2019 18th European Control Conference (ECC)
Napoli, Italy, June 25-28, 2019

Autonomous Waypoint Planning, Optimal Trajectory Generation and Nonlinear Tracking Control for Multi-rotor UAVs

Hossein Eslamiat[1,*] Yilan Li [2,*] Ningshan Wang[1] Amit K. Sanyal[1] Qinru Qiu[2]

***Abstract*— A framework for autonomous waypoint planning, trajectory generation through waypoints, and trajectory tracking for multi-rotor unmanned aerial vehicles (UAVs) is proposed in this work. Safe and effective operations of these UAVs is a problem that demands obstacle avoidance strategies and advanced trajectory planning and control schemes for stability and energy efficiency. To address this problem, a two-level optimization strategy is used for trajectory generation, then the trajectory is tracked in a stable manner. The framework given here consists of the following components: (a) a deep reinforcement learning (DRL)-based algorithm for optimal waypoint planning while minimizing control energy and avoiding obstacles in a given environment; (b) an optimal, smooth trajectory generation algorithm through waypoints, that minimizes a combinaton of velocity, acceleration, jerk and snap; and (c) a stable tracking control law that determines a control thrust force for an UAV to track the generated trajectory.**

I. Introduction

Autonomous waypoint planning, trajectory generation and tracking for a certain class of underactuated maneuvering vehicles, namely multi-rotor UAVs, is considered here. Obstacle-free trajectory planning is particularly important in beyond visual line-of-sight (BVLOS) operations, and applications that require unmanned vehicles to move in cluttered environments [1]. Such applications include indoor operations [2], package delivery in urban and suburban areas, monitoring of civilian infrastructures like bridges and highways, autonomous landing on moving platforms and tracking wildlife in forested areas.

This paper investigates the problem of planning waypoints and generating a time trajectory for the position of an underactuated vehicle that has four independent control inputs for the six degrees of freedom of translational and rotational motion in three dimensional Euclidean space. The control inputs actuate the three degrees of rotational motion and one degree of translational motion in a vehicle body-fixed coordinate frame. The translational motion is controlled by a single thrust along a body-fixed direction vector, while this body-fixed thrust vector can be controlled by controlling the attitude (orientation) of the vehicle. This actuation model covers a wide range of unmanned vehicles like fixed-wing and quadcopter UAVs, unmanned underwater vehicles and spacecraft. The position trajectory is generated by utilizing a two-level optimization strategy.

[1]Department of Mechanical and Aerospace Engineering, Syracuse University, Syracuse, NY 13244, USA {heslamia,nwang16,aksanyal}@syr.edu

[2]Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY 13244, USA {yli41,qiqiu}@syr.edu

* These authors contributed equally to this work.

This work is partially supported by the National Science Foundation under Grant CNS-1739748.

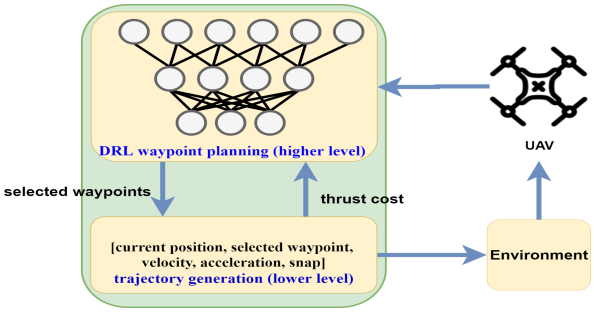

Fig. 1: Overall structure of proposed framework

Goal of the higher level is to use deep learning technique to generate waypoints with minimal thrust cost. Deep learning was developed as a machine learning approach to learn patterns from data, which simulates the activity of the human brain [3]. Its excellent capabilities of learning representations from dynamics of a real environment makes it extremely suitable for many autonomous robotic tasks, especially tasks dealing with complex input-output mappings. Deep neural networks (DNN) were used in [4], [5] to achieve adaptability and robustness to guarantee stable flight. However, they are likely to generate sub-optimal solutions without considering the future effect of the chosen action. Reinforcement learning (RL) provides a mathematical framework for deriving strategies that map situations (i.e. states) into actions with the goal of maximizing an accumulative reward [6]. In RL, the aim of the agent (i.e. learner) is to maximize the cumulative reward by taking the proper action at each time step according to the current state of the environment, while considering the trade-off between exploration and exploitation. Originally developed by DeepMind Technologies, deep RL (DRL) provides a promising data-driven, adaptive technique in handling large space of transitions for complex control systems [7].

On the lower level, the generated waypoints are used to construct an optimal trajectory that passes through them, in a discrete time setting. Figure 1 shows the overall structure of the proposed framework. After the trajectory is generated, it is tracked by a nonlinear stable algorithm. Prior research on trajectory tracking control with this actuation model includes [8], [9], [10]. Past research on trajectory generation can be classified into three types. One type decouples time and geometry, constructs a geometric trajectory and then parameterizes it in time, e.g., [11], [12], [13]. Utilizing Bezier curves to create trajectories also fall into this first type [14]. Another type utilizes differential flatness of dynamics to generate a

trajectory [15]. The last type uses a high level trajectory generator in conjunction with a motion primitive generator to choose an optimized trajectory among different motion primitives [16]. Trajectory planning can also be considered a smoothing problem; however, this problem is ill-posed [17] and in addition, needs all the waypoints to be able to create a trajectory, whereas in real life applications, all waypoints might not be available a priori. The actuation of these vehicles dictates that the position trajectory be generated through control of the attitude of the vehicle, such that the desired thrust vector direction for the position control is achieved. Trajectory planning schemes for quadrotor UAVs, which satisfy this actuation model, have been treated in the past, for example, in [8], [18]. For large and rapid maneuvers that go beyond hovering or level flight, this can be done using an asymptotically stable or finite-time stable attitude control scheme with a large domain of convergence, as in [19].

Section II details the structure of our proposed model, learning process and different ways to select training data for learning and waypoints generation. Section III provides details about dynamics model and our proposed trajectory generation scheme through waypoints. Section IV describes the tracking control of the proposed position trajectory and section V presents numerical simulation results. Finally, the research findings of this paper and possible future work are summarized in section VI.

II. Waypoint Planning

A. Problem Formulation

The goal is to select waypoints avoiding obstacles in a known 3D environment, while minimizing the control energy over the entire trip. The UAV takes off from an initial position and reaches a preassigned target position by following planned waypoints, without colliding with obstacles. The entire 3D environment is discretized into $N \times N \times N$ grids. Each grid $p(x, y, z)$ is mapped to a real value R by a function $\mathbf{M}()$, which describes the entire grid environment. W denotes the set of k planned waypoints, $W = \{w_0, w_1, ..., w_i, ..., w_{k-1}\}$ and w_{k-1} is the destination. $f(w_i, w_j)$ denotes the average control thrust for the UAV to follow the trajectory between waypoints w_i and w_j. $\mathbf{G}(w_i, w_j)$ denotes the set of grids that the generated trajectory between w_i and w_j will pass through. The total thrust cost along the entire trajectory is denoted as $\mathbf{F}$. The optimal waypoints planning problem can be formulated as follows:

Problem 1 (Optimal obstacle avoidance waypoints planning).

$$\text{Minimize: } \mathbf{F} = \Sigma_{i=0}^{k-2} f(w_i, w_{i+1}) \tag{1}$$

subject to

1) reaching the target position from current position,

$$\mathbf{M}(w_0) = 1, \mathbf{M}(w_{k-1}) = 10, \tag{2}$$

2) reaching the target position without colliding with obstacles,

$$\mathbf{M}(p) \neq -10, p \in \mathbf{G}(w_i, w_{i+1}), 0 \leq i \leq k-2, \tag{3}$$

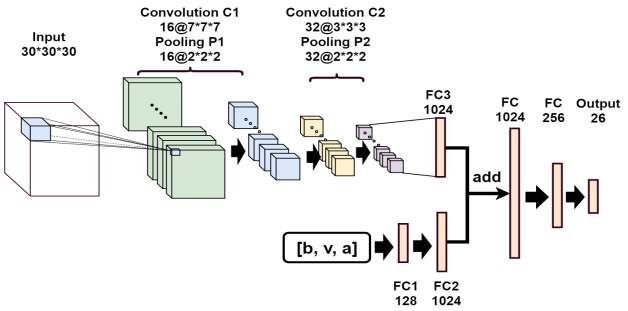

Fig. 2: Overall network structure of DQN

Here p is a grid in the discretized environment, w_i is a vector corresponding to the grid center position (x, y, z) in discretized environment block, and $f(w_i, w_j)$ and $\mathbf{G}(w_i, w_j)$ are determined by the trajectory generation in section III.

B. Deep Q-Network Structure

Incorporated within deep neural network, deep reinforcement learning (DRL) provides a breakthrough in solving problems that are data-driven and have large spaces of possibilities. The proposed waypoints planning framework uses the combination of deep neural network and a model-free RL strategy, namely, Q-learning. Our work aims at planning a set of waypoints that avoid obstacles in a (partially) known environment while minimizing control energy over the entire trip. A deep Q-network (DQN) is adopted for waypoints planning, which uses an offline-built deep neural network (DNN) to derive the correlation between state and action. The DQN approximates the optimal selection of actions in time based on the instantaneous configuration of the environment. With awareness of this configuration and the feedback reward at every time step, the UAV modifies the selection of the next waypoint. As learning proceeds, the UAV discovers its own dynamics and learns how to cope with the external environment by deriving the correlation between each state-action pair $(state, action)$ and its *value function* $Q(state, action)$. At each decision epoch t_i of an execution sequence, the agent performs inference using DNN to estimate the $Q(state_i, action_i)$ and then selects the action either by random choice or by choosing the one that has the highest estimated $Q(state_i, action_i)$. At decision epoch t_{i+1}, the agent stores the newly estimated Q-value calculated based on the total reward $R(state_i, action_i)$ observed in decision epoch t_i. After learning, the model learns the optimal strategy which the agent uses to map the current state to the action that promises highest feedback reward. A deep convolutional network is applied as shown in Figure 2 and the sizes of each layer are also indicated. Details of the waypoints planning process can be found in [20].

III. Trajectory Generation through Waypoints

A. Continuous Time Dynamics

The rigid body model considered in this paper has four control inputs for the six degrees of freedom. These are the three control inputs that generate a torque for the three degrees of freedom of rotational motion, and one thrust along

a body-fixed thrust vector; this model is identical to that used in [19]. This model can be applied to several unmanned vehicles, and the particular case of a quadrotor UAV is considered in section V for numerical results.

In this work, $b \in \mathbb{R}^3$ is the rigid body's position vector expressed in an inertial coordinate frame and $\mathcal{R} \in \mathrm{SO}(3)$ is the rigid body's attitude (orientation) expressed as the rotation matrix from body-fixed frame to inertial frame. Without loss of generality, it is assumed that the thrust vector is along the third body-fixed coordinate frame axis. The translational dynamics equation of motion is:

$$m\dot{v} = mge_3 - fr_3, \tag{4}$$

where $v \in \mathbb{R}^3$ is the translational velocity in inertial frame, g is the acceleration due to gravity, $e_3 = [0,\ 0,\ 1]^T$, $fr_3 \in \mathbb{R}^3$ is the control force vector of magnitude f acting on the body, and r_3 is the unit vector along the third axis of the body-fixed coordinate frame, expressed in the inertial frame. Note that r_3 is also the third column of the rotation matrix $\mathcal{R}$. Equation (4) can be rewritten as:

$$\dot{v} = ge_3 - \frac{1}{m} fr_3. \tag{5}$$

The kinematics for the translational motion expressed in inertial coordinate frame is:

$$\dot{b} = v, \qquad \dot{v} = a, \qquad \dot{a} = z, \tag{6}$$

where a is acceleration and z is the derivative of acceleration (jerk). Equations (5) and (6) can be considered as the system equations. By defining input $u = fr_3$, state space representation of the system can be obtained as follows:

$$\dot{x} = Ax + Bu + \begin{bmatrix} 0_{3\times1} & ge_3 & 0_{3\times1} & 0_{3\times1} \end{bmatrix}^{\mathrm{T}}, \quad y = Cx, \tag{7}$$

where

$$x = \begin{bmatrix} b_{3\times1} \\ v_{3\times1} \\ a_{3\times1} \\ z_{3\times1} \end{bmatrix} \in \mathbb{R}^{12}, \quad u \in \mathbb{R}^3, \quad B = \begin{bmatrix} 0_3 \\ 0_3 \\ 0_3 \\ I_3 \end{bmatrix} \in \mathbb{R}^{12\times3},$$

$$A = \begin{bmatrix} 0_3 & I_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & -\frac{1}{m}I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 \\ 0_3 & 0_3 & 0_3 & 0_3 \end{bmatrix} \in \mathbb{R}^{12\times12},$$

$$C = \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} & 0_{3\times3} \end{bmatrix} \in \mathbb{R}^{3\times12},$$

where I_3 is the 3×3 identity matrix.

Remark 1. The above treatment leads to minimization of *snap*, 4th derivative of position, which results in a smoother path compared to methods minimizing jerk or acceleration.

A trajectory is then to be generated for this system to pass through a given set of k waypoints in position, where $k \geq 1$. The set of waypoints consisting of positions in $\mathbb{R}^3$ with respect to an inertial frame, are generated by the method described in previous section. Discretization facilitates numerical simulation of the system and possible onboard implementation. Therefore, in the next section, the dynamics given by the state space equation (7) is discretized.

B. Discretization of Dynamics

Consider a fixed step size in time, h, and a fixed time interval $[0, T]$ over which the trajectory is to be generated in discrete time. Without loss of generality, the initial time is assumed to be 0. Time is discretized as $t_n = nh$ with $T = m_k h$, so that m_k is a positive integer that corresponds to the final time at which the generated trajectory reaches the final given destination waypoint. Let the discrete-time state variable be given by $x_n = x(nh)$, where $n \in \mathcal{N}$ and $\mathcal{N} = \{0, 1, \ldots, m_k\}$. Denote the discrete time instants at which the trajectory passes through the given position waypoints by m_i, $i = \{1, \ldots, k\}$, with $\{m_1, \ldots, m_k\} \subset \mathcal{N}$. A discrete-time version of equation (7) is obtained as:

$$\begin{aligned} x_{n+1} &= A_d x_n + B_d u_n + \begin{bmatrix} 0_{3\times1} & ge_3 & 0_{3\times1} & 0_{3\times1} \end{bmatrix}^{\mathrm{T}}, \\ y_n &= C_d x_n, \end{aligned} \tag{8}$$

$$\text{where:} \qquad A_d = e^{Ah}, \quad B_d = \int_0^h e^{A\sigma} B d\sigma, \quad C_d = C.$$

Note that A^4, and consequently higher powers of A, are zero. Therefore, A_d and B_d can be obtained in exact form. This leads to the *exact discretization* (equation (8)) of the continuous time system (equation (7)). The system in equation (8) is treated as an optimal control system in the next section to generate a trajectory through waypoints.

C. Problem Formulation

The problem of trajectory generation amounts to constructing a feasible discrete-time desired trajectory through the set of k waypoints. The waypoints are generated by DRL algorithm proposed in the previous section. Let the set of k waypoints be given by tuples $(y^w_{m_1}, m_1), (y^w_{m_2}, m_2), \cdots\ (y^w_{m_k}, m_k)$, where the time instants corresponding to these waypoints are denoted by the subscript $m_i \in \mathcal{N}$, with $i = 1, \cdots k$. We construct a discrete optimal control problem such that the output y_n passes through the waypoints found in section II at specified time instants, i.e. $y_{m_i} = y^w_{m_i}$, for $i = 1, \cdots k$. Let the initial state be given by $x(0) = x_{init}$. The boundary condition at the end point is determined by the last waypoint, $y^w_{m_k}$. The optimal control problem can be formulated as follows:

Problem 2 (Discrete-time Optimal Trajectory Generation).

$$\begin{aligned} \text{Minimize} \ \ \mathcal{J}^d =& h\sum_{i=0}^{m_k} \frac{1}{2}(x_i^{\mathrm{T}} Q x_i + u_i^{\mathrm{T}} R u_i) \\ &+ \frac{1}{2}\sum_{j=1}^{k} \left(C_d x_{m_j} - y^w_{m_j}\right)^{\mathrm{T}} S \left(C_d x_{m_j} - y^w_{m_j}\right), \end{aligned}$$

subject to

1) satisfying the dynamical model,

$$x_{i+1} = A_d x_i + B_d u_i, \tag{9}$$

2) and the boundary conditions given by,

$$x_0 = x_{\text{initial}}, \quad C_d x_{m_k} = y^w_{m_k}. \tag{10}$$

Here $Q \in \mathbb{R}^{12\times12} \geqslant 0$, $R \in \mathbb{R}^{3\times3} > 0$ and $S \in \mathbb{R}^{3\times3} \geqslant 0$ are square, symmetric matrices.

Here, high values of the position, velocity, acceleration, jerk and snap are penalized. Additionally, at the time instances corresponding to waypoints, the error between actual position and the desired position waypoint is penalized.

D. Optimal Control Algorithm to Generate Trajectory

The problem 2 can be approached from the principles of optimal control. An augmented performance index can be defined as

$$\mathcal{J}_a^d = \mathcal{J}^d + \sum_{i=0}^{m_k-1} \lambda_{i+1}^{\mathrm{T}}(A_d x_i + B_d u_i - x_{i+1}), \qquad (11)$$

where $\lambda_i \in \mathbb{R}^{12}$ is a vector of co-states. The discrete Hamiltonian can be defined as,

$$\mathcal{H}_i^d = \lambda_{i+1}^{\mathrm{T}}\big(A_d x_i + B_d u_i\big) + \frac{1}{2}(x_i)^{\mathrm{T}} Q x_i + \frac{1}{2} u_i^{\mathrm{T}} R u_i.$$

Then the augmented index in terms of Hamiltonian is

$$\mathcal{J}_a^d = \sum_{i=0}^{m_k-1} (h\mathcal{H}_i^d - h\lambda_{i+1}^{\mathrm{T}} x_{i+1}) + \frac{1}{2}\sum_{j=1}^{k} \big(C_d x_{m_j} - y_{m_j}^w\big)^{\mathrm{T}} S \big(C_d x_{m_j} - y_{m_j}^w\big). \qquad (12)$$

By setting the first variation of $\mathcal{J}_a^d$ to zero, the necessary conditions for optimality are obtained as following:

$$\frac{\partial \mathcal{H}_i^d}{\partial u_i} = 0 \ \text{ for all } i,$$
$$\frac{\partial \mathcal{H}_i^d}{\partial x_i} - \lambda_i = 0, \ \frac{\partial \mathcal{H}_i^d}{\partial \lambda_{i+1}} - x_{i+1} = 0, \ \text{ for } i \neq m_j,$$
$$\frac{\partial \mathcal{H}_{m_j}^d}{\partial x_{m_j}} + \frac{1}{h} C_d^{\mathrm{T}} S (C_d x_{m_j} - y_{m_j}^w) = \lambda_{m_j}, \text{for } j = 1, ..., k-1$$
$$\frac{1}{h} C_d^{\mathrm{T}} S (C_d x_{m_k} - y_{m_k}^w) = \lambda_{m_k} \text{ for } i = m_k.$$

The aforementioned necessary conditions for the discrete time optimal control problem 2 lead to:

$$u_i = -R^{-1} B_d^{\mathrm{T}} \lambda_{i+1}, \quad x_{i+1} = A_d x_i + B_d u_i. \qquad (13)$$

Let the optimal control be of the form $u_i = K_i x_i + \xi_i$ where ξ_i is the part of the control meant to enforce the waypoint constraints, then the co-state can be expressed as $\lambda_i = P_i x_i + \eta_i$. Thus, the optimal control input can be rewritten as,

$$u_i = -[R + (B_d)^{\mathrm{T}} P_{i+1} B_d]^{-1} (B_d)^{\mathrm{T}} (P_{i+1} A_d x_i + \eta_{i+1}).$$

By substituting the expressions for λ_i and u_i from above into the co-states equation, one obtains:

$$P_i x_i + \eta_i = A_d^{\mathrm{T}}\big(P_{i+1}(A_d x_i - M_i(P_{i+1} A_d x_i + \eta_{i+1}))\big) + A_d^{\mathrm{T}} \eta_{i+1} + Q x_i, \qquad (14)$$

$$\text{where} \qquad M_i = B_d [R + (B_d)^{\mathrm{T}} P_{i+1} B_d]^{-1} (B_d)^{\mathrm{T}}. \qquad (15)$$

The above equation is true for arbitrary x_i, so

$$P_i = (A_d)^{\mathrm{T}} P_{i+1} A_d + Q - (A_d)^{\mathrm{T}} P_{i+1} M_i P_{i+1} A_d, \qquad (16)$$

which is true for $i \neq m_j$ and has to be solved backwards in time. The rest of equation (14) gives the expression for η_i:

$$\eta_i = (A_d)^{\mathrm{T}} \{I - P_{i+1} M_i\} \eta_{i+1}, \qquad (17)$$

which is true for $i \neq m_j$ and also has to be solved backwards in time. At intermediate waypoints ($i = m_j$ and $i \neq m_k$), the co-states are given by:

$$P_{m_j} = (A_d)^{\mathrm{T}} P_{m_j+1} A_d + Q + \frac{1}{h} C_d^{\mathrm{T}} S C_d - (A_d)^{\mathrm{T}} P_{m_j+1} M_{m_j} P_{m_j+1} A_d, \qquad (18)$$
$$\eta_{m_j} = (A_d)^{\mathrm{T}} \{I - P_{m_j+1} M_{m_j}\} \eta_{m_j+1} - \frac{1}{h} (C^d)^{\mathrm{T}} S \, y_{m_j}^w.$$

Finally, for the last waypoint (final destination) we have:

$$P_{m_k} = \frac{1}{h} C_d^{\mathrm{T}} S C_d, \ \ \eta_{m_k} = -\frac{1}{h} C_d^{\mathrm{T}} S y_{m_k}^w. \qquad (19)$$

Using (19) the final value of the co-state vector is written as

$$\lambda_{m_k} = \frac{1}{h} C_d^{\mathrm{T}} S (C_d x_{m_k} - y_{m_k}^w). \qquad (20)$$

Remark 2. Let $K_i = [R + (B_d)^{\mathrm{T}} P_{i+1} B_d]^{-1} (B_d)^{\mathrm{T}}$, then the optimal control can be written as $u_i = -K_i\big((P_{i+1} A_d x_i + \eta_{i+1})\big)$. After applying the optimal control, the dynamics of the discrete system given in (8) becomes,

$$x_{i+1} = \big(A_d - B_d K_i P_{i+1} A_d\big) x_i - B_d K_i \eta_{i+1}. \qquad (21)$$

Remark 3. The thrust force is calculated according to:

$$f_i = m \| a_i - g e_3 \|. \qquad (22)$$

The development presented above gives all the necessary equations for trajectory generation through waypoints. The generated trajectory is considered as a desired trajectory, to be tracked as given in the next section.

IV. Tracking of the Generated Trajectory

Tracking is used to complete the integrated waypoint planning and trajectory generation framework. Also, by considering the thrust associated with tracking of the proposed trajectory and a cubic spline trajectory, we show that the proposed trajectory results in a better thrust profile for a UAV. In this section, the dynamics of the multi-rotor UAV and its control scheme are briefly described to show how the control system tracks the generated trajectory.

The total dynamics involved here consists of the translational dynamics from (5) and (6), and the rotational dynamics of the UAV. Both of them are described in details in [21]. For the attitude control, the attitude $\mathcal{R}$ of the UAV is controlled with finite-time stability, which is more robust to disturbance than asymptotic stability or exponential stability.

In this work, the asymptotically stable translational motion control in [21] is applied. Desired velocity in inertial frame is denoted v_d. Then translational errors are constructed:

$$\dot{\tilde{b}} = \tilde{v} = v - v_d, \ m\dot{\tilde{v}} = mge_3 - f\mathcal{R}e_3 - m\dot{v}_d \qquad (23)$$

To design the translational motion control system with asymptotic stability, a Lyapunov candidate is chosen as

$$V_{tr}(\tilde{b},\tilde{v}) = \frac{1}{2}m\tilde{v}^T\tilde{v} + \frac{1}{2}\tilde{b}^T\mathcal{P}\tilde{b} \tag{24}$$

Design the translational control law as follows:

$$f = e_3^T\mathcal{R}^T(mge_3 + \mathcal{P}\tilde{b} + L_v\tilde{v} - m\dot{v_d}) \tag{25}$$

Here, $\mathcal{P}, L_v \in \mathbb{R}^{3\times3}$ are positive definite control gain matrices. Asymptotic stability of the translational control can be determined by taking time derivative of the Lyapunov candidate (24), substituting (23) and (25):

$$\begin{aligned}\dot{V}_{tr} &= v^T m\dot{\tilde{v}} + \dot{\tilde{b}}^T\mathcal{P}\tilde{b} = \tilde{v}^T(m\dot{\tilde{v}} + \mathcal{P}\tilde{b}) \\ &= \tilde{v}^T(mge_3 - f\mathcal{R}e_3 - m\dot{v}_d + \mathcal{P}\tilde{b}) \\ &= -\tilde{v}^T L_v\tilde{v} < 0\end{aligned} \tag{26}$$

As we have finite-time stable attitude tracking control, this ensures $r_3 = \mathcal{R}e_3$ converges to the desired thrust direction in finite time. Therefore, the feedback control law (25) for translational motion control of the multi-rotor UAV is asymptotically stable. With the control scheme described in this section, the method to track the generated trajectory is simulated in the following section.

V. Numerical Simulation

To investigate the robustness and effectiveness of the proposed algorithm and tracking control scheme from section II to IV, three results are shown in this section. (a) A sequence of waypoints is generated through DRL-based waypoints generation algorithm described in section II in a simulated indoor environment. (b) With the waypoints generated in (a), two trajectories are generated with two different algorithms. (c) Two numerical experiments for trajectory tracking are shown here. Using the UAV tracking control scheme alongside its dynamics model described in the last section, both the trajectories generated in (b) are tracked. The computational results in (a), (b) and (c) are all shown for comparison and evaluation.

A. Simulated environment and DRL-based waypoints

The indoor environment considered is bounded and includes obstacles. The start waypoint, is located in front of the entrance door. The third wapoint is just above the center of the smaller desk. Last waypoint, or destination, is at the center of the larger desk's surface. With the DRL-based algorithm proposed in section II, the generated waypoints are shown in Fig. 3, in which the pillar and desks are considered as obstacles and hence, successfully avoided.

B. Trajectory generation

With the waypoints generated by the DRL-based algorithm, two trajectories are generated such that they go through the waypoints. One is the proposed trajectory in section III and the other is a cubic spline trajectory, generated by 'spline' function in MATLAB. Both trajectories are shown in Fig. 3. and each will be considered as a desired trajectory to be tracked in the next part.

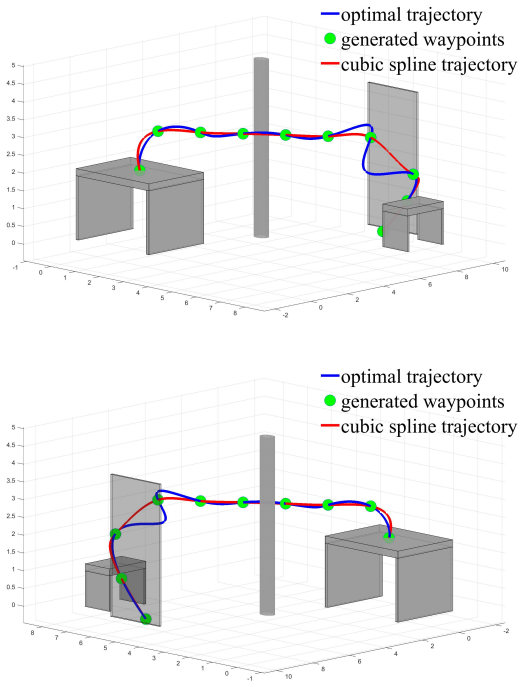

Fig. 3: Generated waypoints, optimal trajectory and cubic spline trajectory in different views

C. Discrete dynamics and control of UAV

To carry out the numerical simulation, the dynamics and control of the UAV is discretized in time. This discrete model is obtained as a Lie group variational integrator (LGVI) using discrete Lagrange d'Alembert principle [22]. Note that the time step is $h = t_{k+1} - t_k$, ν is the translational velocity of the UAV in its body frame, so $v = \mathcal{R}\nu$. The discrete equations of motions are listed as follows:

$$b_{k+1} = h\mathcal{R}_k\nu_k + b_k, \tag{27}$$

$$m\nu_{k+1} = mF_k^T\nu_k + hmg\mathcal{R}_{k+1}^T e_3 - hf_k e_3, \tag{28}$$

where $F_k \approx \exp(h\Omega_k^\times) \in \mathrm{SO}(3)$ guarantees that $\mathcal{R}_k$ evolves on $\mathrm{SO}(3)$. The Ω_k here denotes angular velocity of the UAV. Details for cross map:$(\cdot)^\times : \mathbb{R}^3 \to \mathrm{SO}(3)$ are in [22].

D. Results of the trajectory tracking control

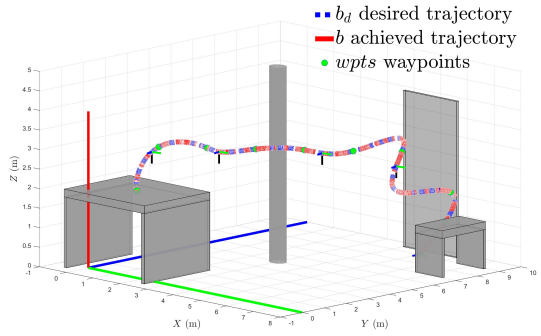

Fig. 4: Comparison between proposed optimal trajectory and trajectory tracked by UAV.

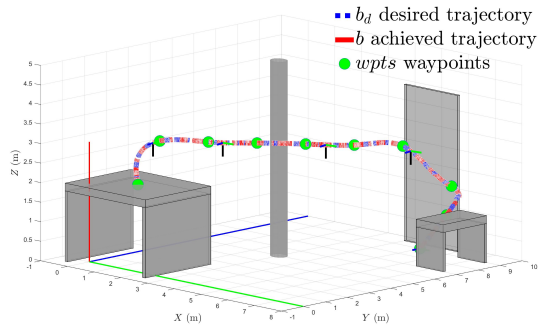

Fig. 5: Comparison between cubic spline trajectory and trajectory tracked by UAV.

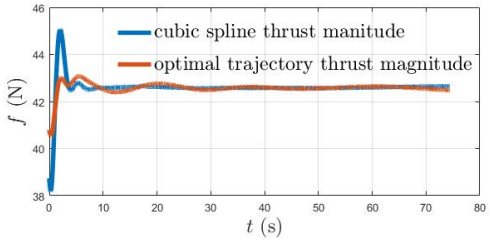

Fig. 6: Comparison of thrust magnitude profile between cubic spline and proposed optimal trajectory.

Figure 4 shows tracking of the proposed trajectory and Fig. 5 shows tracking of the cubic spline trajectory. It is observed the tracking control scheme can track both trajectories well. However, for the thrust magnitude profile, the proposed optimal trajectory shows an advantage in Fig. 6. Its maximum thrust magnitude is smaller than the maximum thrust required to track the cubic spline. Therefore, tracking of the proposed trajectory has a better transient behaviour, and overall requires less control effort than tracking of the cubic spline trajectory.

VI. Conclusion and Future Work

A framework for optimal waypoint planning, trajectory generation, and trajectory tracking in discrete time, is proposed. The framework utilizes a two-level optimization strategy for trajectory generation that can be used to construct optimal, smooth and obstacle-free trajectories through cluttered environments. Numerical simulation results illustrate how the integrated waypoint planning, trajectory generation and tracking scheme work together to generate and track a smooth trajectory in a modeled environment. In the future, experimental verification of the framework on a quadrotor UAV will be carried out in an indoor laboratory environment.

References

[1] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 34, pp. 467–497, 2015.

[2] M. Hehn and R. DAndrea, "Real-time trajectory generation for quadrocopters," *Robotics, IEEE Transactions on*, vol. 31, no. 4, pp. 877–892, 2015.

[3] I. Aizenberg, N. N. Aizenberg, and J. P. Vandewalle, *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*. Springer Science & Business Media, 2013.

[4] M. Collotta, G. Pau, and R. Caponetto, "A real-time system based on a neural network model to control hexacopter trajectories," in *Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), 2014 International Symposium on*. IEEE, 2014.

[5] Q. Li, J. Qian, Z. Zhu, X. Bao, M. K. Helwa, and A. P. Schoellig, "Deep neural networks for improved, impromptu trajectory tracking of quadrotors," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017.

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[7] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[8] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[9] P. Casau, R. G. Sanfelice, R. Cunha, D. Cabecinhas, and C. Silvestre, "Global trajectory tracking for a class of underactuated vehicles," in *American Control Conference (ACC), 2013*. IEEE, 2013, pp. 419–424.

[10] P. Foehn and D. Scaramuzza, "Onboard state dependent lqr for agile quadrotors," in *Robotics and Automation (ICRA), 2018 IEEE International Conference on*, 2018.

[11] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, 2016, pp. 649–666.

[12] F. Stoican and D. Popescu, *Trajectory Generation with Way-Point Constraints for UAV Systems*. Springer International Publishing, 2016.

[13] F. Gao and S. Shen, "Online quadrotor trajectory generation and autonomous navigation on point clouds," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Oct 2016, pp. 139–146.

[14] R. Cimurs, J. Hwang, and I. H. Suh, "Bezier curve-based smoothing for path planner with curvature constraint," in *Robotic Computing (IRC), IEEE International Conference on*. IEEE, 2017, pp. 241–248.

[15] M. P. Vitus, W. Zhang, and C. J. Tomlin, "A hierarchical method for stochastic motion planning in uncertain environments," *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2263–2268, 2012.

[16] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, Dec 2015.

[17] B. Dey and P. S. Krishnaprasad, "Trajectory smoothing as a linear optimal control problem," in *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct 2012, pp. 1490–1497.

[18] M. H. Dhullipalla, R. Hamrah, and A. K. Sanyal, "Trajectory generation on se(3) with applications to a class of underactuated vehicles," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec 2017, pp. 2557–2562.

[19] S. P. Viswanathan, A. K. Sanyal, and M. Izadi, "Integrated guidance and nonlinear feedback control of underactuated unmanned aerial vehicles in SE(3)," in *AIAA Guidance, Navigation, and Control Conference*, Gaylord, TX, January 2017.

[20] Y.Li, H.Eslamiat, N.Wang, Z.Zhao, A. K. Sanyal, and Q.Qiu, "Autonomous waypoints planning and trajectory generation for multi-rotor UAVs," in *Proceedings of Design Automation for CPS and IoT*, 2019.

[21] S. P. Viswanathan, A. K. Sanyal, and E. Samiei, "Integrated guidance and feedback control of underactuated robotics system in se (3)," *Journal of Intelligent & Robotic Systems*, vol. 89, no. 1-2, pp. 251–263, 2018.

[22] A. Sanyal, N. Nordkvist, and M. Chyba, "An almost global tracking control scheme for maneuverable autonomous vehicles and its discretization," *Automatic Control, IEEE Transactions on*, vol. 56, no. 2, pp. 457–462, 2011.