{P}', \boldsymbol{v}', t} \quad J = \varphi[\boldsymbol{P}'_N, \boldsymbol{v}'_N, t_f] + \frac{t_f - t_0}{2} \sum_{k=0}^{N} L(\boldsymbol{P}'_{j,k}, \boldsymbol{v}'_k, t) w_k, \; j = 1, \ldots, n_a \tag{49}
$$

$$
\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}', \boldsymbol{v}', t) \leq 0
$$

where, $\boldsymbol{P}'$, $\boldsymbol{v}'$ and $t$ are considered as optimizable parameters and can be solved by nonlinear programming methods. Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ under three boundary conditions are summarized in Table 2.

Table 2 Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ in the CATD Based Optimization Procedure

| **BC** | **CATD** ( $P_{j,0} \neq x_{a,j,0}, j = 1,...,n_a$ ) |
|---|---|
| 1 | $v_i, i = 1,...,N$ , |
| | $P_{j,i}, i = 0,...,n_{cp} - 1, j = 1,...,n_a$ |
| 2 | $v_i, i = 2,...,N$ , |
| | $P_{k,i}, i = 0,...,n_{cp} - 1, k \in [1, n_a]$ |
| | $P_{m,i}, i = 0,2,...,n_{cp} - 1, m = 1,...,n_a, m \neq k$ |
| 3 | $v_i, i = 2,...,N-1$ , |
| | $P_{k,i}, i = 0,...,n_{cp} - 1, k \in [1, n_a]$ |
| | $P_{m,i}, i = 0,2,...,n_{cp} - 2, m = 1,...,n_a, m \neq k$ |

**Remark 6:** The reason to optimize $P_{k,0}$ even though it can be any value to satisfy Eq. (10) is that it will affect the speed of the virtual prey and will further affect the speed of the vehicle.

## LP based Trajectory Optimization

### Problem Formulation

When a vehicle (e.g. MAV, UAV, ground robot, etc.) is assumed as a predator which chases a virtual prey by the LP strategy, its trajectory will be described by a virtual prey ahead of it. Since the prey is a virtual one when using the methods proposed in this research, the value of $\Delta$ will not affect the solution and thus can be regarded as zero. Therefore the LP strategy can be modified as

$$\dot{\boldsymbol{x}}_a(t) + v\boldsymbol{x}_a(t) = v\boldsymbol{x}_p(t) \tag{50}$$

Then, the higher order derivatives of $\boldsymbol{x}_a$, such as the second derivatives, can be represented as,

$$\ddot{\boldsymbol{x}}_a(t) = \dot{v}\left[\boldsymbol{x}_p(t) - \boldsymbol{x}_a(t)\right] + v\left[\dot{\boldsymbol{x}}_p(t) - \dot{\boldsymbol{x}}_a(t)\right] \tag{51}$$

After discretization, we can get following equations,

$$\boldsymbol{\zeta}_{a,i} = (D^{'} + \Xi)^{-1}\Xi\boldsymbol{\zeta}_{p,i} \tag{52}$$

$$\dot{\boldsymbol{\zeta}}_{a,j} = \left(D^{'} + \Xi\right)^{-1}\left[\dot{\Xi} - \dot{\Xi}(D^{'} + \Xi)^{-1}\Xi\right]\boldsymbol{\zeta}_{p,j} + \left(D^{'} + \Xi\right)^{-1}\Xi\dot{\boldsymbol{\zeta}}_{p,j} \tag{53}$$

And

$$\begin{aligned}\ddot{\boldsymbol{\zeta}}_{a,j} = (D^{'} + \Xi)^{-1}&\left\{\begin{matrix}\ddot{\Xi} - \ddot{\Xi}(D^{'} + \Xi)^{-1}\Xi \\ -2\dot{\Xi}\left(D^{'} + \Xi\right)^{-1}\left[\dot{\Xi} - \dot{\Xi}(D^{'} + \Xi)^{-1}\Xi\right]\end{matrix}\right\}\boldsymbol{\zeta}_{p,j} \\ &+2(D^{'} + \Xi)^{-1}\left[\dot{\Xi} - \dot{\Xi}\left(D^{'} + \Xi\right)^{-1}\Xi\right]\dot{\boldsymbol{\zeta}}_{p,j} + (D^{'} + \Xi)^{-1}\Xi\ddot{\boldsymbol{\zeta}}_{p,j}\end{aligned} \tag{54}$$

Different from VMC and CATD method, it is convenient to use inverse dynamics method to determine $\boldsymbol{x}_{sr}$ and control variables in terms of $v$ and $\boldsymbol{x}_p$.

Now, equivalent with **Problem 2**, trajectory optimization problem by using LP method is to solve,

$$\begin{aligned}&\min_{\boldsymbol{x}_p, \boldsymbol{v}, t_f} \quad J = \varphi[\boldsymbol{x}_p, \boldsymbol{v}, t_f] + \int_{t_0}^{t_f} L(\boldsymbol{x}_p, \boldsymbol{v}, t)dt \\ &\textbf{(Problem 9)} \ \text{Subject to} \quad \boldsymbol{g}(\boldsymbol{x}_p, \boldsymbol{v}, t) \le 0 \\ &\qquad\qquad\qquad\quad \boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{v}, t) = 0\end{aligned} \tag{55}$$

Note that similarly as MC rule, now equality constraints $\boldsymbol{h}'(\boldsymbol{x}_p, \boldsymbol{x}_r, \boldsymbol{v}, t) = 0$ only include boundary conditions, because the dynamic equation of the vehicle has been already considered when calculating states and control variables. If combine with Eq. (17)), solving Eq. (13) will be equivalent to solving **Problem 10**,

$$\min_{\boldsymbol{P}_j,\boldsymbol{v},t} \quad J = J = \varphi[\boldsymbol{P}_N, \boldsymbol{v}_N, t_f] + \frac{t_f - t_0}{2}\sum_{k=0}^{N} L(\boldsymbol{P}_{j,k}, \boldsymbol{v}_k, t) w_k \quad , j = 1,...,n_a, \quad j = 1,...,n_a$$
$$\text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}_j, \boldsymbol{v}, t) \le 0 \qquad (56)$$
$$\boldsymbol{h}'(\boldsymbol{P}_j, \boldsymbol{v}, t) = 0$$

where, $\boldsymbol{P}_j^T = [P_{j,0},...P_{j,n_{cp}}]$, and $\boldsymbol{v}^T = [v_0,...,v_N]$.

Boundary Conditions Incorporation

This section will talk about how to incorporate the three boundary conditions (illustrated in the **Remark 1**) by using LP motion rule.

**Lemma 8:** For BC 1, the "position state" of predator (i.e. the actual vehicle) at all discretized nodes except the first and last ones are calculated using

$$\boldsymbol{\zeta}_{a,i}^{\odot} = \left(M_{LP}^{\odot}\right)^{+}\left(\Xi B \boldsymbol{P}_i - M_{LP,1}\zeta_{a,i,0} - M_{LP,N+1}\zeta_{a,i,N}\right), \qquad i = 1,...,n_a \qquad (57)$$

in which $M_{LP} \triangleq (D' + \Xi)$, and $M_{LP,1}$, $M_{LP,N+1}$, and $M_{LP}^{\odot}$ are the first, last, and remaining columns of $M_{LP}$, respectively. $\zeta_{a,i,0}$, $\zeta_{a,i,N}$, and $\boldsymbol{\zeta}_{a,i}^{\odot}$ are respectively the first, last, and remaining "position state" variables of the vehicle in the $i^{th}$ direction. $(\ )^{+}$ denotes the pseudo-inverse operation.

**Proof.** According to LP strategy, $\boldsymbol{\zeta}_{a,i} = (D' + \Xi)^{-1}\Xi\boldsymbol{\zeta}_{p,i}$, $i = 1,...,n_a$. Let us define $M_{LP} \triangleq (D' + \Xi)$. For BC 1, the LP strategy $M_{LP}\boldsymbol{\zeta}_{a,i} = \Xi\boldsymbol{\zeta}_{p,i} = \Xi B\boldsymbol{P}_i$ can be re-arranged as $M_{LP,1}x_{a,i,0} + M_{LP}^{\odot}\boldsymbol{x}_{a,i}^{\odot} + M_{LP,N+1}x_{a,i,N} = \Xi B\boldsymbol{P}_i$, which further can be rearranged as Eq. (57).

**Lemma 9:** For BC 2, $P_{j,0} = x_{a,j,0} + \dot{x}_{a,j,0}/v_0$, $j = 1,...,n_a$, and the vehicle's "position state" at all the discretized nodes, except the first and last nodes will be calculated using Eq. (57).

**Proof.** The vehicle's "position state" at all the discretized nodes, except the first and last nodes, will be calculated using Eq. (57). According to the initial condition in LP,

$x_{p,j,0} = x_{a,j,0} + \dot{x}_{a,j,0} / v_0$, and Property 1 in the B-spline representation,

$P_{i,0} = x_{p,i,0} = x_{a,i,0} + \dot{x}_{a,i,0} / v_0,\ j = 1,...,n_a$.

**Lemma 10:** For BC 3 (LP), the vehicle's "position state" at all the discretized nodes, except the first and last nodes, will be calculated using Eq. (57), $P_{j,0} = x_{a,j,0} + \dot{x}_{a,j,0} / v_0$, and

$P_{j,N} = x_{a,j,N} + \dot{x}_{a,j,N} / v_N,\ j = 1,...,n_a$.

**Proof.** The proof is the same as that of Lemma 9, thus it is not listed here.

Optimization Algorithm

After all of the boundary conditions have been incorporated to solve the certain PCPs and control points, all of the equality constraints are eliminated. Now, Eq. (56) is equivalent to solving,

$$\textbf{(Problem 11)} \quad \min_{\boldsymbol{P}',\boldsymbol{v}',t} \ J = \varphi[\boldsymbol{P}'_N, \boldsymbol{v}'_N, t_f] + \frac{t_f - t_0}{2} \sum_{k=0}^{N} L(\boldsymbol{P}'_{j,k}, \boldsymbol{v}'_k, t) w_k,\ j = 1,...,n_a \quad \text{Subject to} \quad \boldsymbol{g}(\boldsymbol{P}', x\boldsymbol{v}', t) \le 0 \tag{58}$$

where, $\boldsymbol{P}'$, $\boldsymbol{v}'$ and $t_f$ are considered as optimizable parameters and can be solved by nonlinear programming methods. Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ under three boundary conditions are summarized in Table 3.

Table 3 Parameters $\boldsymbol{v}'$ and $\boldsymbol{P}'$ in the LP Based Optimization Procedure

| BC | LP motion rule |
|---|---|
| 1 | $v_i, i = 0,...,N$, $P_{j,i}, i = 0,...,n_{cp}, j = 1,...,n_a$ |

| BC | LP motion rule |
|---|---|
| 2 | $v_i, i=1,...,N,\ P_{j,i}, i=1,...,n_{cp}, j=1,...,n_a$ |
| 3 | $v_i, i=1,...,N-1,$ |
| | $P_{j,i}, i=1,...,n_{cp}$ -1, $j=1,...,n_a$ |

Summary

Unified Linear Algebraic

A unified linear algebraic (ULA) formulation, describing the relationship between the positions of predator and prey, is now summarized for three motion strategies (MC, CATD and LP) in this section.

The positions of predator $\boldsymbol{x}_a$ (i.e. the actual vehicle) and prey $\boldsymbol{x}_p \in \Re^{n_a \times 1}$ (a virtual one) are discretized using high order discretization methods, for example the Legendre-Gauss-Lobatto (LGL) method [49], as $\boldsymbol{\zeta}_{a,j}^T = \left[ x_{a,j,0},..., x_{a,j,N} \right]$ and $\boldsymbol{\zeta}_{p,j}^T = \left[ x_{p,j,0},..., x_{p,j,N} \right]$, where $j=1,...,n_a$ is the $j^{th}$ direction of the "position state" and the third subscript $0,...,N$ denotes the discretization node. Superscript "T" means the transpose operation.

With the above defined notations, the following Lemma about the ULA formulation is proposed.

**Lemma 11:** The "position state" of predator (i.e. the actual vehicle) in the MC, CATD, and LP motion strategies is abstracted as

$$\boldsymbol{\zeta}_{a,j} = A_p(\boldsymbol{v}, \boldsymbol{\xi}_j)\boldsymbol{\zeta}_{p,j} + A_b(\boldsymbol{v})\boldsymbol{\xi}_j,\ j=1,...,n_a \tag{59}$$

The detailed representations of matrices $A_p$ and $A_b$ in Eq. (59) depend on the chosen motion strategies and will be later introduced. In Eq. (59), $\boldsymbol{v} = [v_0, v_1,..., v_N]^T$ is the discretized

version of a one-dimensional path control parameter (PCP) vector $v(t)$. $\boldsymbol{\xi}_j$ represents a set of optimizable constants with a finite dimension that are used in the MC and CATD motion strategies.

The first derivative of predator's "position state" is

$$\dot{\boldsymbol{\zeta}}_{a,j} = \dot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)\boldsymbol{\zeta}_{p,j} + A_p(\boldsymbol{v},\boldsymbol{\xi}_j)\dot{\boldsymbol{\zeta}}_{p,j} + \dot{A}_b(\boldsymbol{v})\boldsymbol{\xi}_j,\ \ j = 1,...,n_a \tag{60}$$

while the second derivative is

$$\begin{aligned}\ddot{\boldsymbol{\zeta}}_{a,j} = {} & \ddot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)\boldsymbol{\zeta}_{p,j} + 2\dot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)\dot{\boldsymbol{\zeta}}_{p,j} \\ & + A_p(\boldsymbol{v},\boldsymbol{\xi}_j)\ddot{\boldsymbol{\zeta}}_{p,j} + \ddot{A}_b(\boldsymbol{v})\boldsymbol{\xi}_j\end{aligned},\ \ j = 1,...,n_a \tag{61}$$

The detailed descriptions of ULA formulation for the MC, CATD, and LP strategies (i.e. $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)$, $A_b(\boldsymbol{v})$, and $\boldsymbol{\xi}_j$) are summarized in Table 4. In the table, the PCP variable matrix $\Xi = diag\{\boldsymbol{v}\}$ is a diagonal matrix. $I$ is an identity matrix. $D$ is the differentiation matrix in the LGL discretization, and $D' \triangleq \left[2/(t_f - t_0)\right]D$ [49].

Table 4 ULA in Different Motion Strategies

| Motion strategies | ULA | $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)$ | $A_b(\boldsymbol{v})$ | $\boldsymbol{\xi}_j$ |
|---|---|---|---|---|
| MC | $\boldsymbol{\zeta}_{a,j} = \Xi\boldsymbol{\zeta}_{p,j} + \left(I_{N+1} - \Xi\right)\boldsymbol{\xi}_j$ | $\Xi$ | $I_{N+1} - \Xi$ | $x_{r,j}\boldsymbol{1}$ |
| CATD | $\boldsymbol{\zeta}_{a,j} = I_{N+1}\boldsymbol{\zeta}_{p,j} + \Xi\boldsymbol{\xi}_j$ | $I_{N+1}$ | $\Xi$ | $\boldsymbol{\zeta}_{a,j,0} - \boldsymbol{\zeta}_{p,j,0}$ |
| LP | $\boldsymbol{\zeta}_{a,j} = \left[(D' + \Xi)^{-1}\Xi\right]\boldsymbol{\zeta}_{p,j}$ | $(D' + \Xi)^{-1}\Xi$ | 0 | 0 |

The detail proof of Table 4 is shown below.

**Lemma 12:** In MC, $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=\Xi$, $A_b(\boldsymbol{v})=I_{N+1}-\Xi$, and $\boldsymbol{\xi}_j=x_{r,j}\boldsymbol{1}$, $j=1,...,n_a$. $\boldsymbol{1}\in\Re^{(N+1)\times 1}$ is a column vector with all the elements equal to one, and $x_{r,j}$ is the $j^{th}$ component of the reference point.

**Proof**. Comparing Eq. (19) and Eq. (59), $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=\Xi$, $A_b(\boldsymbol{v})=I_{N+1}-\Xi$, and $\boldsymbol{\xi}_j=x_{r,j}\boldsymbol{1}$, $j=1,...,n_a$.

**Remark 7:** Following the **Lemma 12**, when the MC strategy is applied, the derivatives used in Eqs. (60)-(61) are $\dot{A}_p(\boldsymbol{v})=\dot{\Xi}$, $\ddot{A}_p(\boldsymbol{v})=\ddot{\Xi}$, $\dot{A}_b(\boldsymbol{v})=-\dot{\Xi}$, $\ddot{A}_b(\boldsymbol{v})=-\ddot{\Xi}$, and so on.

**Lemma 13:** In the CATD strategy observed in bats and dragonflies, $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=I_{N+1}$, $A_b(\boldsymbol{v})=\Xi$, and $\boldsymbol{\xi}_j=\boldsymbol{\zeta}_{a,j,0}-\boldsymbol{\zeta}_{p,j,0}$, in which $\boldsymbol{\zeta}_{a,j,0}=x_{a,j}(t_0)\boldsymbol{1}$ and $\boldsymbol{\zeta}_{p,j,0}=x_{p,j}(t_0)\boldsymbol{1}$.

Proof: Comparing Eq. (36) and Eq. (59), $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=I_{N+1}$ and $A_b(\boldsymbol{v})=\Xi$.

**Remark 8:** Following Lemma 13, when the CATD strategy is applied, the derivatives used in Eq.(60)-(61) are $\dot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)=0$, $\ddot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)=0$, $\dot{A}_b(\boldsymbol{v})=\dot{\Xi}$, $\ddot{A}_b(\boldsymbol{v})=\ddot{\Xi}$, and so on.

**Lemma 14:** In the LP strategy, $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=(D^{'}+\Xi)^{-1}\Xi$ and $A_b(\boldsymbol{v})=0$.

**Proof**: According to Eq. (52) and Eq. (59), $\boldsymbol{\zeta}_{a,i}=(D^{'}+\Xi)^{-1}\Xi\boldsymbol{\zeta}_{p,i}$, $A_p(\boldsymbol{v},\boldsymbol{\xi}_j)=(D^{'}+\Xi)^{-1}\Xi$, and $A_b(\boldsymbol{v})=0$.

**Remark 9:** Due to Eqs. (53)-(54) and Eq. (59),

$$\dot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)=\left(D^{'}+\Xi\right)^{-1}\left[\dot{\Xi}-\dot{\Xi}(D^{'}+\Xi)^{-1}\Xi\right] \tag{62}$$

and,

$$\ddot{A}_p(\boldsymbol{v},\boldsymbol{\xi}_j)=(D^{'}+\Xi)^{-1}\begin{Bmatrix}\ddot{\Xi}-\ddot{\Xi}(D^{'}+\Xi)^{-1}\Xi\\ -2\dot{\Xi}\left(D^{'}+\Xi\right)^{-1}\left[\dot{\Xi}-\dot{\Xi}(D^{'}+\Xi)^{-1}\Xi\right]\end{Bmatrix} \tag{63}$$

Framework of Bio-inspired Trajectory Optimization Method

The procedure of Bio-inspired Trajectory Optimization Method will be summarized in this part.

**Assumption 1:** In P1 (Defined in 2.1), the mapping from $(\boldsymbol{x}_a,t)$ to $(\boldsymbol{x}_{sr},\boldsymbol{u})$ is assumed to be injective [75], that is, $\boldsymbol{u}$ and $\boldsymbol{x}_{sr}$ can be solved using $\boldsymbol{x}_a$ explicitly.

The "position state" $\boldsymbol{x}_a\in\Re^{n_a\times 1}$ in **Problem 1** (P1) is intentionally limited in an iteratively refined subspace (or manifold) defined by the ULA formulation (Eq. (59)). As mentioned in Assumption 1, all the control and state variables in P1 are functions of $v(t)$, $\boldsymbol{x}_p(t)$, and possibly $\boldsymbol{\xi}$ and $t_f$. Thus, the performance index to be optimized is $J_2=J_2(\boldsymbol{v}',\boldsymbol{P}',\boldsymbol{\xi},t_f)$ now. The I. E. C. is $\boldsymbol{g}_2(\boldsymbol{v}',\boldsymbol{P}',\boldsymbol{\xi},t_f)\le 0$. (P2) Here $J_2$ and $\boldsymbol{g}_2$ are discretized versions.

The steps involved in BiCF are summarized in **Algorithm 1**.

**Algorithm 1 : BiCF**

| | | |
|---|---|---|
| Steps in the Initialization | Step 0: | Select one of the three motion strategies: MC, CATD, or LP. |
| | Step 1: | Provide related initial guesses of optimizable parameters for certain strategy, e.g. PCPs, reference point, control points of virtual prey motion. |
| Steps inside the NLP | Step 2: | Calculate the remaining parameters by using the appropriate necessary conditions discussed in Section 3.3. |
| | Step 3: | Construct the virtual prey motion using B-splines (Eq. (14)). |
| | Step 4: | Compute the predator (vehicle's) motion by using predator-prey relations discussed in Section 3.1 |
| | Step 5: | Calculate the control variables by using the dynamics equation of vehicle. |
| | Step 6: | Check the performance index |

| Algorithm 1 : BiCF | | |
|---|---|---|
| | Step 7: | Check the inequality constraints |
| | Step 8: | Generate a new initial guess of optimizable parameters and go back to Step 2, if the maximum number of iterations and the convergence criterion are not reached. Otherwise, the optimization is terminated. |

Optimization and Dimension Analysis

**Theorem 1:** The solution of **Problem 1 (P1)** is equal to the limiting solution of **Problem 2 (P2)** obtained through the procedure described in Algorithm 1, which is optimal as the number of discretized nodes $N$ and the number of control points $n_{cp}$ approach $\infty$.

**Proof**. In **P1**, the continuous or piecewise continuous state $\boldsymbol{x}$ and control $\boldsymbol{u}$, and possibly the final time $t_f$ are solved to minimize the performance index $J_1 = J_1(\boldsymbol{x}, \boldsymbol{u}, t_f)$, subject to the E.C.s. $\boldsymbol{h}_1(\boldsymbol{x}, \boldsymbol{u}, t) = 0$ and the I.E.C.s. $\boldsymbol{g}_1(\boldsymbol{x}, \boldsymbol{u}, t) \le 0$.

Based on Assumption 1, $\boldsymbol{x}_{sr}$ and $\boldsymbol{u}$ can be calculated using $\boldsymbol{x}_a$. Through Step 4 of Algorithm 1, the state vector $\boldsymbol{x}_a$ is represented in the ULA formulation (1) as a function of the PCPs $v(t)$, $\boldsymbol{x}_p$ (the "position state" of the virtual prey), and $\boldsymbol{\xi}$ (the finite dimensional set of constant but optimizable parameters). The virtual prey, PCPs, and possibly final time span over the full search space of P1. Therefore, solving **P1** is in fact to solve problem **P3**: minimizing $J_3 = J_3(\boldsymbol{x}_p, v, \boldsymbol{\xi}, t_f)$ subject to $\boldsymbol{h}_3(\boldsymbol{x}_p, v, \boldsymbol{\xi}, t_f) = 0$ and $\boldsymbol{g}_3(\boldsymbol{x}_p, v, \boldsymbol{\xi}, t_f) \le 0$. Here $\boldsymbol{\xi}$ includes all $\boldsymbol{\xi}_j$, $\boldsymbol{j} = 1, ..., n_a$.

In Step 3 of Algorithm 1, since the solution is assumed to be piece-wise differentiable, the virtual prey motion can be represented by the B-spline. Then the system is evaluated at the

discretized node. As shown in [52], solving **P3** is approximately equivalent to solving the following problems P4 as the numbers of discretized nodes and the control points increase. **P4**: solving the control points $\boldsymbol{P}$, the PCP $\boldsymbol{v}$, and possibly $\boldsymbol{\xi}$ and $t_f$ to minimize $J_4 = J_4(\boldsymbol{v}, \boldsymbol{P}, \boldsymbol{\xi}, t_f)$, subject to $\boldsymbol{h}_4(\boldsymbol{v}, \boldsymbol{P}, \boldsymbol{\xi}, t_f) = 0$ and $\boldsymbol{g}_4(\boldsymbol{v}, \boldsymbol{P}, \boldsymbol{\xi}, t_f) \le 0$.

Following Step 2 of Algorithm 1, certain control points and PCP nodes in Set $S_C$ can be calculated instead of being optimized using the necessary conditions derived based on the boundary conditions; therefore, solving **P4** is equivalent to solving **P2**: minimize $J_2 = J_2(\boldsymbol{v}', \boldsymbol{P}', \boldsymbol{\xi}, t_f)$, subject to $\boldsymbol{g}_2(\boldsymbol{v}', \boldsymbol{P}', \boldsymbol{\xi}, t_f) \le 0$. The nonlinear dynamics have already been considered when solving the "state rate" and control variables. The boundary conditions are considered in the necessary conditions (Step 2 of Algorithm 1).

Based on the above discussions, **P1** to **P3** and **P4** to **P2** are equivalent. **P3** to **P4** is approximately equivalent as described next.

We assume that the optimal solution achieved in **P4** is $\{\boldsymbol{P}^*, \boldsymbol{v}^*, \boldsymbol{\xi}^*, t_f^*\}_N^{n_{cp}}$ with $N$ discretized nodes and $n_{cp}$ control points, and the superscript "*" denotes the optimum. As $N$ and $n_{cp}$ increase, a sequence of optimal solutions can be achieved in **P4**. As proven in [52], if there exists an optimal solution $\{\boldsymbol{x}_p^*, v^*, \boldsymbol{\xi}^*, t_f^*\}$ for the continuous version of **P2**, the following limits converge uniformly for $\lim_{\substack{N\to\infty \\ n_{cp}\to\infty}} \{\boldsymbol{x}_p^* - (B\boldsymbol{P}^*)_N^{n_{cp}}\} = 0$, $\lim_{N\to\infty} \{v(t)^* - (\boldsymbol{v}^*)_N\} = 0$, $\lim_{N\to\infty} \{\boldsymbol{\xi}^* - (\boldsymbol{\xi}^*)_N\} = 0$, and $\lim_{N\to\infty} \{t_f^* - (t_f^*)_N\} = 0$, and the performance indices equal each other. Here $()_N^{n_{cp}}$ denotes the solution when $N$ and $n_{cp}$ are used, and $()_N$ is the case when $N$ is used.

Therefore, the limiting trajectory in solving **P4** is equivalent to solving P3 as the numbers of discretized nodes and the control points increase to infinite. The parameters to be optimized here $\{\boldsymbol{P}^*, \boldsymbol{v}^*, \boldsymbol{\xi}^*, t_f^*\}_N^{n_{cp}}$ are different from those of [52], however, the procedure of the proof is the same.

**Remark 10:** It is not practical to choose an infinite value for the number of the control points and the discretized nodes. Also in many cases, as the number of optimizable parameters increases in the NLP, the convergence gets more difficult. Therefore, a tradeoff exists among the solution optimality, convergence rate, and computational cost.

The problem dimension of the achieved NLPs is compared to the other two methods as shown in **Table 5**. The first approach is a baseline method (BL), in which the "position state" in **P1** is discretized using popular pseudospectral collocation methods such as the methods in [49]. The second approach is the B-spline based (BS) method [46], in which the "position state" in **P1** is directly represented by the B-spline curves. The third approach is the proposed BiCF methods. To have a fair comparison, the differential inclusion procedure is used in all these approaches such that the control and "state rate" variables are calculated by the "position" state variables.

The optimizable parameters in the BL approach is on the order of $\mathrm{O}(n_a N)$. The dimension of the BS approach is on the order of $\mathrm{O}(n_a n_{cp1})$, in which $n_{cp1}+1$ is the number of control points used in representing the "position state" $\boldsymbol{x}_a \in \Re^{n_a \times 1}$. The number of optimizable parameters in BiCF is on the order of $\mathrm{O}(N + n_a n_{cp})$.

Typically $n_{cp}$ and $n_{cp1}$ are much smaller than $N$, therefore the BS based method and the

methods in BiCF have smaller dimensions. Here the E.C.s related to the dynamic equations have been eliminated. Furthermore, in BiCF the E.C.s about the boundary conditions can be eliminated via the necessary conditions.

Table 5 Dimension Comparisons

| | Baseline Approach | B-spline Approach | Methods in BiCF |
|---|---|---|---|
| Parameters | $O(n_a N)$ | $O(n_a n_{cp1})$ | $O(N + n_a n_{cp})$ |

The problem dimension comparisons between the BS based approach and the proposed BiCF are further discussed in the following two cases.

**Case 1:** if there is a small number of (or no) obstacles or only a few constraints in **P1**, small values of $n_{cp1}$ and $n_{cp}$ are flexible enough to represent the trajectory.

**Case 2:** If there are many obstacles and/or severe constraints, the proposed methods in BiCF only need a lower degree curve and fewer control points as compared with the BS based method.

**Explanation 1:** Based on Eq. (15) and Eq. (59), the $j^{th}$ component of the "position state" evaluated at node k can be approximated as

$$\begin{aligned} x_{a,j,k} &= \sum_{m=0}^{N} A_{p,k,m}(\boldsymbol{v}, \boldsymbol{\xi}_j) x_{p,j,m} + \sum_{m=0}^{N} A_{b,k,m}(\boldsymbol{v}) \boldsymbol{\xi}_{j,m} \\ &= \sum_{m=0}^{N} \left( \sum_{i=0}^{n_{cp}} A_{p,k,m}(\boldsymbol{v}, \boldsymbol{\xi}_j) B_{i,d}(t_m) \right) P_{j,i} + \sum_{m=0}^{N} A_{b,k,m}(\boldsymbol{v}) \boldsymbol{\xi}_{j,m} \end{aligned} \tag{64}$$

while in a typical BS based approach, the "position state" is represented by [46].

$$x_{a,j,k} = \sum_{i=0}^{n_{cp1}} B_{i,d}(t_k) P_{j,i} \tag{65}$$

In Eq. (64), $A_{p,k,m}$ is the ($k$, $m$) element of matrix $A_p$. It can be seen in Eq. (64) that any control point $P_{j,i}, j=1,...,n_a, i=0,...,n_{cp}$ has been used $(n_{cp}+1)(N+1)$ times to form $x_{a,j,k}$ in the full space, while in Eq. (65) $P_{j,i}, j=1,...,n_a, i=0,...,n_{cp1}$ is only used $n_{cp1}+1$ times. Therefore under the same obstacles or constraints scenarios, Eq. (64) needs a much smaller $n_{cp}$ than $n_{cp1}$ in Eq. (65) to achieve the same curve flexibility condition.

**Explanation 2:** When there are many obstacles and severe states and control variable constraints, the B-spline curve requires a larger number of control points $n_{cp1}$ and a higher degree curve to avoid those obstacles and constraints. In comparison, the proposed method in BiCF only needs a lower degree curve and fewer control points, because the obstacles can be avoided by varying the PCP values.

Comparison of Bio-inspired Trajectory Optimization Method

In this part, MC, CATD, LP, BS and BL methods will be compared through a robot minimum arrival time example.

The minimum time trajectory is solved for a mobile robot to move from an initial position to another position, while avoiding collisions with many obstacles. The discretized performance index is $J=0.5\left(t_f-t_0\right)\sum_{i=0}^{N}\omega_i$, in which $\omega_i$ is the weight associated with the discretization node. The nonlinear dynamic model of the robot [76] is assumed to be

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\cos\theta\\ \sin\theta\\ 0\end{bmatrix}\upsilon+\begin{bmatrix}0\\ 0\\ 1\end{bmatrix}\omega \tag{66}$$

where the "position state" is the midpoint of the vehicle $[x, y]^T$. The "state rate" is the direction of the vehicle $\theta$, which is calculated using $\theta = \tan^{-1}(\dot{y}/\dot{x})$. $v$ and $\omega$, the speed and angular velocity of vehicle, are regarded as the control variables and computed by

$$v = \begin{cases} \dot{x}/\cos(\theta), \cos(\theta) \neq 0 \\ \dot{y}/\sin(\theta), \cos(\theta) = 0 \end{cases} \tag{67}$$

and

$$\omega = (\ddot{y}\dot{x} - \ddot{x}\dot{y})/(\dot{x}^2 + \dot{y}^2), \ (\dot{x}^2 + \dot{y}^2) \neq 0 \tag{68}$$

To compare the robustness of algorithms and solution optimality, a Monte Carlo simulation is used. In each Monte Carlo run, the same but randomly generated scenario is used in all the five methods. A total of 4,000 runs are used in three different computers.

The testing area is defined to be $11\ m$ in both *x* and *y* directions. Ten to twelve obstacles with a radius varying uniformly within $[0.4, 0.8]m$ are generated randomly in the testing area. The initial position of the robot is uniformly distributed in the lower left corner of the test area (i.e. $[0,1]m$ in the *x* position and $[1,5]m$ in the *y* position, respectively). Accordingly, the final position of the robot is uniformly distributed in the upper right part of the test area (i.e. $[9,10]m$ in the *x* position and $[6,10]m$ in the *y* position, respectively). Without the loss of generality, the robot is initially heading towards the final position, and the initial speed is set to be $0.1m/s$. The robot has a maximum speed of $V_{\max} = 0.1m/s$ and a maximum turning rate of $135°/s$, respectively.

To improve the success rate, particularly for the BL method, an obstacle-free path is generated by the wavefront algorithm [77] without considering the nonlinear dynamics, and state

and control variable constraints.

Twenty one discretization nodes are used in these five optimization methods. $n_{cp} = 5$ and $d = 3$ are selected for the MC, LP, CATD and BS methods. The PCPs in the MC, LP and CATD methods are initially guessed to be 0.998, 1.1 and 0.001, respectively. In the CATD method, the start point of the virtual prey is randomly guessed to be $[-0.1, -0.1]m$, away from the initial position of the actual robot. In the MC method, the reference point is randomly set to be $[-120, -120]m$. The initial guess of the final time is

$$t_f = \left\| \boldsymbol{x}_a(t_f) - \boldsymbol{x}_a(t_0) \right\| / V_{\max} \tag{69}$$

In any of these methods, if a converged solution is not achieved within 100 iterations or 5000 function evaluations, a failure case is counted against this method. Additionally, the minimum performance index among all five methods is regarded as the best solution. If the performance index from any other method is within 5% difference of the best solution, the solution found using this method is regarded as an optimal one; otherwise it will be regarded as a feasible solution only.

The following observations are obvious from Figure 4. (i) The BL method has the highest failure rate among five methods. As analyzed in Section 3.5, the problem dimension achieved using the BL method is the highest, which practically makes the convergence more challenging. (ii) All the other four methods have a similar success rate and all are above 90%. (iii) All the BiCF methods have higher successful rates in finding optimal solutions as compared with the BS method. (iv) LP strategy has a relatively higher success rate than MC and CATD strategies. Since when using a small number of control points and a lower degree, the BS method is not flexible enough as compared with the BiCF methods.

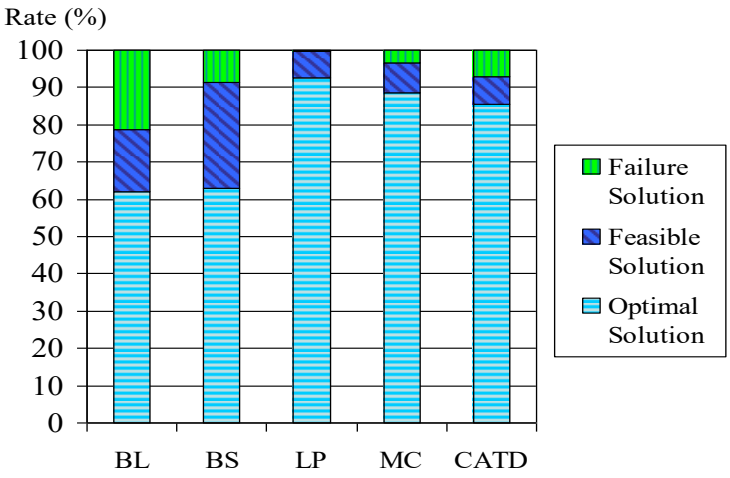


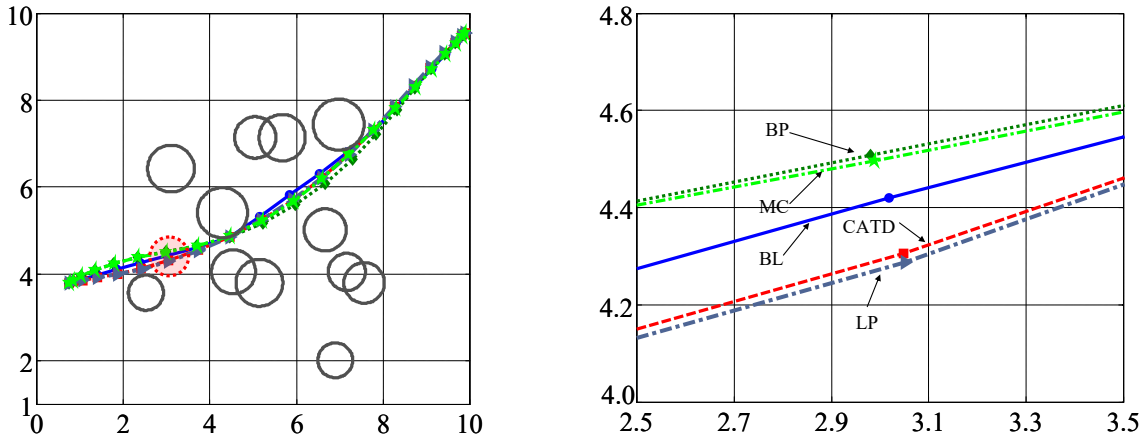

Figure 4 The failure, feasible, and optimal solution rates for five methods

The average computational cost of the Monte Carlo simulation is summarized in Table 6. The average CPU time of BL method is the longest, and that of BS method is the shortest. The BiCF methods are only a little bit slower than the BS method. However, as shown in Figure 4, the methods in BiCF can achieve optimal solutions with a much higher success rate as compared with that of the BS method.

Table 6 Average CPU Time of Monte Carlo Simulation

| Algorithm | Average CPU Time |
|---|---|
| BL | 7.32 s |
| BS | 0.79 s |
| LP | 1.47 s |
| MC | 1.95 s |
| CATD | 1.69 s |

One example Monte Carlo run is shown here. Figure 5 shows the minimum time trajectories generated by five methods with 12 obstacles in the area. In Figure 5(b), the trajectories generated by the five methods are close to each other.




(a) (b)

Figure 5 One Monte Carlo run: (a) optimal trajectories found by five methods, (b) zoomed in for the dotted area in (a)

As listed in Table 7, the performance index of BL method is the smallest but the CPU time is the highest, while the performance indices of three BiCF methods are within 0.3% differences with that of the BL method. The BS method has the lowest computation time but only generates a feasible solution.

Table 7 The CPU Time and the Performance Index for One Run

| Algorithm | CPU Time | Performance Index |
|---|---|---|
| BL | 9.77 s | 110.67 |
| BS | 1.21s | 111.95 |
| LP | 1.43 s | 111.09 |
| MC | 1.57 s | 111.04 |
| CATD | 1.26 s | 111.08 |

## Simulation Examples

Two non-trivial examples are conducted to demonstrate the capabilities of the algorithms in BiCF: a supersonic aircraft minimum time-to-climb problem and a robot obstacle free trajectory planning problem.

All the simulations are conducted in the same computer with a frequency of 3.16 GHz and a RAM of 4 GB, and the MATLAB function "fmincon" (7.12.0.635 R2011a) is used as the NLP solver.

Robot Obstacle Avoidance Trajectory Optimization

In this example, the minimum time trajectory is solved for a two-wheel mobile robot to move from the initial position $[1,1]^T m$ nto the final position $[9,9]^T m$, while avoiding collisions with obstacles. The discretized minimum time performance index is $J = 0.5\left(t_f - t_0\right)\sum_{i=0}^{N}\omega_i$. The model of the two-wheel robot is governed by Eq. (66). The initial heading angle and speed of the vehicle are assumed to be $45°$ and $0.1m/s$, respectively. The control variables are constrained by $|v| \le v_{\max} = 0.1m/s$ and $|\omega| \le \omega_{\max} = 135°/s$. The "state rate" variable is calculated using $\theta = \tan^{-1}(\dot{y}/\dot{x})$.

Two scenarios with different obstacles are simulated to demonstrate the effectiveness of the methods in BiCF. The first case includes three obstacles: $(x-4)^2 + (y-4)^2 = 4$, $(x-6)^2 + (y-7)^2 = 1$, and $(x-8)^2 + (y-6)^2 = 1$. There are four obstacles involved in the second case: $(x-4)^2 + (y-4)^2 = 4$, $(x-7.5)^2 + (y-4)^2 = 1$, $(x-8)^2 + (y-6)^2 = 0.5$, and $(x-7)^2 + (y-8)^2 = 1$. The initial settings for both cases are the same.

A randomly selected curve is used as the trivial guess of the robot trajectory. $n_{cp} = 5$ and $d = 3$ are tuned in the MC and CATD methods, while $n_{cp} = 4$ and $d = 3$ are selected in the LP method. PCPs are guessed to be 1 in the MC method and 0.1 in the other two methods. The

initial position of the virtual prey is randomly guessed to be $[1.01,1.01]m$ in the LP method, and the reference point is randomly set to be $[130,-120]m$ in the CATD method. The final time is arbitrarily assumed to be 130s in all three methods.

The computational cost and achieved performance indices of the three-obstacle case are shown in Table 8. The overall CPU runtimes for the BiCF methods increase as the number of the discretized nodes increases from 10 to 25 nodes; however the increase is small (MC: from 2.91s to 5.44s; CATD: from 2s to 5.30s; and LP: from 2.02s to 5.13s). For example, in the 25-node case, the maximum CPU time among all three methods is only 5.44 seconds. The performance indices are consistent for different node cases. Due to the numerical precision issues, the results are slightly different, but the maximum difference is only 1.59%.

Table 8 Collision Avoidance Problem (3 Obstacles)

| Algorithm | Performance | 10-node | 15-node | 20-node | 25-node |
|---|---|---|---|---|---|
| MC approach | Index (s) | 123.3 | 124.41 | 123.92 | 122.85 |
| | CPU Time (s) | 2.91 | 3.57 | 4.25 | 5.44 |
| LP approach | Index (s) | 123.86 | 124.06 | 124.01 | 123.73 |
| | CPU Time (s) | 2.02 | 2.82 | 3.77 | 5.13 |
| CATD approach | Index (s) | 124.16 | 124.84 | 124.38 | 124.81 |
| | CPU Time (s) | 2.00 | 2.16 | 3.95 | 5.30 |

The results for the 4-obstacle case are shown in Table 9. Analysis is the same as three-obstacle case and omitted here. The performance indices in Table 8 and Table 9 are different due to different number and position of obstacles involved. The four-obstacle cases generally take a little bit longer CPU time to find the optimal trajectories.

Table 9 Collision Avoidance Problem (4 Obstacles)

| Algorithm | Performance | 10-node | 15-node | 20-node | 25-node |
|---|---|---|---|---|---|
| MC approach | Index (s) | 122.34 | 123.66 | 122.66 | 122.41 |
| | CPU Time (s) | 2.58 | 4.14 | 4.25 | 5.59 |
| LP approach | Index (s) | 122.81 | 122.66 | 122.85 | 123.24 |
| Algorithm | Performance | 10-node | 15-node | 20-node | 25-node |
| LP approach | CPU Time (s) | 2.10 | 3.91 | 4.95 | 5.17 |
| CATD approach | Index (s) | 122.82 | 123.17 | 122.85 | 122.34 |
| | CPU Time (s) | 2.44 | 3.4 | 4.42 | 5.60 |

The minimum time trajectories for the 3-obstacle and 4-obstacle cases (25 nodes) are shown in Figure 6-Figure 7. It is worth noting that the trajectories obtained via these three methods are slightly different, because the minimum times achieved in the different motion strategies are slightly different.

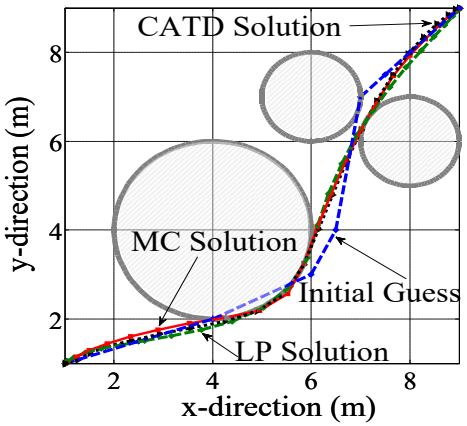



Figure 6 Optimal trajectories for the 25-node case with 3 obstacles

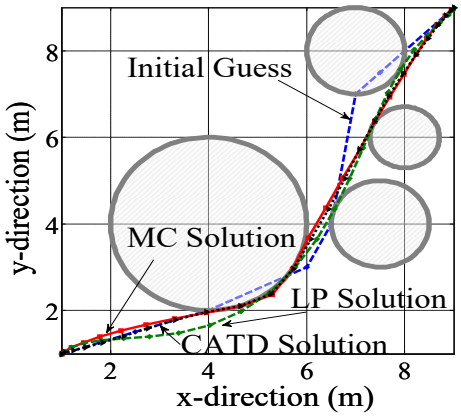



Figure 7 Optimal trajectories for the 25-node case with 4 obstacles

Supersonic Aircraft Minimum Time-to-Climb Problem

The aircraft is assumed to be a point mass and governed by the following 3-DOF equations [78]

$$\begin{aligned} \dot{x} &= V\cos\gamma\cos\psi & \dot{y} &= -V\cos\gamma\sin\psi \\ \dot{h} &= V\sin\gamma & \dot{V} &= [(T-D)/W-\sin\gamma]g \\ \dot{\gamma} &= (n_V-\cos\gamma)g/V & \dot{\psi} &= (n_h/\cos\gamma)g/V \end{aligned} \tag{70}$$

where $x$, $y$, and $h$ are the down range, cross range, and altitude of the aircraft, respectively. $V$, $\psi$, and $\gamma$ are respectively the speed, heading angle and flight path angle,. The weight of the aircraft $W$ is assumed to be $15512.86\,kg$ [79]. The thrust $T$, vertical load factor $n_V$, and horizontal load factor $n_h$ are the control variables. $g$ is the gravitational cost. $D$ is the drag and calculated by $D=1/2\rho V^2 S[C_{D_0}+\eta C_{L_\alpha}^2\alpha^2]$, in which $\rho$ is the atmospheric density, $S$ is the cross-section area, $\alpha$ is the angle of attack, and $\eta$ is the drag due to lift factor. The zero lift drag coefficient $C_{D_0}$ and lift curve slope $C_{L_\alpha}$ are functions of the Mach number $M$ that can be found

in [80].

The initial position and velocity of the supersonic aircraft are set to be $[0,0,0]m$ and $[170,0,0]m/s$. The objective is to find the minimum time trajectory for the supersonic aircraft to climb from its initial position to the final position of $[10000,5000,5000]m$, with a desired final velocity of $[100,100,100]m/s$. The performance index in the LGL discretized form is $J = 0.5\left(t_f - t_0\right)\sum_{i=0}^{N} w_i$, in which $w_i$ is the weight for the $i^{th}$ node. $\left[x, y, h\right]^T$ is regarded as the "position state", while its "state rate" $\boldsymbol{x}_{sr}$ is $[V,\gamma,\psi]^T$. The control variables can be computed by

$$\begin{aligned} T &= (\dot{V}/g + \sin\gamma)W + D \\ n_V &= \dot{\gamma}V/g + \cos\gamma \\ n_h &= \dot{\psi}V\cos\gamma/g \end{aligned} \tag{71}$$

where $\psi = \tan^{-1}(-\dot{y}/\dot{x})$ and $\gamma = \sin^{-1}(\dot{h}/V)$. The control variables are constrained by $0 \le T \le T_{\max} = 89000\,N$, $|n_V| \le n_{V,\max} = 3.8$, and $|n_h| \le n_{h,\max} = 3.8$. The constraints on the speed, flight angle, and heading angle are $0 \le V \le 500\,m/s$, $|\gamma| \le 45^{\circ}$, and $|\psi| \le 45^{\circ}$, respectively.

The cases with 10, 15, 20, and 25 nodes are tested in each of the three methods in BiCF. Here a straight line is used as the initial guess of aircraft trajectory. Six control points ($n_{cp} = 5$) and a degree of four ($d = 4$) are set for the B-spline curves. The reference point in the MC method is guessed to be $[2000,2000,2000]m$, while the initial position of the virtual prey in the CATD method is assumed to be $[1,1,1]m$. The PCPs are guessed to be 1.

The performance indices and the computational cost are shown in **Table 10**. The

following observations are apparent: (1) CPU runtimes are low and only grow slightly as *N* increases (MC: from 4.07s to 6.86s; CATD: from 4.33 to 8.37s; and LP: from 3.01 to 6.45s). (2) The performance indices of all three methods are consistent for different node cases, and the maximum difference is only about 3.85%.

The trajectory solutions for different number of nodes and different motion strategies are similar. Therefore, only the 25-node case is shown in **Figure 8**-**Figure 12**. Since the minimum times between different motion strategies are slightly different, the optimal trajectories for different methods are also slightly different.

**Figure 10-Figure 12** show the control and "state rate" variable time histories of the aircraft for the 25-node case. The thrust remains at the maximum level for most of the time in order to minimize the climbing time. The constraints of the speed, flight path angle, heading angle, g-loads, and thrust are not violated. The oscillations occurring in the beginning and final stages (**Figure 10** and **Figure 12**) are caused by the Gibbs phenomenon in the pseudospectral discretization.

Table 10 Minimum Time-to-climb Problem

| Algorithm | Performance | 10-node | 15-node | 20-node | 25-node |
|---|---|---|---|---|---|
| MC approach | Index (s) | 38.51 | 38.63 | 38.84 | 38.68 |
| | CPU Time (s) | 4.07 | 5.73 | 5.92 | 6.86 |
| LP approach | Index (s) | 38.19 | 38.61 | 38.87 | 38.8 |
| | CPU Time (s) | 3.01 | 5.03 | 5.86 | 6.45 |
| CATD approach | Index (s) | 39.66 | 39.44 | 39.17 | 39.54 |
| | CPU Time (s) | 5.02 | 4.33 | 5.52 | 8.37 |

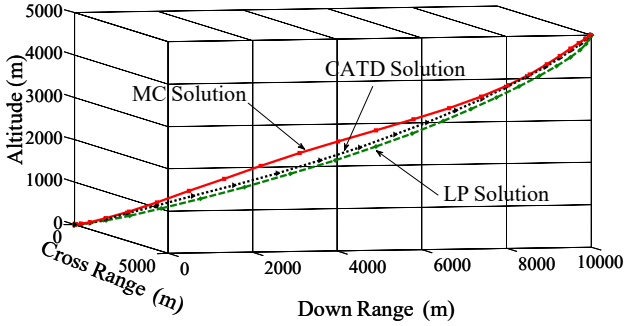



Figure 8 Minimum time-to-climb problem for the 25-node case

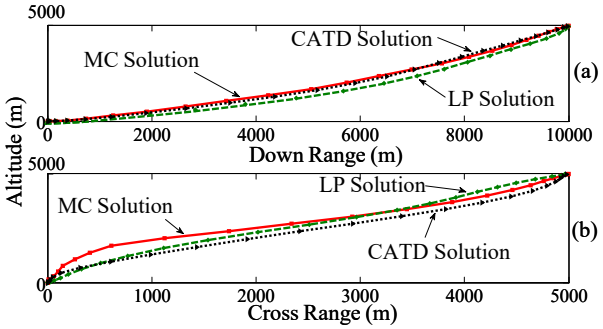



Figure 9 Minimum time-to-climb problem plotted in the (a) down range-altitude and (b) cross range-altitude coordinates

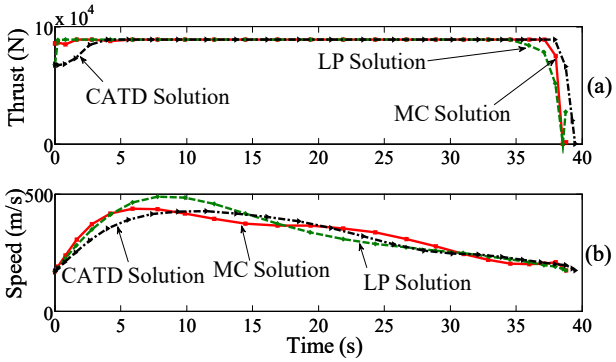



Figure 10 Control and speed histories: (a) thrust and (b) speed

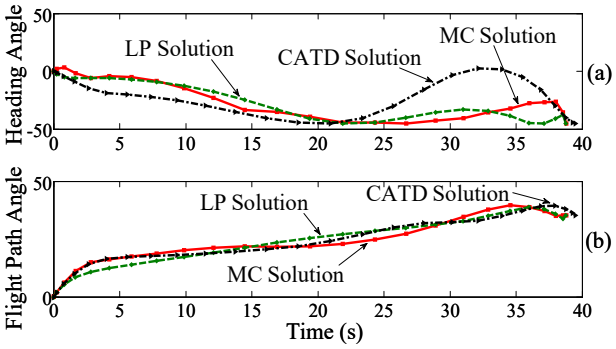



Figure 11 State variables histories: (a) heading angle and (b) flight path angle

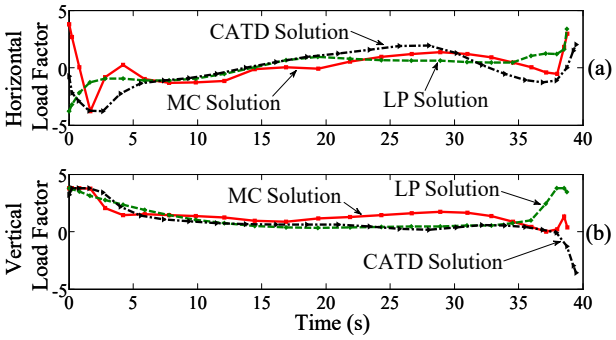



Figure 12 G-load histories: (a) horizontal load factor and (b) vertical load factor

# CHAPTER FOUR: INITIAL GUESS TECHNIQUE

The computational cost associated with the Bio-inspired Computational Framework (BiCF) methods is small. As any other NLP problems, if the initial guess is not good, it is challenging for the optimization iterations to converge. This chapter will discuss initial guess techniques for BiCF method based on Motion Camouflage (MC) strategy.

## Background

Different strategies have been proposed to enhance the initial guesses of: (1) rocket trajectory optimization problems [81]-[82], direct optimization methods can be applied to generate an initial guess, which is then used in indirect trajectory optimization approaches; and (2) aircraft landing trajectory optimization problems [83], a geometric path is computed based on the simplified Dubin's car model is used as the initial guess, and then a time-optimal speed profile is generated.

In this dissertation, to enhance convergence success rate and speed of BiCF based optimization algorithms, the initial guess of the reference point, control points and Path Control Parameters (PCPs) are placed or tuned according to the necessary conditions based on the obstacles and speed constraints. If the initially guessed path does not satisfy these conditions, the algorithm will return to the initial guess step without executing the optimization iteration. Thus, the computational cost will be reduced by not wasting time in one optimization iteration if the chance of violating the obstacle avoidance and speed constraints is high.

Necessary Conditions Based on Velocity Constraints

Necessary condition based on velocity constraint will be derived in this section. The derivation of lemmas is based on the assumption that a vehicle during trajectory optimization has speed requirement, as,

$$0 \le V_k \le V_{\max}, \qquad k = 1,...,N \tag{72}$$

**Lemma 15:** A necessary condition for a vehicle to satisfy the speed limitation $V_k \le V_{\max}$ at node $k$ by using Virtual Motion Camouflage (MC) strategy is

$$\left[ (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k} \right]^2 - \left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r \right\|^2 (V_{p,k}^2 - V_{\max}^2 / v_k^2) \ge 0 \tag{73}$$

**Proof:** To satisfy the speed constraint of $0 \le V_k \le V_{\max}$, i.e. $V_k^2 \le V_{\max}^2$, at node $k$, the velocity equation of the vehicle can be expressed using MC strategy (Eq.(18)), it requires

$$V_k^{\,2} - V_{\max}^2 = \dot{\boldsymbol{x}}_{a,k}^T \dot{\boldsymbol{x}}_{a,k} - V_{\max}^2 = \left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r \right\|^2 \dot{v}_k^2 + 2 v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k} \dot{v}_k + v_k^2 V_{p,k}^2 - V_{\max}^2 \le 0 \tag{74}$$

Let's define $A_k \triangleq \left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r \right\|^2$, $B_k \triangleq 2 v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}$, $C_k \triangleq v_k^2 V_{p,k}^2 - V_{\max}^2$. Then Eq. (74) is expressed as $A \dot{v}_k^2 + B \dot{v}_k + C \le 0$. Based on the quadratic inequality [84], for Eq. (74) to be valid, $B_k^2 - 4 A_k C_k \ge 0$ is required and the solution to Eq. (73) is

$\frac{-B_k - \sqrt{B_k^2 - 4 A_k C_k}}{2 A_k} \le \dot{v}_k \le \frac{-B_k + \sqrt{B_k^2 - 4 A_k C_k}}{2 A_k}$. Therefore the lower and upper bounds of $\dot{v}$ are

$$\dot{v}_{k,\min} = \frac{-v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k} - \sqrt{\left[ v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k} \right]^2 - \left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r \right\|^2 (v_k^2 V_{p,k}^2 - V_{\max}^2)}}{\left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r \right\|^2} \tag{75}$$

and

$$\dot{v}_{k,\max} = \frac{-v_k(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k} + \sqrt{\left[v_k(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 - \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2 (v_k^2 V_{p,k}^2 - V_{\max}^2)}}{\left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2} \quad (76)$$

respectively, in which

$$\left[(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 - \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2 (V_{p,k}^2 - V_{\max}^2 / v_k^2) \geq 0 \quad (77)$$

must be satisfied.

## Initial Guess Techniques

The initial guess techniques are based on the assumptions that the vehicle during trajectory optimization has both a speed requirement (Eq. (72)) and an obstacle avoidance requirement, described as,

$$\left\|\boldsymbol{x}_a - \boldsymbol{x}_{o,i}\right\| \geq r_i, \ i = 1, ...., n_o \quad (78)$$

where $\boldsymbol{x}_{o,i}$ and $r_i$ are the center position and radius of the $i^{th}$ obstacle, respectively, and $n_o$ is the number of obstacles.

For the scenarios with many obstacles, the convergence rate can be improved if the initial guess of vehicle's path is collision free instead of a randomly generated path.

Since the vehicle's path in the dissertation is relying on the choice of the virtual prey path, a collision free corridor will be found for this path based on the wavefront method. In 2D space, the wavefront method can be directly used, but in 3D space, a third direction (i.e. up direction) is added to find 3D path. It is assumed that all the obstacles in 2D space can be approximated by circles. It is additionally assumed that the obstacles in 3D space are approximated using cylinders, and the vehicles will not walk/fly over the top of obstacles

without losing of generosity.

Step 1: In 2D space, a collision free corridor will be found directly by wavefront algorithm [77] as shown in Table 11. In 3D space, a 2D collision free corridor will be first generated in the horizontal plane by using waterfront algorithm as shown in Table 11. Then by assuming the virtual prey is ascending or descending continuously from an initial position until reaching a final position, the up direction of the 3D virtual prey path is a linearly interpolated.

Table 11 Wavefront Algorithm Used in the Horizontal Plane

| Step # | Content |
|---|---|
| Step 1-1: | Dividing east-north horizontal plane into rectangular cells along the east and north directions |
| Step 1-2: | Assigning "1" to the cell occupied by the initial position of vehicle |
| Step 1-3: | Assigning "i+1" to all eight neighbors surrounding the cell labeled with "i", i=1,2,… until reaching the final position of the vehicle. In this process, if a cell is occupied by an obstacle, a big number is assigned (e.g. 1000). |
| Step 1-4: | By following the grid with the fastest decreased value from the destination grid until the initial grid is reached, a 2D obstacle-free path is obtained |

Step 2: The path generated in the Algorithm 1 is typically not smooth. Thus, an ad-hoc NLP is solved to smooth the generated obstacle free path. In this step the performance index is the length of the path generated in Algorithm 1 as

$$J = \sum_{i=0}^{M} l_i \tag{79}$$

where $M$ is the number of interpolation nodes, which can be different from the LGL discretization nodes $N$. $l_i$ is the distance between each interpolated node. The only constraint involved in the trajectory smooth process is that the smoothened path should be obstacle collision free.

**Note 2**: In the MC strategy, the virtual prey path can be generated first because the initial position and the final position of virtual prey are overlapped with those of the vehicle, as shown in **Lemma 1**-**Lemma 3**.

The initial guess of PCPs is suggested to first given the value equal to or very close to 1. According to Eq. (9), if we choose the initial guess of PCPs around 1, then the path of vehicle will be very close or overlapped to the path of prey. If the prey path is obstacle collision free, the path of vehicle will be collision free.

Then, the following Lemma and remarks will be used to tune initial guess of PCPs if **Lemma 15** is violated at certain nodes.

**Lemma 16:** If the necessary condition at node $k$ (Eq.(73)) is not satisfied, a strategy is to decrease PCP $v_k$ in the next initial guess as

$$v_k^+ = v_k^- - \Delta_2,\ \Delta_2 > 0 \tag{80}$$

and $\Delta_2$ should be selected in the range of

$$\begin{aligned} &-\sqrt{\frac{V_{\max}^2}{V_{p,k}^2 - \left[(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 / \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2}} + v_k^- \le \Delta_2 \\ &\le \sqrt{\frac{V_{\max}^2}{V_{p,k}^2 - \left[(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 / \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2}} + v_k^- \end{aligned} \tag{81}$$

**Proof:** Eq. (73) can be rewritten as

$$\left[(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 / \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2 - V_{p,k}^2 + V_{\max}^2 / (v_k^-)^2 \ge 0 \tag{82}$$

Due to the term $V_{\max}^2 / (v_k^-)^2$ is always lager than 0, when Eq. (73) is not satisfied at node $k$, it indicates that $\left[(\boldsymbol{x}_{p,k} - \boldsymbol{x}_r)^T \dot{\boldsymbol{x}}_{p,k}\right]^2 / \left\|\boldsymbol{x}_{p,k} - \boldsymbol{x}_r\right\|^2 - V_{p,k}^2$ must be smaller than 0. To make Eq. (73)

satisfied, new PCP $v_k^+$, Eq. (83) needs to be satisfied,

$$\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2-V_{p,k}^2+V_{\max}^2/(v_k^--\Delta_2)^2\geq 0 \tag{83}$$

Then it can be easily get that

$$V_{\max}^2/(v_k^--\Delta_2)^2\geq V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2 \tag{84}$$

To satisfy Eq. (84), $(v_k^--\Delta_2)^2$ should satisfy

$$(v_k^--\Delta_2)^2\leq\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2} \tag{85}$$

which means

$$\begin{aligned}&-\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}\leq v_k^--\Delta_2\\&\leq\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}\end{aligned} \tag{86}$$

Thus, we can get

$$\begin{aligned}&\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}+v_k^-\geq\Delta_2\\&\geq-\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}+v_k^-\end{aligned} \tag{87}$$

Though changing the position of reference point $\boldsymbol{x}_r$ is another choice to make Eq. (73) satisfied, $\boldsymbol{x}_r$ is not tuned here because the change of $\boldsymbol{x}_r$ at node $k$ will affect the values of Eq. (73) at other nodes.

**Remark 11:** Normally, minimum $\Delta_2$ in Eq. (87) value is chosen so that the adjustment of PCP will now affect the position of predator too much. If new PCP $v_k^+$ make the position of predator fall into the $j^{th}$ obstacle area, then, we should choose a new $\Delta_2$ that satisfy

$$\Delta_2 \geq \frac{2(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)}{\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}-2v_k^- \tag{88}$$

so that the obstacle avoidance condition can be satisfied.

**Proof:** According to Eq. (78), the obstacle avoidance constraint with the new PCP at node $k$ is

$$\left\|\boldsymbol{x}_{o,i}-\boldsymbol{x}_r-v_k^+(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r))\right\|^2 \geq r_i^2 \tag{89}$$

which can be further organized as

$$\left[\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)^T-v_k^+(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\right]\left[\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)-v_k^+(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)\right]\geq r_i^2 \tag{90}$$

and then

$$\begin{aligned}&\left\|\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right\|^2-2v_k^-(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)+\left(v_k^-\right)^2\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2\\&+2v_k^-\Delta_2\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2+\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2\Delta_2^2-2\Delta_2(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)\geq r_i^2\end{aligned} \tag{91}$$

To avoid the negative effects on obstacle avoidance, it is desirable to have

$$2v_k^-\Delta_2\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2+\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2\Delta_2^2-2\Delta_2(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)\geq 0 \tag{92}$$

Since $\Delta_2>0$, Eq. (92) can be further simplified as

$$\Delta_2 \geq \frac{2(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)}{\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}-2v_k^- \tag{93}$$

**Remark 12:** In practice, sometimes, Eq. (93) may conflict with Eq. (81), which means, the value of $\frac{2(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\left(\boldsymbol{x}_{o,i}-\boldsymbol{x}_r\right)}{\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}-2v_k^-$ is larger than $\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}+v_k^-$. In this case, empirical strategy is to choose $\Delta_2$ as the maximum value, $\sqrt{\frac{V_{\max}^2}{V_{p,k}^2-\left[(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r)^T\dot{\boldsymbol{x}}_{p,k}\right]^2/\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r\right\|^2}}+v_k^-$, in Eq. (81).

**Remark 13:** As can be seen in Eq. (75) and Eq. (76), when PCP $v_k$ decreases, the range of $\dot{v}_k$ increases, which will have a higher chance to find proper PCPs to satisfy the speed limitation.

Now, new guess of PCPs, together with calculated control points $\boldsymbol{P'}$ and reference point, will be regarded as the optimizable parameters in the NLP problem. In case NLP software cannot find a converged solution, the reference point will be tuned along or against its original direction according to following Lemmas and Remarks.

**Lemma 17**: If $2\dot{\boldsymbol{x}}_{p,k}^T\boldsymbol{e}_r^-v_k/\dot{v}_k+2\left(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right)^T\boldsymbol{e}_r^-$ is positive. The new reference point $\boldsymbol{x}_r^+$ can be tuned along the original direction as

$$\boldsymbol{x}_r^+=\boldsymbol{x}_r^-+\Delta_1\hat{\boldsymbol{e}}_r^-,\qquad 0<\Delta_1\le\frac{2v_k\dot{\boldsymbol{x}}_{p,k}^T\boldsymbol{e}_r^-+2\left(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right)^T\boldsymbol{e}_r^-\dot{v}_k}{\dot{v}_k} \tag{94}$$

with regard to certain node *k*, or the reference point can be tune against to its original direction as $\boldsymbol{x}_r^+=\boldsymbol{x}_r^--\Delta_1\hat{\boldsymbol{e}}_r^-$ with $\Delta_1>0$. The superscripts "+" and "-" denote the new and old reference points, respectively, and $\hat{\boldsymbol{e}}_r^-=\frac{\boldsymbol{x}_r^-}{\left\|\boldsymbol{x}_r^-\right\|}$.

**Proof**: The speed constraint $V_k\le V_{\max}$ after the reference is adjusted can be expressed using the

reference point, PCP, and the prey motion as

$$\left\| \dot{v}_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r^+) + v_k \dot{\boldsymbol{x}}_{p,k} \right\|^2 \le V_{\max}^2 \tag{95}$$

which can be further expanded as

$$\left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^+ \right\|^2 \dot{v}_k^2 + 2v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r^+)^T \dot{\boldsymbol{x}}_{p,k} \dot{v}_k + v_k^2 V_{p,k}^2 - V_{\max}^2 \le 0 \tag{96}$$

Representing the new reference point guess using the previous guess and the adjustment, Eq. (96) can be written and simplified as

$$\begin{aligned} &\left( \left\| \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right\|^2 \dot{v}_k^2 + 2v_k (\boldsymbol{x}_{p,k} - \boldsymbol{x}_r^-)^T \dot{\boldsymbol{x}}_{p,k} \dot{v}_k + v_k^2 V_{p,k}^2 - V_{\max}^2 \right) + \\ &\left[ \Delta_1^2 \dot{v}_k^2 - 2v_k \Delta_1 \left( \boldsymbol{e}_r^- \right)^T \dot{\boldsymbol{x}}_{p,k} \dot{v}_k - 2\Delta_1 \left( \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right)^T \boldsymbol{e}_r^- \dot{v}_k^2 \right] \le 0 \end{aligned} \tag{97}$$

To make Eq. (97) satisfied, the following equation

$$\Delta_1^2 \dot{v}_k^2 - 2v_k \Delta_1 \left( \boldsymbol{e}_r^- \right)^T \dot{\boldsymbol{x}}_{p,k} \dot{v}_k - 2\Delta_1 \left( \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right)^T \boldsymbol{e}_r^- \dot{v}_k^2 \le 0 \tag{98}$$

is desired. Since $\Delta_1 > 0$, the inequality Eq. (98) can be further simplified as

$$\Delta_1 \le \frac{2v_k \dot{\boldsymbol{x}}_{p,k}^T \boldsymbol{e}_r^- + 2\left( \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right)^T \boldsymbol{e}_r^- \dot{v}_k}{\dot{v}_k} \tag{99}$$

with $\dfrac{2v_k \dot{\boldsymbol{x}}_{p,k}^T \boldsymbol{e}_r^- + 2\left( \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right)^T \boldsymbol{e}_r^- \dot{v}_k}{\dot{v}_k} > 0$. If $\dfrac{2v_k \dot{\boldsymbol{x}}_{p,k}^T \boldsymbol{e}_r^- + 2\left( \boldsymbol{x}_{p,k} - \boldsymbol{x}_r^- \right)^T \boldsymbol{e}_r^- \dot{v}_k}{\dot{v}_k} < 0$, it means the second term in Right-Hand Side (RHS) of Eq. (97) must be larger 0, if the first term in Eq. (97) is negative enough, it is still possible that Eq. (97) with a small $\Delta_1$ is valid. Or, reference point can be tune against to its original direction.

To make speed constraint $V_k \le V_{\max}$ satisfied, with $\boldsymbol{x}_r^+ = \boldsymbol{x}_r^- - \Delta_1 \hat{\boldsymbol{e}}_r^-$, Eq. (96) can be written and simplified as

$$\left(\left\|\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right\|^2\dot{v}_k^2+2v_k(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-)^T\dot{\boldsymbol{x}}_{p,k}\dot{v}_k+v_k^2V_{p,k}^2-V_{\max}^2\right)-$$
$$\left[\Delta_1^2\dot{v}_k^2-2v_k\Delta_1\left(\boldsymbol{e}_r^-\right)^T\dot{\boldsymbol{x}}_{p,k}\dot{v}_k-2\Delta_1\left(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right)^T\boldsymbol{e}_r^-\dot{v}_k^2\right]\le 0 \tag{100}$$

To make Eq. (100) satisfied, the following equation

$$\Delta_1^2\dot{v}_k^2-2v_k\Delta_1\left(\boldsymbol{e}_r^-\right)^T\dot{\boldsymbol{x}}_{p,k}\dot{v}_k-2\Delta_1\left(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right)^T\boldsymbol{e}_r^-\dot{v}_k^2\ge 0 \tag{101}$$

is desired. Since $\dfrac{2v_k\dot{\boldsymbol{x}}_{p,k}^T\boldsymbol{e}_r^-+2\left(\boldsymbol{x}_{p,k}-\boldsymbol{x}_r^-\right)^T\boldsymbol{e}_r^-\dot{v}_k}{\dot{v}_k}<0$, if $\Delta_1>0$, Eq. (101) will be satisfied.

The structure of BiCF optimization algorithm based on MC strategy with the enhanced initial guess techniques is shown in Figure 13. The wavefront algorithm is first modified to find a 3D collision free corridor for the virtual prey. This corridor is then smoothed by solving an ad-hoc trajectory optimization problem and the control points $\boldsymbol{P'}$ of the prey is calculated according to Eq. (15). The speed of the vehicle will be checked if satisfy with speed necessary condition (i.e. **Lemma 15**). If the speed at certain node violates **Lemma 15**, PCP at that node needs to be amended according to **Lemma 16** and **Remark 11**-**Remark 12**. Calculated the control points $\boldsymbol{P'}$, together with initial guess of reference point and new PCPs, will be regarded as the optimizable parameters in the NLP problem. Followed the procedure described in the Section III, the trajectory optimization problem can be solved by any NLP software. In case NLP software cannot find a convergent solution, the reference points will be tuned to a new value according to **Lemma 17**, and the above procedure will be repeated until a convergent solution is found or maximum iteration times is achieved.

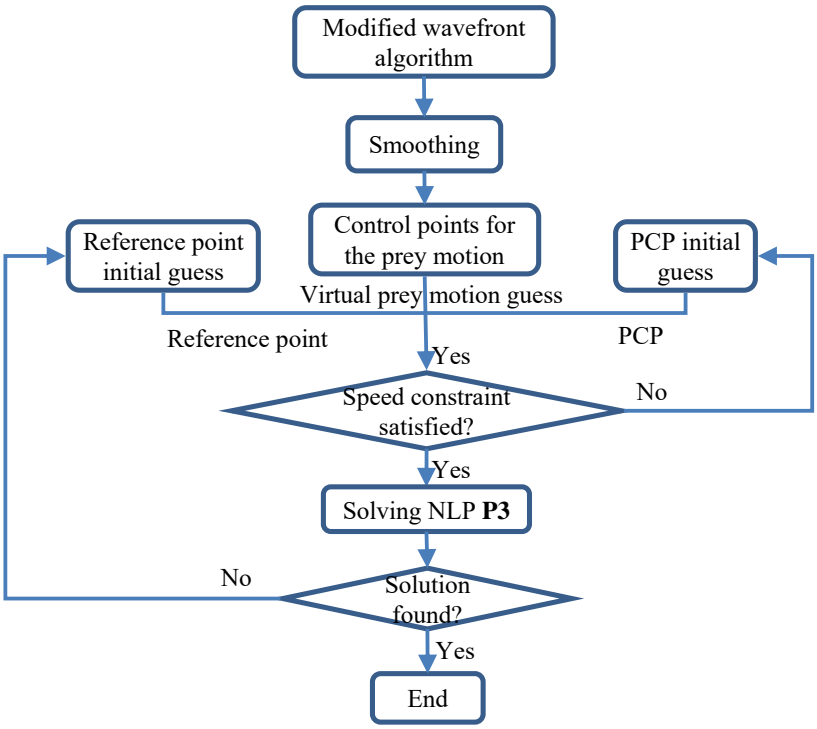



Figure 13 Enhanced initial guess technique for BiCF optimization algorithm based on MC strategy

## Simulation and Analysis

To show the necessity of enhanced initial guess approach, this section will show the Monte Carlo simulations on MAV 3D trajectory optimization process. The dynamic model is [103]

$$\begin{bmatrix} \dot{x}_E \\ \dot{x}_N \\ \dot{x}_U \\ \dot{V} \\ \dot{\gamma} \\ \dot{\chi} \end{bmatrix} = \begin{bmatrix} V\cos\chi\cos\gamma \\ V\sin\chi\cos\gamma \\ V\sin\gamma \\ g[(T-D)/W-\sin\gamma] \\ (g/V)(k_n n\cos\mu-\cos\gamma) \\ (gk_n n\sin\mu)/(V\cos\gamma) \end{bmatrix} \tag{102}$$

Here $\boldsymbol{x}_a \triangleq [x_E, x_N, x_U]^T \in \Re^{3\times 1}$ are the MAV east, north, and up positions in the wind coordinate, while $\boldsymbol{x}_s \triangleq [V, \gamma, \chi]^T \in \Re^{3\times 1}$ are the speed, flight path angle, and heading angle. $T$, $n$, and $\mu$ are the thrust, g-load, and bank angle, respectively.

$D = 0.5\rho V^2 S C_{D0} + 2kk_n^2 n^2 W^2 / (\rho V^2 S)$ represents the drag, where $g$ is the gravitational constant, $W$ is the weight of MAV, $S$ is the surface area, and $\rho$ is the atmospheric density. The induced drag coefficient and the load factor effectiveness are denoted by $k$ and $k_n$, respectively.

In each run, the same but randomly generated scenarios are used in the MAV 3D trajectory optimization process with two different initial guess generation approaches: 1) in the first optimization process, the initial guess of control points are found based on MAV obstacle free corridor, and the choice of initial guess of reference points and PCPs are also chosen according to the necessary conditions mentioned in the Section III.C, and perturbation technique will be used to tune reference point if "fmincon" could not find convergent solution of achieved NLP problem; 2) in the second optimization process, the initial guess of control points are based on a straight line connecting the initial position and final position, and the reference points and PCPs are randomly guessed. It is worth noting that only the initial guess methods are different, and the optimization procedure, which are solved by the numerical NLP optimization solver "FMINCON", are the same. A total of 3,000 runs are conducted in set of simulations.

The MAV flying testing area is defined to be $600\ m$ in both x and y directions. Cylindrical buildings are in the testing area generated with a radius varying uniformly within $20 \sim 30\,m$ and a height varying uniformly within $50 \sim 90\,m$. In the first experiment, the number

of buildings varies between two and five, while in the second experiment the number of buildings is between fourteen and sixteen. The level of constraints in the experiment 2 is severer than that in the experiment 1.

The initial positions of the MAV in the two experiments are uniformly distributed in the lower left corner of the test area (i.e. $[0,1]m$ in the x position, $[0,1]m$ in the y position, and $[0,25]m$ in the z position respectively). Accordingly, the final position of the MAV is uniformly distributed in the upper right part of the test area (i.e. $[590,600]m$ in the x position, $[590,600]m$ in the y position, $[25,50]m$ in the z position respectively). Without the loss of generality, the MAV is initially heading towards the final position, and the initial speed is set to be $0.1m/s$. The constraints of the MAV motion are: (i) the flight path angle is bounded by +/- $45^{\circ}$, (ii) the maximum speed is $10m/s$, (iii) the maximum thrust is assumed to be $2.5N$, and (iv) the bank angle $\mu$ is within +/-$10^{\circ}$. Because the simulation goal in this section is to demonstrate the effectiveness of the enhance initial guess approach, the aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ are directly given as 0.015, 0.0158, and 0.4, respectively .

In both of the experiments, $N=15$, $n_{cp}=5$ and $d=3$. The initial guess of the final time is calculated by

$$t_f = \left\| \boldsymbol{x}_a(t_f) - \boldsymbol{x}_a(t_0) \right\| / V_{\max} \tag{103}$$

In two different initial guess generation methods, if a converged solution is not achieved within 100 iterations or 5000 function evaluations in the “FMINCON”, a “failure” case is counted against this method. Additionally, the minimum performance index among all two optimization procedure is regarded as the best solution. If the performance index from other

initial guess method is within 5% difference of the best solution, the solution found using this method is regarded as an “optimal” one; otherwise it will be regarded as a “feasible” solution only.

The results of two experiments are shown in Figure 7 and Table 3. The following observations are obvious: 1) compared with the MC strategy based varying manifold optimization approach with randomly initial guess, no matter how severe the constraints are, the optimization procedure with the enhanced initial guess can always solve 3D MAV trajectory optimization problem with highest success rate; 2) when the initial guess is randomly given, 3D MAV trajectory optimization procedure can be solved with high successful rate only when there are small number of obstacles; (3) compare with the optimization procedure with random initial guess, the CPU calculation time of optimization procedure with enhanced initial guess approach is less; and (4) the differences of CPU time between the optimization procedure with and without enhanced initial guess are higher when the number of obstacles is larger.

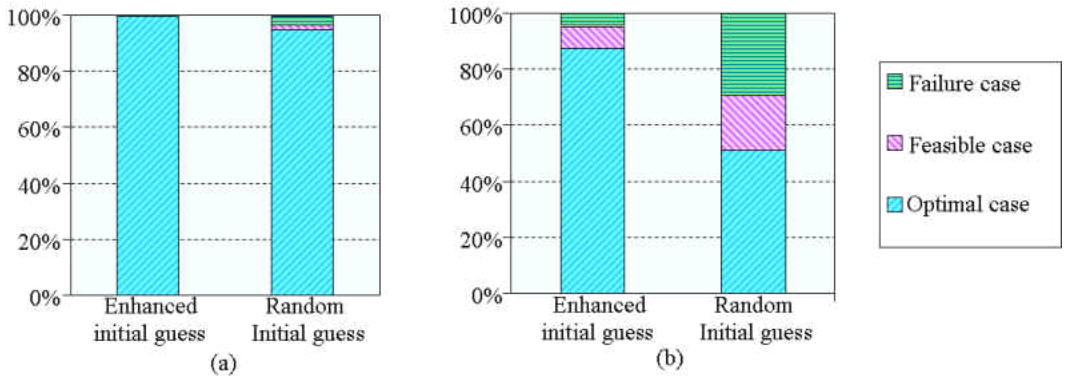



Figure 7 the optimal, feasible and failure solution rate between the optimization procedure with and without enhance initial guess approach under different obstacle scenarios: a) 2~5 obstacles; b) 14~16 obstacles.

Table 5 Average CPU Time of Monte Carlo Simulations

| Algorithms | Average CPU Time | |
|---|---|---|
| | Enhance initial guess approach | Random initial guess approach |
| 1st experiment | 0.94 s | 1.68 s |
| 2nd experiment | 1.45 s | 4.56 s |

# CHAPTER FIVE: APPLICATION OF MC TRAJECTORY PLANNING METHOD ON MICRO AIR VEHICLE WITH UNKNOWN PARAMETERS

In this chapter, the proposed trajectory optimization method will be applied to a MAV 3D trajectory planning problem with unknown dynamic parameters. The content of this chapter is modified from our conference paper [5].

## Background

Recent advances in miniaturization have made Micro Air Vehicles (MAVs) a viable option for many applications, such as surveillance, information collection, and environment monitoring [85]-[90]. As a new generation of Unmanned Aerial Vehicles (UAVs), MAV is smaller, lighter, and more agile as compared to larger UAVs, which makes it very attractive in obstacle-laden urban environments. To date, majority of the MAV research has been focused on MAV designs considering different sensing and navigation methods [91]-[93]. However, for a MAV mission to be successful, trajectory planning and control capability is also crucial.

MAV/UAV's trajectory planning has been investigated in several works [94]-[101]. In [94]-[97], waypoints are generated between MAV's initial and final positions, and then are connected using line segments. Voronoi diagram is used to find threat locations, based on which feasible paths are generated along the edge of convex cells using search methods such as the k-best first method and the A*method [100]. Receding horizon control methods divide a long horizon optimization problem into a series of short time-horizon optimization problems over a sliding window [101]. In each planning horizon, a local path planning algorithm, such as the spherical cone based path planning method, is implemented to lead the MAV toward the goal

while avoiding obstacles.

An enhanced motion camouflage (MC) based varying manifold method for constrained optimal trajectory planning in each planning horizon. The MC optimization method has been presented in Chapter 3, in which the actual vehicle is regarded as a predator moving in a varying manifold formed by a virtual prey (typically the initial guess of the vehicle trajectory) and a reference point. The optimization within the varying manifold is controlled by a one-dimension path control parameters (PCPs). The advantages of this method are: (1) the dimension of the converted nonlinear programming (NLP) problem within the varying manifold is very small; and (2) the dimension of the optimizable parameters representing the varying manifold is small.

In addition to the open-loop trajectory optimization in each planning horizon, a linear quadratic regulator (LQR) is designed for the MAV to track the rapidly generated optimal trajectory. Meanwhile, the unknown aerodynamics will be estimated and updated in the next horizon. Different from [102], in which the parameters of MAV models are identified off-line based on frequency responses using Schroeder sweeps, an extended Kalman filter (EKF) is applied to estimate the unknown parameters online.

## Receding Horizon Framework

To respond to unexpected events such as pop-up obstacles in urban areas and to update unknown aerodynamic coefficients, the trajectory planning and control is casted into a receding horizon framework to form a close-loop scheme, as shown in Figure 14. In each horizon, an optimal trajectory is generated for MAV to fly from its initial position to the final position of the current horizon. The dynamic model used in the MAV trajectory planning is a nominal one, in

which the aerodynamic coefficients are estimated, no noise is considered, and non-affine terms are neglected. In order to compensate the mismatches between the control commands calculated in the open-loop trajectory optimization and the real control commands needed by the MAV, at every sampling time of the control loop, the calculated open-loop control commands and state variables will be interpolated and fed into the regulator to compute the actual control commands for the MAV. In the mean time, a nonlinear filter will use the measured state variables and the control commands to estimate the unknown parameters, which are used to update the open-loop planning and LQR subsystems at the beginning of the next horizon.

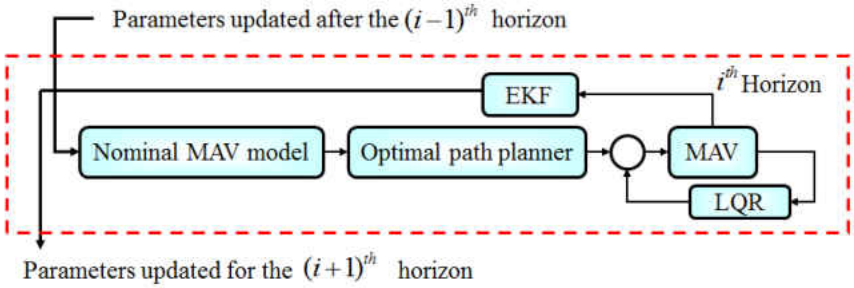



Figure 14 Receding horizon framework

## MAV Dynamic Model

The MAV model is adopted from[103], and specific MAV parameters and aerodynamic coefficients are chosen from [104]. The 3D dynamic model is [103]

$$\begin{bmatrix} \dot{x}_E \\ \dot{x}_N \\ \dot{x}_U \\ \dot{V} \\ \dot{\gamma} \\ \dot{\chi} \end{bmatrix} = \begin{bmatrix} V\cos\chi\cos\gamma \\ V\sin\chi\cos\gamma \\ V\sin\gamma \\ g[(T-D)/W-\sin\gamma] \\ (g/V)(k_n n\cos\mu-\cos\gamma) \\ (gk_n n\sin\mu)/(V\cos\gamma) \end{bmatrix} \tag{104}$$

Here $\boldsymbol{x}_a \triangleq [x_E, x_N, x_U]^T \in \Re^{3\times 1}$ are the MAV east, north, and up positions in the wind coordinate, while $\boldsymbol{x}_s \triangleq [V, \gamma, \chi]^T \in \Re^{3\times 1}$ are the speed, flight path angle, and heading angle. $T$, $n$, and $\mu$ are the thrust, g-load, and bank angle, respectively. $D = 0.5\rho V^2 S C_{D0} + 2kk_n^2 n^2 W^2/(\rho V^2 S)$ represents the drag, where $g$ is the gravitational constant, $W$ is the weight of MAV, $S$ is the surface area, and $\rho$ is the atmospheric density. The induced drag coefficient and the load factor effectiveness are denoted by $k$ and $k_n$, respectively. To obtain a control affine nominal model, new control variables are defined as $\boldsymbol{u} \triangleq [T, n\cos\mu, n\sin\mu]^T$, and the MAV model is rewritten as

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{x}}_a \\ \dot{\boldsymbol{x}}_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_a(\boldsymbol{x}_s) \\ \boldsymbol{f}_s(\boldsymbol{x}_s) \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{3\times 3} \\ \boldsymbol{B}(\boldsymbol{x}_s) \end{bmatrix} \boldsymbol{u} = \boldsymbol{f}(\boldsymbol{x}_s) + \boldsymbol{G}(\boldsymbol{x}_s)\boldsymbol{u} \tag{105}$$

in which

$$\begin{aligned} \boldsymbol{f}_a &\triangleq [V\cos\chi\cos\gamma, V\sin\chi\cos\gamma, V\sin\gamma]^T \\ \boldsymbol{f}_s &\triangleq [-g\sin\gamma - \tilde{D}/W, -g\cos\gamma, 0]^T \\ \boldsymbol{B} &\triangleq diag\{g/W, gk_n/W, gk_n/(V\cos\gamma)\}^T \end{aligned} \tag{106}$$

and

$$\tilde{D} = -0.5 g \rho V^2 S C_{D0} \tag{107}$$

Note that to achieve the control affine nominal model, the drag term is simplified as $\tilde{D}$, and the

mismatches will be compensated by the tracking controller. Here $\boldsymbol{f}_a(\boldsymbol{x}_s)$ , $\boldsymbol{f}_s(\boldsymbol{x}_s)$ and $\boldsymbol{B}(\boldsymbol{x}_s)$ are smooth.

MC Strategy Based Trajectory Optimization

The cost function is assumed to be minimum time, which is represented as

$$J = \int_{t_0}^{t_f} dt \tag{108}$$

The following inequality constraints $\boldsymbol{g}_1(\boldsymbol{x},\boldsymbol{u},t) \le 0$ are considered including

$$\begin{aligned} &0 \le V \le V_{\max}, \ \ |\gamma| \le \gamma_{\max}, \ \ |\chi| \le \chi_{\max}, \\ &0 \le T \le T_{\max}, |\mu| \le \mu_{\max}, \ \ |n| \le n_{\max}, \\ &\left\| \boldsymbol{x}_a - \boldsymbol{x}_{o,i} \right\| \ge r_i, \ i = 1,....,n_o \end{aligned} \tag{109}$$

where $\boldsymbol{x}_{o,i}$ and $r_i$ are the center position and radius of the $i^{th}$ obstacle, respectively, and $n_o$ is the number of obstacles. It is assumed that all the obstacles in the urban area are approximated using cylinders without loss of generality and the MAV will not fly over the top of obstacles. Boundary conditions and MAV dynamics are regarded as the equality constraints. In each planning horizon, the initial state and final position state variables are known and defined as

$$\boldsymbol{x}(t_0) = [\boldsymbol{x}_{a,0}, \boldsymbol{x}_{s,0}]^T, \ \ \boldsymbol{x}_a(t_f) = \boldsymbol{x}_{a,f} \tag{110}$$

The final velocity $\boldsymbol{x}_s(t_f)$ in the planning horizon is assumed to be free.

The Virtual Motion Camouflage (MC) theory talked in the Chapter 3 is used to solve MAV trajectory optimization problem. Based on the Eq. (25), since the relative degree of $y = x_a$ is 2, and the control variables for the MAV to stay on the manifold is

$$\boldsymbol{u} = \left[\frac{\partial \boldsymbol{f}_a}{\partial \boldsymbol{x}_s}\boldsymbol{B}\right]^{-1}\left[\ddot{v}(\boldsymbol{x}_p - \boldsymbol{x}_r) + 2\dot{v}\dot{\boldsymbol{x}}_p + v\ddot{\boldsymbol{x}}_p - \frac{\partial \boldsymbol{f}_a}{\partial \boldsymbol{x}}\boldsymbol{f}_a - \frac{\partial \boldsymbol{f}_a}{\partial \boldsymbol{x}_s}\boldsymbol{f}_s\right] \tag{111}$$

Excluded control variables, other state variables also need to be expressed using the position state. In the MAV model, the state variables $V$, $\gamma$, and $\chi$ can be found by

$$V = \|\dot{\boldsymbol{x}}_a\|, \quad \gamma = \sin^{-1}\left(\dot{\boldsymbol{x}}_{a,3} / \|\dot{\boldsymbol{x}}_a\|\right), \quad \chi = \tan^{-1}\left(\dot{\boldsymbol{x}}_{a,2} / \dot{\boldsymbol{x}}_{a,1}\right) \tag{112}$$

Now, all the state and control variables in the Eq. (105) can be expressed in terms of $\boldsymbol{x}_p$, $\boldsymbol{x}_r$ and $v$. Therefore, solving the MAV trajectory problem is equivalent to solving as the following constrained optimization problem

$$\begin{aligned} &\min_{\boldsymbol{x}_p, \boldsymbol{x}_r, v, t_f} \quad J_2 = t_f \\ &\text{subject to}: \quad \boldsymbol{g}_2(\boldsymbol{x}_p, \boldsymbol{x}_r, v, t) \le 0 \\ &\qquad\qquad \boldsymbol{h}_2'(\boldsymbol{x}_p, \boldsymbol{x}_r, v, t) = 0 \end{aligned} \tag{113}$$

It is worth noting that the equality constraints $\boldsymbol{h}_2'(\boldsymbol{x}_a, \boldsymbol{x}_r, v, t) = 0$ now only include the boundary conditions, because the nonlinear dynamic has been considered in the process of finding the state variable $\boldsymbol{x}_s$ and the equivalent control.

To convert MAV trajectory optimization problem to a NLP problem, optimizable variables $\boldsymbol{x}_p$ and $v$ are discretized along the time nodes $\{t_k\}_{k=0}^{N}$ with $t_N = t_f$ and $t_k < t_{k+1}$. To be more specific, the PCP variable $v$ can be discretized using the Legendre-Gauss-Lobatto based pseudospectral method. The virtual prey motion to be evaluated at the pseudospectral nodes is represented using a B-spline as

$$x_{p,j,k} = \sum_{i=0}^{n_{cp}} \Upsilon_{i,d}(t_k) P_{j,i}; \; j = 1,2,3; \; k = 0,...,N \tag{114}$$

where $\Upsilon_{i,d}(t_k), i = 0,...,n_{cp}$ are the $d^{th}$ degree basis functions evaluated at the pseudospectral time node $t_k$, $P_{j,i}$ is the $i^{th}$ control point for the $j^{th}$ direction of virtual prey posit-ion, and $n_{cp}+1$ is the number of control points. More information on B-splines can refer to [66]-[67].

Thus, when the number of control points and the number of pseudospectral node approach infinite, solving P2 is equivalent to solving P3

$$\begin{aligned} &\min_{\boldsymbol{P}_i, \boldsymbol{x}_r, v_k, t_f} \quad J_3 = t_f \\ &\text{subject to}: \quad \boldsymbol{g}_3(\boldsymbol{P}_i, \boldsymbol{x}_r, v_k, t) \le 0 \\ &\qquad\qquad \boldsymbol{h}_3^{'}(\boldsymbol{P}_i, \boldsymbol{x}_r, v_k, t) = 0 \end{aligned} \tag{115}$$

where $\boldsymbol{P}_i = [P_{E,i}, P_{N,i}, P_{U,i}]^T$, $k = 0,...,N$, and $i = 0,...,n_{cp}$.

As mentioned in the Chapter 3, different boundary conditions, belonging to equality constraints, can be used to calculate certain PCP and control point nodes. For the specific boundary condition in the MAV problem (i.e. the initial state $\boldsymbol{x}$ and final position state $\boldsymbol{x}_a$ are known), (i) the initial and final PCPs, $v_0$ and $v_N$, are set to be 1; (ii) the second PCP $v_1$ can be calculated

$$v_1 = \left[(x_{p,k,0} - x_{r,k}) D_{01}^{'}\right]^{-1} \left\{ \dot{x}_{k,0} - (x_{p,k,0} - x_{r,k}) \sum_{j=0,\, j\neq 1}^{N} D_{0j}^{'} v_j - v_0 \sum_{l=0}^{n_{cp}} \dot{\Upsilon}_{l,d}(t_0) P_{k,l} \right\}, \quad k = 1,2,3 \tag{116}$$

in which $\dot{x}_{1,0} = V_0 \cos\chi_0 \cos\gamma_0$, $\dot{x}_{2,0} = V_0 \sin\chi_0 \cos\gamma_0$, and $\dot{x}_{3,0} = V_0 \sin\gamma_0$; (iii) the first control point $\boldsymbol{P}_0 = [P_{1,0}, P_{2,0}, P_{3,0}]^T$ and the last control point $\boldsymbol{P}_f == [P_{1,f}, P_{2,f}, P_{3,f}]^T$ equal to the initial position state $\boldsymbol{x}_0$ and final position state $\boldsymbol{x}_f$, respectively. (iv) Given $P_{k,1}$,of which $k$ is the selected east, north or up direction of $P_1$, the other direction of $P_1$, $P_{m,1}$, $m \neq k$ can be

calculated by

$$P_{m,1}=\frac{1}{\dot{\Upsilon}_{1,d}(t_0)}\left\{\dot{x}_{m,0}-(x_{p,m,0}-x_{r,m})\sum_{j=0,\,j\neq 1}^{N}D_{0j}^{'}v_j-\sum_{l=0,l\neq 1}^{n_{cp}}\dot{\Upsilon}_{l,d}(t_0)P_{m,l}\right\}- \\ \frac{\left[(x_{p,m,0}-x_{r,m})\right]}{\dot{\Upsilon}_{1,d}(t_0)\left[(x_{p,k,0}-x_{r,k})\right]}\left\{\dot{x}_{k,0}-(x_{p,k,0}-x_{r,k})\sum_{j=0,\,j\neq 1}^{N}D_{0j}^{'}v_j-\sum_{l=0}^{n_{cp}}\dot{\Upsilon}_{l,d}(t_0)P_{k,l}\right\},\quad m\neq k \tag{117}$$

where $k$ is an arbitrarily selected direction of MAV state $\boldsymbol{x}$ and satisfies $x_{p,k,0}\neq x_{r,k}$. In Eq. (116) and Eq. (117), $D_{ij}^{'}=[2/(t_f-t_0)]D_{ij}$ and D is the differentiation matrix.

Since certain control points and PCP nodes can be calculated instead of being optimized as shown in the Eqs. (116)-(117), solving Eq. (115) is equivalent to solve

$$\begin{aligned}&\min_{\boldsymbol{P}',\boldsymbol{x}_r,v',t_f}\quad J_4=t_f\\&\text{Subject to}\qquad \boldsymbol{g}_4(\boldsymbol{P}',\boldsymbol{x}_r,v',t)\leq 0\end{aligned} \tag{118}$$

where $\boldsymbol{P}'$ includes $P_{k,i},i=1,...,n_{cp}-1,k=1,...,n_a$ and $P_{m,i},i=2,...,n_{cp}-1,m=1,...,n_a,m\neq k$, and $v'$ includes $v_i,i=2,...,N-1$. Note that now there are no equality constraints, since all of the boundary conditions are incorporated in the Eqs. (116)-(117).

## Trajectory Tracking and Parameters Estimation

There are unknown parameters and uncertainties in the real model, this section will talk about the methods applied to estimate the parameters online and adjust the small difference between the real and nominal control commands.

### Linear Quadratic Regulator

The open-loop control cannot be directly used in the real MAV due to the following

reasons: (i) the aerodynamic coefficients are not perfectly known, (ii) there are numerical mismatches among the pseudo-spectral discretization, b-spline representations, interpolation, and numerical integration, and (iii) the open-loop planned controller is not robust with respect to noises and uncertainties. Therefore, a linear quadratic regulator (LQR) is designed here to reject the disturbance and mismatches. Let's assume that the state and control variables are the summation of the nominal values (found from trajectory optimization) and the small disturbances [105] as

$$\boldsymbol{x}_a = \hat{\boldsymbol{x}}_a + \Delta\boldsymbol{x}_a,\ \boldsymbol{x}_s = \hat{\boldsymbol{x}}_s + \Delta\boldsymbol{x}_s,\ \boldsymbol{u} = \hat{\boldsymbol{u}} + \Delta\boldsymbol{u} \tag{119}$$

where "^" represents nominal value that are calculated in the last section.

By neglecting the high order terms, the state equations are derived based on Eq. (105) and Eq. (119) as

$$\begin{cases} \dot{\hat{\boldsymbol{x}}}_a + \Delta\dot{\boldsymbol{x}}_a = \boldsymbol{f}_p(\hat{\boldsymbol{x}}_s) + \left.\dfrac{\partial \boldsymbol{f}_p}{\partial \boldsymbol{x}_s}\right|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s \\ \dot{\hat{\boldsymbol{x}}}_s + \Delta\dot{\boldsymbol{x}}_s = \boldsymbol{f}_s(\hat{\boldsymbol{x}}_s) + \boldsymbol{B}(\hat{\boldsymbol{x}}_s)\hat{\boldsymbol{u}} + \left.\dfrac{\partial \boldsymbol{f}_s}{\partial \boldsymbol{x}_s}\right|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s + \left.\dfrac{\partial \boldsymbol{B}}{\partial \boldsymbol{x}_s}\right|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s\hat{\boldsymbol{u}} + \boldsymbol{B}(\hat{\boldsymbol{x}}_s)\Delta\boldsymbol{u} \end{cases} \tag{120}$$

and the small disturbance model is derived as

$$\begin{cases} \Delta\dot{\boldsymbol{x}} = \left.\dfrac{\partial \boldsymbol{f}_p}{\partial \boldsymbol{x}_s}\right|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s \\ \Delta\dot{\boldsymbol{x}}_s = \left.\dfrac{\partial \boldsymbol{f}_s}{\partial \boldsymbol{x}_s}\right|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s + \boldsymbol{A}_{22}\Delta\boldsymbol{x}_s + \boldsymbol{B}(\hat{\boldsymbol{x}}_s)\Delta\boldsymbol{u} \end{cases} \tag{121}$$

in which $\boldsymbol{A}_{22}\Delta\boldsymbol{x}_s$ is reorganized from $\left(\partial\boldsymbol{B}/\partial\boldsymbol{x}_s\right)\big|_{\hat{\boldsymbol{x}}_s} \Delta\boldsymbol{x}_s\hat{\boldsymbol{u}}$. The linear disturbance model is derived as

$$\Delta\dot{\boldsymbol{X}} = \boldsymbol{A}(\hat{\boldsymbol{x}}_s, \hat{\boldsymbol{u}})\Delta\boldsymbol{X} + \boldsymbol{B}(\hat{\boldsymbol{x}}_s)\Delta\boldsymbol{u} \tag{122}$$

where the small disturbance state variable is defined as $\Delta\boldsymbol{X}^T = \left[\Delta\boldsymbol{x}_a{}^T, \Delta\boldsymbol{x}_s^T\right]$.

For MAV to track the designed trajectory under small control tuning, a standard LQ controller is designed for Eq. (122) with the following performance index [35]

$$\min J = 0.5\int_0^\infty \left(\Delta\boldsymbol{X}^T\boldsymbol{Q}_l\Delta\boldsymbol{X} + \Delta\boldsymbol{u}^T\boldsymbol{R}_l\Delta\boldsymbol{u}\right)dt \tag{123}$$

in which $\boldsymbol{Q}_l$ and $\boldsymbol{R}_l$ are the weighting matrices. $\Delta\boldsymbol{u}^T\boldsymbol{R}_l\Delta\boldsymbol{u}$ in the Eq. (123) is used to make sure the control will not be tuned too much in practice, so that constraints on control variables will not be violated. The boundary conditions are $\Delta\boldsymbol{X}(t_0) = \boldsymbol{X}_0 - \hat{\boldsymbol{X}}_0$ and $\Delta\boldsymbol{X}(\infty) = 0$. According to, the optimal control commands $\Delta\boldsymbol{u}^*$ can be calculated through [35]

$$\Delta\boldsymbol{u}^* = -\boldsymbol{R}_l^{-1}\boldsymbol{B}\boldsymbol{P}\Delta\boldsymbol{X}^* \tag{124}$$

where,

$$\Delta\dot{\boldsymbol{X}}^*(t) = \left[\boldsymbol{A} - \boldsymbol{B}\boldsymbol{R}_l^{-1}\boldsymbol{B}'\bar{\boldsymbol{P}}\right]\Delta\boldsymbol{X}^*(t) \tag{125}$$

and $\bar{\boldsymbol{P}}$ is solved by the matrix algebraic Riccati equation [35]

$$-\bar{\boldsymbol{P}}\boldsymbol{A} - \boldsymbol{A}'\bar{\boldsymbol{P}} - \boldsymbol{Q}_l + \bar{\boldsymbol{P}}\boldsymbol{B}\boldsymbol{R}_l^{-1}\boldsymbol{B}'\bar{\boldsymbol{P}} = 0 \tag{126}$$

Extended Kalman Parameter Estimation Method

The aerodynamic coefficients may be unknown since they will be changed with the speed of MAV and the wind. Here an extended Kalman filter (EKF) is designed to estimate the unknown coefficients.

The aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ are seemed as unknowns. To estimate

these coefficients, a new state vector is defined as $\boldsymbol{x}' = [k_n, C_{d0}, k]^T$, and the augmented state vector is $\boldsymbol{X} = \left[\dot{\boldsymbol{x}}, \dot{\boldsymbol{x}}_s, \dot{\boldsymbol{x}}'\right]^T$. Therefore the processing model and the measurement model are written as

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{x}}_s \\ \dot{\boldsymbol{x}}' \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_1(\boldsymbol{x}_s) \\ \boldsymbol{f}_s(\boldsymbol{x}_s, \boldsymbol{x}') + \boldsymbol{B}(\boldsymbol{x}_s, \boldsymbol{x}')\boldsymbol{u} + \boldsymbol{w} \\ \boldsymbol{w}' \end{bmatrix} = \boldsymbol{f}(\boldsymbol{X}, \boldsymbol{u}) + \boldsymbol{W}, \quad \boldsymbol{y} = \boldsymbol{H}\boldsymbol{X} + \boldsymbol{v} \tag{127}$$

where the noise vector $\boldsymbol{W} = \left[\boldsymbol{w}, \boldsymbol{w}'\right]^T$. $\boldsymbol{w}' = [w_{k_n}, w_{k_n}, w_k]^T$ is an artificial noise added to the MAV system that allows the EKF to estimate its new state $\boldsymbol{x}'$. $\boldsymbol{w} \in \Re^{3\times 1}$ and $\boldsymbol{v} \in \Re^{6\times 1}$ are zero-mean Gaussian noises in the processing and measurement models, with the covariance matrices of $E[\boldsymbol{w}(t)\boldsymbol{w}(\tau)^T] = \mathbf{Q}(t)\delta(t-\tau)$ and $E[\boldsymbol{v}(t)\boldsymbol{v}(\tau)^T] = \boldsymbol{R}(t)\delta(t-\tau)$, respectively. $\boldsymbol{H} = \left[\boldsymbol{I}_6\ \boldsymbol{0}_3\right]$, where $\boldsymbol{I}_6$ is an identity matrix with a size of $6\times 6$, and $\boldsymbol{0}_3$ is zero matrix with a size of $3\times 3$.

The continuous-time extended Kalman filter [106] is written as

$$\begin{aligned} \dot{\hat{\boldsymbol{X}}} &= \boldsymbol{f}(\hat{\boldsymbol{X}}, \boldsymbol{u}) + \boldsymbol{K}(\boldsymbol{y} - \boldsymbol{H}\hat{\boldsymbol{X}}) \\ \boldsymbol{K} &= \boldsymbol{P}\boldsymbol{H}^T\boldsymbol{R}^{-1} \\ \dot{\boldsymbol{P}} &= \boldsymbol{F}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{F}^T + \boldsymbol{L}\boldsymbol{Q}\boldsymbol{L}^T - \boldsymbol{P}\boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H}\boldsymbol{P} \end{aligned} \tag{128}$$

The matrices $\boldsymbol{F}$ and $\boldsymbol{L}$ are found by [106]

$$\boldsymbol{F} = \partial \boldsymbol{f}(\boldsymbol{X}) / \partial \boldsymbol{X}\,|_{\hat{\boldsymbol{X}}} \tag{129}$$

$$\boldsymbol{L} = \partial \boldsymbol{f}(\boldsymbol{X}, \boldsymbol{W}) / \partial \boldsymbol{W}\,|_{\hat{\boldsymbol{X}}} \tag{130}$$

As shown in Figure 15, at the beginning of each horizon, MC based BICF trajectory optimization is used to rapidly generate the optimal trajectory for MAVs based on the initial and final conditions and the updated aerodynamic coefficients. At every sampling time of the control

loop, the calculated nominal control commands and state variables will be interpolated and fed into the LQR to compute the actual control commands for the MAV. The EKF will use the measured state variables and the control commands to estimate the aerodynamic coefficients, which are used to update the planning and LQR blocks at the beginning of the next horizon. In Figure 15, the sampling rates of EKF and LQR are assumed to be the same here, but the sampling rates of EKF needs to be higher than LQR in practice to be converged quickly. It is worth noting that the discretized trajectory commands generated in the planning block will be interpolated before being used in the LQR. In Figure 15, "~" represents updated parameters, "^" represents estimated or nominal values, and the parameters without any special notation represent real values.

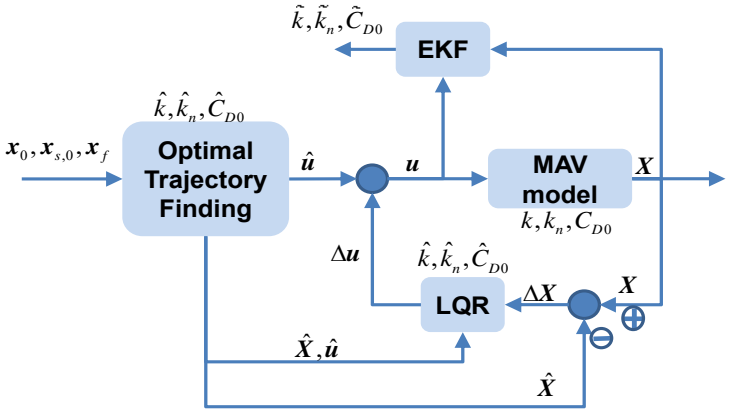



Figure 15 Outline of the rapid trajectory planning, estimation, and control in the receding horizon framework

## Simulation and Analysis

Three re-planning events are conducted in the simulations. The 2nd and 3rd re-planning are triggered at 30s and 60s, respectively. Without losing generality, all the obstacles in the

simulated urban area, such as buildings and trees, are assumed to be encircled by cylinders with different height and radius. The simulated urban area is shown in Figure 16. The cylindrical obstacles, numbered from 1 to 6, are detected before the 1st planning, while obstacles 7 to 12 and 13 to 14 are assumed to be detected right before the 2nd and the 3rd trajectory re-planning, respectively.

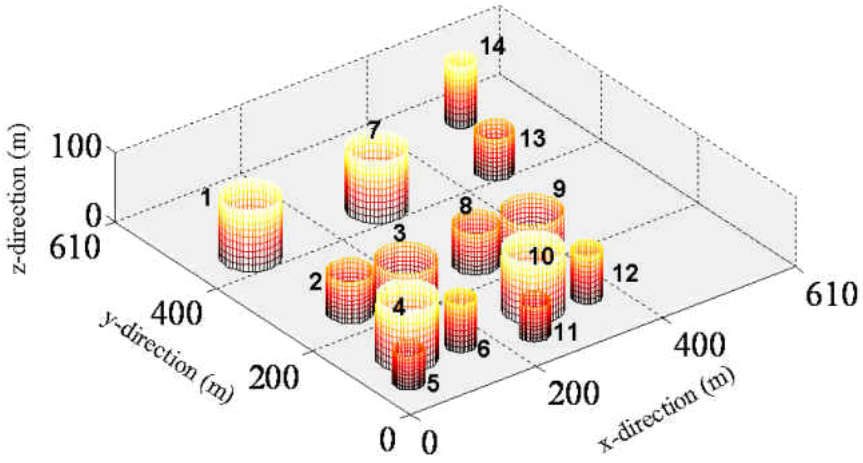



Figure 16 3D view of the simulated urban area

The MAV is commanded to fly from the initial position of $[1,1,20]^T\ m$ to the final position of $[600,600,50]^T\ m$ with an initial speed of $5 m/s$, a heading angle of $5^\circ$, and a flight path angle of $5^\circ$. The constraints of the MAV motion are that (i) the flight path angle is bounded by +/- $45^\circ$, (ii) the maximum speed is $10 m/s$, (iii) the maximum thrust is assumed to be $0.4N$, and (iv) the bank angle $\mu$ is within +/-$10^\circ$. The actual aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ are 0.015, 0.0158, and 0.4, respectively.

The following settings are used in the trajectory planning blocks: $n_{cp}=5$, $d=3$, and $N=15$.

$\nu(t) = 1$ and $\boldsymbol{x}_r = [-1500, -1500, -180]^T m$ (after trial and error) are used as the initial guesses of PCPs and reference point.

The weighting matrices in the LQ regulator are tuned to be $Q_l = 100 \cdot \boldsymbol{I}_6$ and $R_l = 0.1 \cdot \boldsymbol{I}_3$ ($\boldsymbol{I}$ is the identity matrix), so that the closed-loop system can closely follow the commanded trajectory and be robust with respect to uncertainties. The step sizes used in the EKF and LQ regulator are selected to be 0.001s. Because the EKF is designed to estimate the unknown aerodynamic coefficients $C_{d0}$, $k$, and $k_n$, the covariance matrices of the process noise and the measurement noise are small and are set as $\boldsymbol{Q} = diag\{[0.01, 0, 0]\}$ and $\boldsymbol{R} = diag\{[0.1, 0.1, 0.1, 0.001, 0.01, 0.01]\}$, respectively.

Part of the simulations results are shown here. In the 1st planning horizon, the aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ are initially guessed to be 0.03, 0.005, and 0.5, respectively. Figure 17 shows the rapidly generated optimal trajectories in the trajectory planning block with the actual flight trajectory achieved by the MAV after the control block overlapped. T LQ regulator can closely track the planned optimal trajectory and is robust with respect the parametric uncertainties, linearization, and discretization mismatches. The real MAV trajectory is shorter than the commanded one, because the optimal trajectory planning block computes the whole course for the MAV, while only the first section, around 30 s, is implemented by the MAV before the 2nd reconfiguration is triggered.

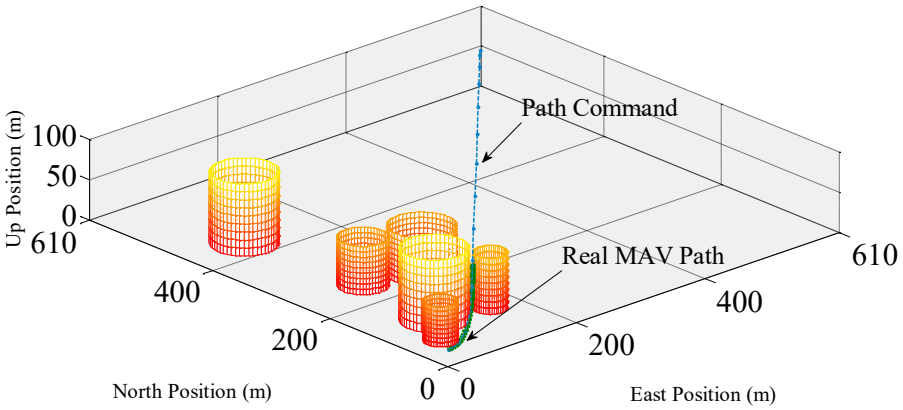



Figure 17 3D view of the optimal trajectory planned and the actual MAV trajectory achieved in the 1st horizon

Figure 18 shows the estimates of $C_{d0}$, $k$, and $k_n$ using the EKF . Initially $C_{d0}$, $k$, and $k_n$ equal to the guessed values, but quickly they converge to the actual values.

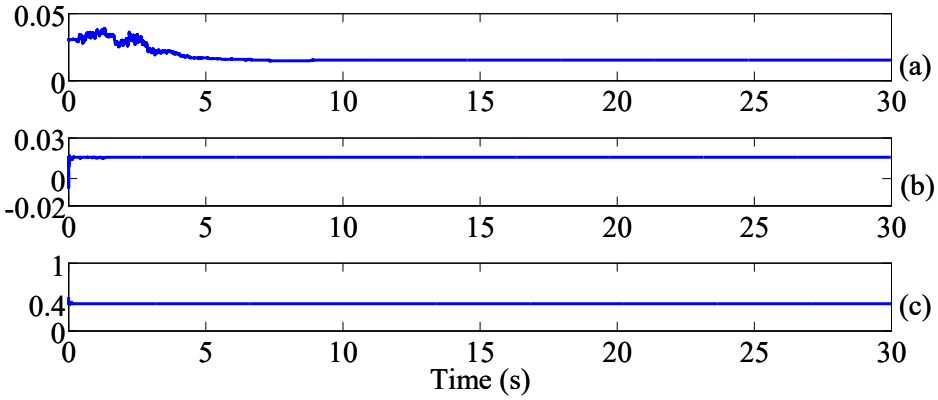



Figure 18 Aerodynamic coefficients' estimation via the EKF in the 1st horizon: (a) $C_{d0}$, (b) $k$, and (c) $k_n$

Figure 19 shows the optimal control commands generated by the trajectory planning block and the actual control command. The control calculated by the optimal path finding block is different from what is needed by the MAV in reality due to the discretization mismatches.

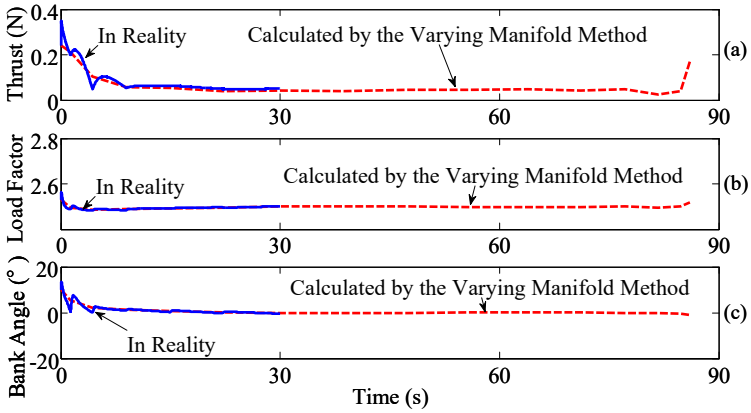



Figure 19 The control calculated by the optimal path finding block and the actual control commands for the MAV in the 1st horizon

The simulated results during the 2nd horizon which starts at 30s are shown in Figure 20 through Figure 22Figure 21. The aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ have been updated to the true values estimated using the EKF in the 1st horizon.

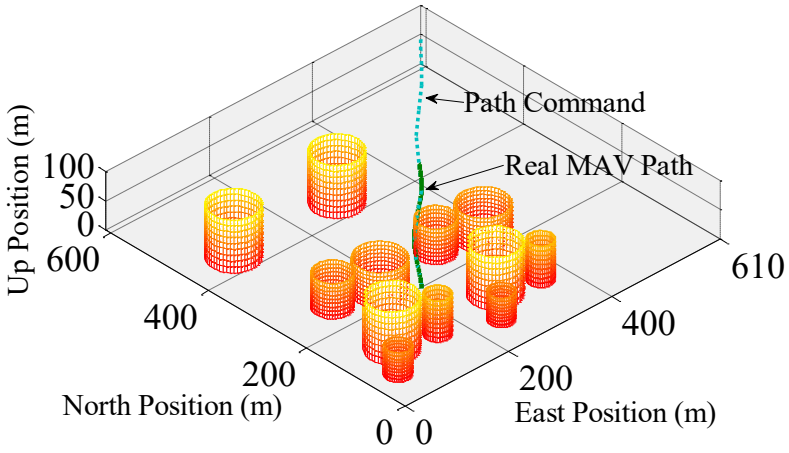



Figure 20 3D view of the optimal path command and the actual MAV path in the 2nd horizon

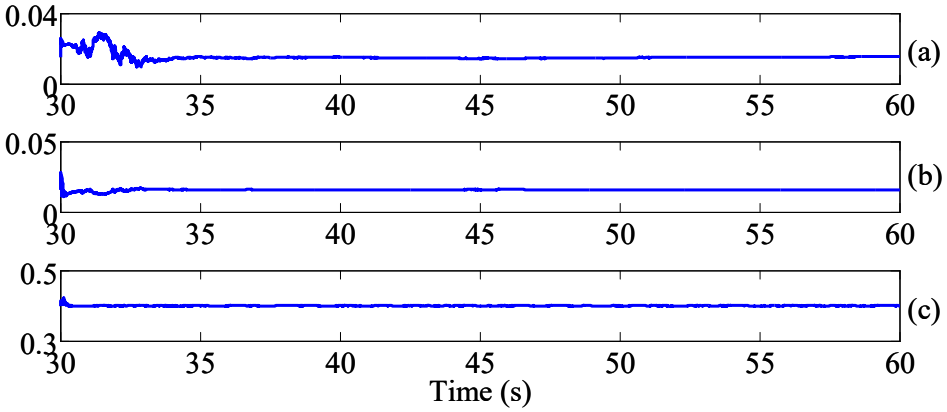



Figure 21 Aerodynamic coefficients estimated using the EKF in the 2nd horizon: (a) $C_{d0}$, (b) $k$, and (c) $k_n$

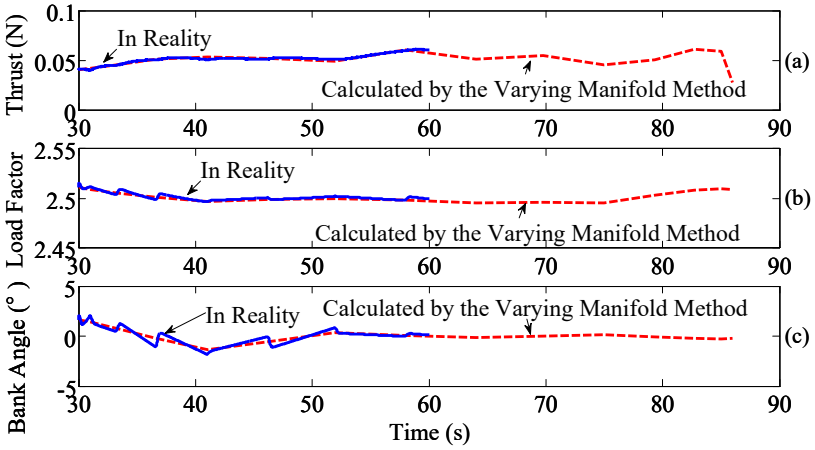



Figure 22 The control calculated by the optimal path finding block and the actual control commands for MAV in the 2nd horizon

The 3rd horizon is triggered at 60s in the simulation (Figure 23-Figure 25). The aerodynamic coefficients $C_{d0}$, $k$, and $k_n$ used in the optimal path finding block are updated using the final value generated from the EKF in the 2nd horizon.

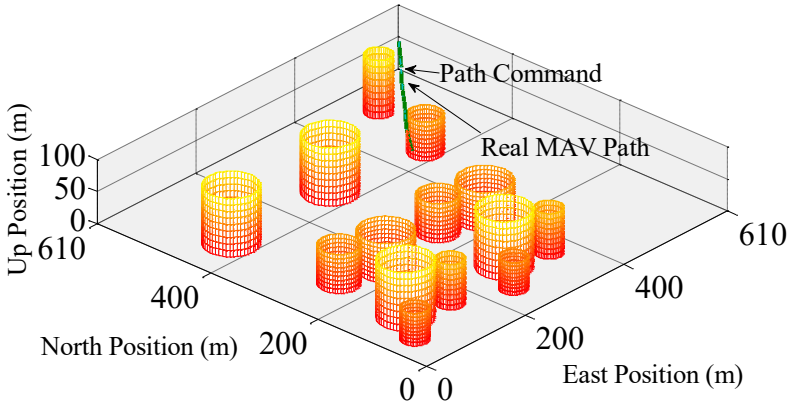



Figure 23 3D view of the optimal path command and the actual MAV path in the 3rd horizon

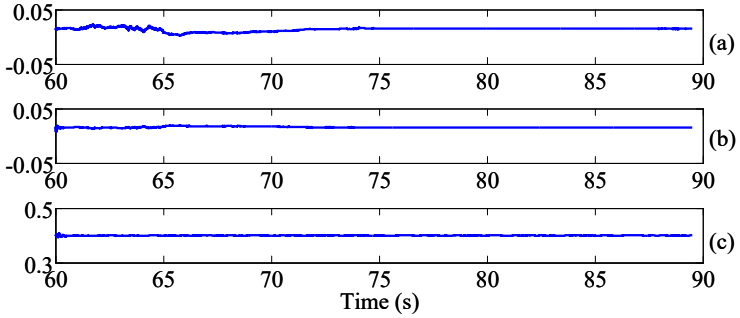



Figure 24 Aerodynamic coefficients estimation in the 3rd horizon: (a) $C_{d0}$; (b) $k$; and (c) $k_n$

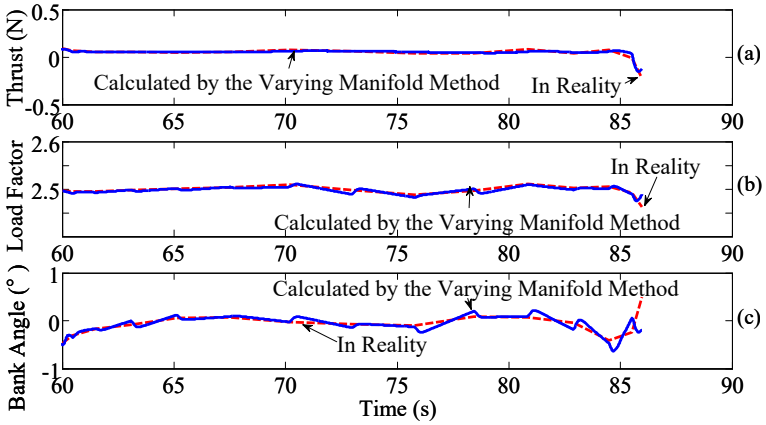



Figure 25 The control calculated by the optimal path finding block and the actual control command in the 3rd horizon

Table 12 shows the minimum arrival time calculated in three horizons and the CPU time used in calculating the optimal trajectories. It is worth mentioning that the minimum arrival time listed in the second and third horizons include the time lapsed in the previous horizons. Because not all of the obstacle are known *a priori*, the estimated arrival time calculated in these 3 horizon windows are different. Due to the time spent on detours to avoid the newly detected obstacles, the calculated minimum arrival time are different after updated in these three horizon windows. The CPU time using in the trajectory planning is 1.26 seconds in average, which shows the capability and efficiency of the proposed path planning method. Also the CPU time used in planning the 1st to 3rd horizons is reduced because of the distance to the desired position is reduced.

Table 12 The Minimum Time and Computation Cost in Trajectory Planning

| Horizon | Minimum Time | CPU Time in the Optimal Trajectory Planning |
|---|---|---|
| 1st Horizon | 88.54s | 1.65s |
| 2nd Horizon | 89.21s | 1.34s |
| 3rd Horizon | 89.25s | 0.78s |

# CHAPTER SIX: SUMMARY AND FUTURE WORK

In this dissertation, the bio-inspired, varying manifold based optimization methods are studied to rapidly solve nonlinear constrained trajectory optimization problems.

The proposed trajectory optimization methods utilize predator-prey relations, in which a vehicle (e.g. ground robot, micro air vehicle, or unmanned air vehicle) is considered as a "predator" while the prey is represented by a virtual path (can be an initial guess). The vehicle's trajectory is calculated in a simultaneously optimized manifold constructed by the virtual prey path and possibly other constant but optimizable parameters. Three "predator-prey" relationships: motion camouflage, local pursuit, and constant absolute target direction, are unified and investigated. Compared to some other traditional trajectory optimization methods, the computational cost of proposed method is lower.

Strategies to enhance the initial guess of optimizable variables are derived based on the necessary conditions for a vehicle to satisfy both obstacle avoidance and speed constraints. These strategies can enhance the convergent rate and reduce the computational time of achieved NLP problem.

The optimal trajectory generation simulations for a supersonic aircraft, ground robot, and micro air vehicle are conducted to show the capabilities of the proposed method. Specifically, Monte-Carlo simulations are conducted on a robot obstacle avoidance problem and a micro air vehicle 3D minimum-time flight problem. The first Monte Carlo simulation is used to compare the performance of the proposed methods, baseline optimization method, and B-spline based collocation method; while the second is to show the effectiveness of the proposed initial guess strategies.

To deal with the uncertainties and pop-up obstacles, the bio-inspired optimal trajectory planning method is embedded into a receding horizon framework. This method is applied to a micro air vehicle flight problem. In each planning horizon, the aerodynamic coefficients are updated and the micro air vehicle's optimal trajectory is planned. Meanwhile, a linear quadratic regulator is designed for the micro air vehicle to track the generated nominal path and be robust with respect to uncertainties. An extended Kalman filter is used to estimate the unknown aerodynamic coefficients for the next planning horizon.

The following conclusions can be drawn from this dissertation: (1) the bio-inspired trajectory optimization methods reduce the computational cost; and (2) the enhanced initial guess techniques increase the convergence rate and reduce the computational cost of the studied method.

Currently, the bio-inspired trajectory optimization methods rely on the assumption that the control variables and state rates of vehicles can be represented using the position state explicitly. In the future, research needs to be done on how to implement the proposed trajectory generation methods to complex systems where the dynamics cannot be fully inverted. Furthermore, simulation results show that the trajectory optimization method based on the local pursuit strategy has a relatively higher success rate compared to the other two approaches (i.e. the motion camouflage strategy and constant absolute target direction strategy). Therefore the advantages and disadvantages of these three strategies will be further studied theoretically. In addition, the bio-inspired trajectory optimization methods are only applied on a single vehicle. Future research can be conducted to apply the proposed methods on multiple-vehicle control

problems. The virtual prey can act as a virtual leader, while the other vehicles in the group can be regarded as the predators.

# LIST OF REFERENCES


[1] Richards, A., Schouwenaars, T., How, J. P., and Feron, E., "Spacecraft Trajectory Planning with Avoidance Constraints Using Mixed Integer Linear Programming," *Journal of Guidance, Control and Dynamics*, Vol. 25, No. 4, 2002, pp. 755-764.

[2] Jacobsen, S., Lee, C., Zhu, C. and Dubowsky, S., "Planning of Safe Kinematic Trajectories for Free Flying Robots Approaching an Uncontrolled Spinning Satellite," Proceedings of ASME Design Engineering Technical Conference and Computer and Information in Engineering Conference, Montreal, Canada, September 29-October 2, 2002.

[3] Alejo, D., Cobano, J. A., Heredia, G., Ollero, A., "Collision-free 4D Trajectory Planning in Unmanned Aerial Vehicles for Assembly and Structure Construction," *Journal of Intelligent and Robotic Systems*, 2013, DOI: 10.1007/s10846-013-9948-x.

[4] Zhao, Y. M., and Tsiotras, P., "Time-optimal Path Following for Fixed-Wing Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No.1, 2013, pp. 83-95.

[5] Li, N., Xu, Y., and Pham, K. D., "Micro Air Vehicle's 3D Trajectory Planning and Parametric Estimation," Proceedings of AIAA Guidance, Navigation and Control Conference, Boston, MA, USA, August 19-22, 2013, DOI: 10.2514/6.2013-4764.

[6] Flores, G., Zhou, S., Lozano, R., and Castillo P., "A Vision and GPS-Based Real-Time Trajectory Planning for a MAV in Unknown and Low-Sunlight Environments," *Journal of Intelligent and Robotic Systems*, 2013, DOI: 10.1007/s10846-013-9975-7.

[7] Remeikas, C., Li, N., Xu, Y., Jayasuriya, S., and Ehsani, R., "Task Assignment and Trajectory Planning Algorithm for a Class of Cooperative Agricultural Robots", submitted

to the *ASME Journal of Dynamics Systems, Measurement, and Control*, August 2013.

[8] Chrpa, L., and Osborne, H., "Towards a Trajectory Planning Concept: Augmenting Path Planning Methods by Considering Speed Limit Constraints," *Journal of Intelligent and Robotic Systems*, 2013, DOI: 10.1007/s10846-013-9886-7

[9] Gal, O., "Unified Trajectory Planning Algorithms for Autonomous Underwater Vehicle Navigation," *ISRN Robotics*, 2013, Article ID: 329591.

[10] Pfeiffer, F., and Johanni, R., "A Concept for Manipulator Trajectory Planning*," IEEE Journal of Robotics and Automation*, Vol. RA-3, No.2, 1987, pp.115-123

[11] Akira, A., "Minimum Energy Trajectory Planning Method for Robot Manipulator Mounted on Flexible Base," 9th Asian Control Conference, Istanbul, Turkey, June 23-26, 2013, pp. 1-7.

[12] Masajedi, P., Shirazi, K. H., and Ghanbarzadeh, A., "3D Trajectory Planning for a 6R Manipulator Robot Using BA and ADAMS," *Information Technology Convergence Lecture Notes in Electrical Engineering*, Vol. 253, 2013, pp. 807-816.

[13] Shamir, R. R., Tamir, I., Dabool, E., Joskowicz, L., and Shoshan, Y., "A Method for Palling Safe Trajectories in Image-Guided Keyhole Neurosurgery," *Medical Image Computing and Computer Assisted Intervention*, Vol. 13, Part. 3, 2010, pp. 457-464.

[14] Betts, J., "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 2, pp. 193-207, 1998.

[15] Nusyirwan, I. F., and Bil, C., "Effect of Uncertainties on UCAV Trajectory Optimization using Evolutionary Programming," IEEE Information, Decision, and Control Conference, Adelaide, Australia, 2007, pp. 219-223.

[16] Aydin, S., and Temeltas, H., "Fuzzy-Differential Evolution Algorithm for Planning Time-Optimal Trajectories of a Unicycle Mobile Robot on a Predefined Path," *Advanced Robotics*, Vol. 18, No. 7, 2004, pp. 725-748.

[17] Merchan-Cruz, E. A., and Morris, A. S., "Fuzzy-GA-based Trajectory Planner for Robot Manipulators Sharing a Common Workspace," *IEEE Transactions on Robotics*, Vol. 22, No. 4, 2006, pp. 613-624.

[18] Zhang, D., "Space Vehicle Orbit Transfer Optimization Based on Direct Transcription and Simulated Annealing Genetic Algorithm," *IEEE Conference on Intelligent Computing and Intelligent Systems*, Shanghai, China, 2009, pp. 599-602.

[19] Masehian, E., and Amin-Naseri, M. R., "Sensor-based Robot Motion Planning – A Tabu Search Approach," *IEEE Robotics & Automation Magazine*, Vol. 15, No. 2, 2008, pp. 48-57.

[20] Zhang, Q., Liu, C., Yang, B., and Ren, Z., "Reentry trajectory planning optimization based on ant colony algorithm," IEEE International Conference on Robotics and Biomimetics, Sanya, China, pp.1064-1068, 2007.

[21] Parsapoulos, K. E., and Vrahatis, M. N., "Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization," *Natural Computing*, Vol. 1, No. 2-3, 2002, pp. 235-306.

[22] Clerc, M., and Kennedy, J., "The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Transaction on Evolutionary Computation*. Vol. 6, No. 1, 2002, pp. 58-73.

[23] Curkovic, P. and Jerbic, B., "Honey-bees optimization algorithm applied to path planning problem," *International Journal of Simulation Modeling*, Vol. 6, No. 3, 2007, pp. 154-164.

[24] Ali, M. M., and Gabere, M. N., "A Simulated Annealing Driven Multi-Start Algorithm for Bound Constrained Global Optimization," *Journal of Computational and Applied Mathematics*, Vol. 233, No. 10, 2009, pp. 2661-2674.

[25] Celeste, F., Dambreville, F., and Le Cadre, J. P., "Optimal Path Planning Using Cross-Entropy Method," 9th International Conference on Information Fusion, Florence, Italy, pp. 1-8, 2006.

[26] Tangpattanakul, P., Meesomboon, A., and Artrit, P., "Optimal Trajectory of Robot Manipulator Using Harmony Search Algorithm," *Recent Advances in Harmony Search Algorithm, Studies in computational Intelligence,* Vol. 270, 2010, pp. 23-26.

[27] Mehrabian, A. R., and Lucas, C., "A Novel Numerical Optimization Algorithm Inspired from weed Colonization," *Ecological Informatics*, Vol. 1, No. 4, 2006, pp. 335-366.

[28] Yokoyama, N., and Susuki, S., "Modified Genetic Algorithm for Constrained Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 1, 2005, pp. 139-144.

[29] Bliek, C., Spellucci, P., Vicente, L. N., Neumaier, A., Granvilliers, L., Huens, E., Hentenryck, P. V., Sam-Haroud, D., and Faltings, B., "Algorithms for Solving Nonlinear Constrained and Optimization Problems," Coconut Project Report, 2001.

[30] Bomze, I. M., Csendes, T., Horst, R., and Pardalos, P. M (Eds.), *Developments in Global Optimization*, Edited by Kluwer Academic Publishers, Dordrecht, Netherlands: Kluwer, 1997.

[31] Horst, R. and Pardalos, P.M. (Eds.) *Handbook of Global Optimization*, Dordrecht, Netherlands: Kluwer, 1995.

[32] Horst, R. and Tuy, H., *Global Optimization: Deterministic Approaches*, 3rd ed. Berlin: Springer-Verlag, 1996.

[33] Kearfott, R. B., *Rigorous Global Search: Continuous Problems*. Dordrecht, Netherlands: Kluwer, 1996.

[34] Stryk, O. V., and Bulirsch, R., "Direct and Indirect Methods for Trajectory Optimization," *Annals of Operations Research*, Vol. 37, 1992, pp. 357-373.

[35] Naidu, D. S., *Optimal Control Systems*, CRC press, 2003.

[36] Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., and Mishchenko, E. F., *The Mathematical Theory of Optimal Processes*, Wiley-Interscience, New York, NY, 1962.

[37] Bryson, A. E., and Ho, Y. C., *Applied Optimal Control, Hemisphere*, New York, 1975.

[38] Miele, A., Gradient Algorithms for the Optimization of Dynamic Systems, in: *Control and Dynamic Systems*, ed. Leondes, C. T., Vol. 16, pp. 1-52, 1980.

[39] Lawden, D. F., "Rocket trajectory optimization: 1950 – 1963," *Journal of Guidance, Control, and Dynamics*, Vol.14, No. 4, 1991, pp. 705-711.

[40] Qu, Z., Wang, J., and Plaisted, C. E., "A New Analytical Solution to Mobile Robot Trajectory Generation in the Presence of Moving Obstacles," *IEEE Transactions on Robotics*, Vol. 20, No. 6, December 2004, pp. 978-993.

[41] Ocampo, C., "Finite Burn Maneuver Modeling for a Generalized Spacecraft Trajectory Design and Optimization System," *Annuals of the New York Academy of Sciences*, Vol. 1017, 2004, pp. 210-233.

[42] Jacobson, D. H., and Lele, M. M., "A Transformation Technique for Optimal Control Problems with a State Variable Inequality Constraint," *IEEE Transaction on Automatic*

*Control*, Vol. 14, No. 5, 1969, pp. 457-464.

[43] Mehra, R. K., and Davis, R. E., "A Generalized Gradient Method for Optimal Control Problems with Inequality Constraints and Singular Arcs," *IEEE Transactions on Automatic Control*, Vol. 17, No. 1, 1972, pp. 69-79.

[44] Hartl, R. F., Sethi, S. P., and Vickson, R. G., "A Survey of the Maximum Principles for Optimal Control Problems with State Constraints," *SIAM Review*, Vol. 37, No.2, 1995, pp. 181-218.

[45] Betts, J., *Practical Methods for Optimal Control Using Nonlinear Programming*, Society for Industrial and Applied Mathematics, Philadelphia, 2001.

[46] Dai, R., "B-splines based Optimal Control Solution," 2010 AIAA Guidance, Navigation, and Control Conference, Toronto, Ontario Canada, Paper # 2010-7888, August 2-5, 2010.

[47] Yakimenko, O., "Direct Method for Rapid Prototyping of Near-Optimal Aircraft Trajectories," *Journal of Guidance, Control, and Dynamics,* Vol. 23, No. 5, September-October 2000, pp. 865-875.

[48] Yakimenko, O., "Implementation of the Direct Method of Variational Calculus for Aircraft's Spatial Maneuvers Rapid Prototyping," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Portland, OR, 9-11 August, 1999.

[49] Fahroo, F., and Ross, I. M., "Costate Estimation by a Legendre Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, pp. 270-275, 2001.

[50] Kameswaran, S., and Biegler, L. T., "Convergence Rates for Direct Transcription of Optimal Control Problems using Collocation at Radau Points," *Computational Optimization and Applications*, Vol. 41, No. 1, pp. 81-126. 2008

[51] Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Rao, A. V., "Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 6, pp.1435-1440, 2006.

[52] Gong, Q., Ross, I. M., Kang, W., and Fahroo, F., "Connections Between the Covector Mapping Theorem and Convergence of Pseudospectral Methods for Optimal Control," *Computational Optimization and Applications*, Vol. 41, No. 3, pp. 307-335, 2008.

[53] Jackiewicz, Z., and Welfert, B. D., "Stability of Gauss-Radau Pseudospectral Approximations of the One-Dimensional Wave Equation," *Journal of Scientific Computing*, Vol. 18, No. 2, 2003, pp. 287-313.

[54] Fahroo, F. and Ross, I., "Direct Trajectory Optimization by a Chebyshev Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 160-166.

[55] Seywald, H. "Trajectory Optimization Based on Differential Inclusion," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 3, May-June 1994, pp. 480-487.

[56] Yang, B., and Sun, S., "Reentry Trajectory Optimization of Airbreathing Hypersonic Vehicles Based on Gauss Pseudospectral Method," *Advanced Materials Research*, November 2011, pp. 383-390.

[57] Hargraves, C. R., and Paris, S. W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *Journal of Guidance, Control, and Dynamics*, Vol. 10, No. 4, 1987, pp. 338-342.

[58] Hager, W., "Runge-Kutta Methods in Optimal Control and the Transformed Adjoint System," *Numerische Mathematik*, Vol. 87, No. 2, 2000, pp. 247-282.

[59] Haddad, M., Khalil, W., and Lehtihet, H. E., "Trajectory Planning of Unicycle Mobile

Robots with a Trapezoidal-Velocity Constraint," *IEEE Transactions on Robotics*, Vol. 26, No. 5, October 2010, pp. 954-962.

[60] Xu, Y., and Basset, G., "Sequential Virtual Motion Camouflage Method for Nonlinear Constrained Optimal Trajectory Control," *Automatica*, Vol. 8, No. 7, 2012, pp. 1273-1285.

[61] Bazaraa, M. S., Sherali, H. D., and Shetty, C. M., *Nonlinear Programming Theory and Application*, *3rd Edition*, John Wiley & Sons, Inc. New Jersey, 2006.

[62] Betts, J. T., and Frank, P. D., "A Sparse Nonlinear Optimization Algorithm," *Journal of Optimization Theory and Applications*, Vol. 82, No. 3, 1994, pp. 519-541.

[63] Davis, P. J., and Rabinowitz, P., Methods of Numerical Integration, Academic, New York, 1977.

[64] Xu, Y., and Li, N., "Biologically Inspired Computational Framework for a Class of Nonlinear Constrained Optimal Trajectory Planning", submit to *Bioinspiration & Biomimetics*, under review (1st revision).

[65] Basset, G., Xu, Y. and Li, N., "Fast Trajectory Planning via the B-spline Augmented Virtual Motion Camouflage Approach," *ASME Journal of Dynamic Systems, Measurement, and Control*, doi:10.1115/1.4024601.

[66] Piegl, L., and Tiller, W., *The NURBS Book: Second Edition*, Springer-Verlag, New York, 1997.

[67] H. Parutzsch, W. Boehm, and M. Paluszny, *Bezier and B-spline Techniques*, Springer-Verlag Berlin Heidelberg, 2002.

[68] Camazine, S., Deneubourg, J., Franks, N. R., Sneyd, J., Theraulaz, G., and Bonabeau, E., *Self-organization in biological systems*, New Jersey: Princeton University Press, 2010.

[69] Srinivasan, M. V., and Davey, M., "Strategies for Active Camouflage Motion," Proceedings of the Royal Society of London Biological Sciences, Vol. 259, No. 1354, pp. 19-25, 1995.

[70] Olberg, R. M., Worthington, A. H., and Venator, K. R., "Prey Pursuit and Interception in Dragonflies," *Journal of Comparative Physiology A: Sensory Neural and Behavioral Physiology*, Vol. 186, No. 2, 2000, pp. 155-162.

[71] Rafie-Rad, M., "Time-optimal Solution of Parallel Navigation and Finsler Geodesics," arXiv:1101.1537v1, 2011.

[72] Ghose, K., Horiuchi, T. K., Krishnaprasad, P. S., and Moss, C. F., "Echolocating Bats Use a Nearly Time-optimal Strategy to Intercept Prey," *PLoS Biology*, Vol. 4, No. 5, 2006, pp. 865-873.

[73] Hristu-Varsakelis, D., and Shao, C., "Biologically-inspired Optimal Control: Learning from Social Insects," *International Journal of Control*, Vol. 77, No.18, 2004, pp. 1549-1566.

[74] Hristu-Varsakelis, D., and Shao, C., "A Bio-inspired Pursuit Strategy for Optimal Control with Partially Constrained Final State," *Automatica,* Vol. 43, No. 7, 2007, pp. 1265-1273.

[75] Kumar, R. R., and Seywald, H., "Should Controls Be Eliminated While Solving Optimal Control Problems via Direct Methods?," *Journal of Guidance, Control and Dynamics*, Vol. 19,No. 2, 1996, pp. 418-423.

[76] Laumond, J. P., Sekhavat, S., and Lamiraux, F., *Guidelines in Nonholonomic Motion Planning for Mobile Robots*, Lecture Notes in Control and Information Sciences, LNCIS 229, New York: Springer-Verlag, 1998.

[77] Btavia, P. H., and Nourbakhsh, I., "Path Planning for the Cye Personal Robot," Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, Pittsburgh, PA,

USA, pp. 15-20, 2000.

[78] Dai, R., & Cochran, J. E., "Three-dimensional Trajectory Optimization in Constrained Airspace," *Journal of Aircraft*, Vol. 46, No. 2, 2009, pp. 627-634.

[79] Ridder, S. D., 2009 Study on optimal trajectories and energy management capabilities of a winged re-entry vehicle during the terminal area MS Thesis, Department of Aerospace Engineering, Delft University of Technology, Delft, NL

[80] Bryson, A. E., Desai, M. N., & Hoffman, W. C., "Energy-state Approximation in Performance Optimization of Supersonic Aircraft," *Journal of Aircraft*, Vol. 6, No. 6, 1969, pp. 481-488.

[81] Gath, P. F., Well, K. H., and Mehlem, K., "Initial Guess Generation for Rocket Ascent Trajectory Optimization Using Indirect Mehothods*," Journal of Spacecraft and Rockets*, Vol. 39, 2002, pp. 515-521.

[82] Lee, D., Bang, H., and Lee, W., "Fuel Optimal, Low-thrust Earth-to-Moon Trajectories Design Using the Initial Guess Structure," AIAA Guidance, Navigation and Control Conference, Toronto, Canada, August 2-5, 2010.

[83] Bakolas, E., Zhao, Y., and Tsiotras, P., "Initial Guess Generation for Aircraft Landing Trajectory Optimization," AIAA Guidance, Navigation and Control Conference, Portland, Oregon, August 7-11, 2011

[84] Washington, A. J., *Basic Technical Mathematics with Calculus*, 2000, Addison Wesley longman, Inc.

[85] Mohr, B. B., and Fitzpatrick, D. L., "Micro Air Vehicle Navigation System," *IEEE Aerospace and Electronic Systems Magazine*, Vol. 23, No. 4, 2008, pp. 19-24.

[86] Yang, K., and Sukkarieh, S., "3D Smooth Path Planning for a UAV in Cluttered Natural Environments," IEEE International Conference on Intelligent Robots and Systems, Nice, France, September 22-26, pp. 794-800, 2008.

[87] Grzywna, J. W., Plew, J., Nechyba, M. C., and Ifju, P. G., "Enabling Autonomous MAV Flight," 16th Florida Conference on Recent Advances in Robotics, 2003.

[88] Saunders, J. B., Call, B., Curtis, A., Beard, R. W., and McLain, T. W., "Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles," Proceedings of AIAA Infotech@Aerospace Conferences, Arlington, VA, USA, September 25-29, AIAA-2005-6950, 2005.

[89] Goerzen, C., Kong, Z., and Mettler, B., "A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance," *Journal of Intelligent and Robotic Systems*, Vol. 57, No. 1-4, 2010, pp. 65-100.

[90] Green, W. E., and Oh, P. Y., "A Hybrid MAV for Ingress and Egress of Urban Environments," *IEEE Transactions on Robotics*, Vol. 25, No. 2, pp. 253-263, 2009.

[91] Zbikowski, R., "Fly Like a Fly [Micro-Air Vehicle]," *IEEE Spectrum*, Vol. 42, No. 11, 2005, pp. 46-51.

[92] Ifju, P. G., Jenkins, D. A., Ettinger, S., Lian, Y., Shyy, W., and Waszak, M. R., "Flexible-Wing-based Micro Air Vehicles," Proceedings of 40th AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV, USA, January 14-17, AIAA 2002-0705, 2002.

[93] Beard, R., Kingston, D., Quigley, M., Snyder, D., Christiansen, R., Johnson, W., McClain, T., and Goodrich, M. A., "Autonomous Vehicle Technologies for Small Fixed-wing UAVs," *Journal of Aerospace Computing, Information, and Communication*, Vol. 2, 2005, pp. 92-108.

[94] Zhu, R., Guan, X., zhou, Z., and Sun, D., "Collision-free Path Planning and Trajectory Generation for MAVs Flying in Urban Terrain," International Conference on Intelligent Robot and Systems, Beijing, China, October 9-15, pp. 2888-2893, 2006.

[95] Beard, R. W., Mclain, T. W., Goodrich, M. A., and Anderson, E. P., "Coordinated Target Assignment and Intercept for Unmanned Air Vehicles," *IEEE Transactions on Robotics and automation*, Vol. 18, No. 6, 2002, pp. 911-922.

[96] McLain, T., and Beard, R., "Cooperative Rendezvous of Multiple Unmanned Air Vehicles," AIAA Guidance, Navigation and control Conference, Denver, CO, USA, August, pp. 2000-4369, 2000.

[97] Yang, K. and Sukkarieh, S., "3D Smooth Path Planning for a UAV in Cluttered Natural Environments," IEEE/RSJ International Conference on Intelligent Robots and Systems, Acropolis Convention Center, Nice, France, September 22-26, pp. 794-800, 2008.

[98] Ceccarelli, N., Enright, J. J., Frazzoli, E., Rasmussen, S. J., and Schumacher, C. J., "Micro UAV Path Planning for Reconnaissance in Wind," American Control Conference, New York, USA, July 11-13, pp. 5310-5315, 2007,

[99] Bortoff, S. A., "Path Planning for UAVs," American Control Conference, Chicago, Illinois, June, 2000.

[100] Prazenica, R., Kurdila, A., Sharpley, R., and Evers, J., "Multi-resolution and Adaptive Path Planning for Maneuver of Micro-air-vehicles in Urban Environments," AIAA Guidance Navigation and Control Conference, San Francisco, California, August 15-18, 2005, pp.1761-1772

[101] Dong, Z., Chen, Z., Zhou, R., and Zhang, R., "A Hybrid Approach of Virtual Force and

A* Search Algorithm for UAV Path Re-planning," IEEE Conference on Industrial Electronics and Applications, Beijing, China, June 21-23, 2011, pp. 1140-1145.

[102] Jaroszewicz, A., Garbowski, M., Sibilski, K., and Zyluk, A., "Estimation of MAV Unsteady Aerodynamics Parameters From Dynamic Water Tunnel Testing," 49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, January 4-7, 2011, AIAA 2011-1162.

[103] Unnikrishnan, N., and Balakrishnan, S. N., "Neuroadaptive Model Following Controller Design for a Nonaffine UAV model," Proceedings of the American Control Conference, Minneapolis, MN, USA, June 14-16, pp. 2951-2956, 2006.

[104] Mueller, T. J., *Fixed and Flapping Wing Aerodynamics for Micro Air Vehicle Applications*, AIAA Publisher, 2001.

[105] Slotine, J-J. E., and Li, W., *Applied Nonlinear Control*, Prentice Hall, Englewood Cliffs, NJ, 1991.

[106] Simon, D., *Optimal State Estimation*, John Wiley & Sons, Hoboken, NJ, 2006.