CTFF: An Efficient Strategy for Continuously Targeting Fire Fighting with Quadrotor

Weihao Chen, Ziying Lin, Peng Peng, and Wei Dong

***Abstract*— This paper considers the optimal control problem of continuously targeting for extinguishing a moving fire source with a quadrotor. It aims to enhance computational time-efficiency, and real-time targeting performance compared with generic optimizers onboard, even suffering from severe nonlinearities. To this end, an efficient strategy is proposed for this optimal control problem. First, the Gauss Pseudospectral Method is utilized to transform the optimal control problem into a nonlinear programming problem. Then, via the Sequential Quadratic Programming (SQP) algorithm, the control input can be generated analytically. To further improve the computational efficiency, the techniques, separation of variables and prior gradient calculation, are applied. Comparative simulations are carried out to verify the excellent targeting performance and the high computational efficiency. Results demonstrate that the proposed efficient strategy can compute the control action in approximately 0.5 ms, which is 20 times faster than the generic optimizers.**

I. Introduction

In the past few years, the research on robotic firefighting has made great progress, and the high agility of firefighting drones is particularly eye-catching [1]. The quadrotor has demonstrated impressive capabilities in fire detection [2] and suppression [3] research. Motivated by those works, the quadrotor equipped with devices such as fire extinguisher and dry ice bomb might be able to further apply for eliminating outdoor moving fire sources. [4] Due to the flexibility of drones, they can achieve efficient fire extinguishing, significantly reducing the danger or labor of human beings when coping with emergency fires.

However, the research on fire-fighting drones is limited to static scenes. [5] Continuously targeting and extinguishing moving fire sources have not been explored. In reality, there are many scenarios of dynamic fire sources, such as some continuously rotating coal conveyor belts if the coal blocks catch fire. In addition, the nonlinearity of the continuously fixed-point targeting problem of the quadrotor in the continuous time scale has high requirements for computational efficiency, and a more efficient solution method is needed to ensure targeting accuracy.

Different from the case of the static scene as mentioned in [5], the moving fire source should be continuously targeted during the extinguishing process. Such a targeting task requires accurate control simultaneously on the position and pose of the underactuated quadrotor on a continuous-time horizon. Given that the targeted fire source moves randomly during the whole targeting process, the waypoints cannot be obtained a priori in this underactuated targeting problem, as the fully actuated multiple fixed waypoints problem does. Meanwhile, the real-time tracking problem in this paper cannot be directly linearized because three variables, i.e. the pitch angle, the translational position, and the vertical position of the quadrotor, are controlled simultaneously with two control inputs, i.e.,, thrust and pitch, to ensure the targeting. At the same time, the vertical targeting performance target is determined by the pitch angle, the translation, and the vertical position simultaneously. The pitch angle and translation position are also coupled. Worse still is that in the process of dynamic firefighting, devices such as fire extinguishers and dry ice bombs will also bring about problems such as recoil and load changes.

Quadrotor's underactuated characteristics and its coupled kinematics with the moving fire source, smooth feedback control [5]–[7] is not applicable for such a targeting task, and an effective optimal targeting model should be established.

In previous research, mainly two similar types of optimal problems are extensively studied, i.e., state-to-state maneuver problem [8]–[11] and optimal tracking multiple fixed waypoint problem [12]–[16]. For the state-to-state maneuver problem, state interpolation is more straightforward since there are no constraints on the states between the start and end positions. For example, in Refs. [8], [9], the trajectory generator is used in a model predictive control like strategy, where thousands of trajectories are generated and evaluated at every controller update step. In Refs. [10], [11], a time-optimal trajectory generation algorithm is proposed for landing a quadrotor onto a translating (heaving) and tilting (pitching) platform. Although it involves the tracking of the target, the difficulty of this problem mainly lies in the landing action, which is different from the real-time targeting error concerned in this paper. On the other hand, the optimal tracking problem of multiple fixed waypoint is fully actuated, so its is enough to solve the trajectory planning problem given an invariant terminal target point or a set of fixed waypoint.

To tackle this problem, we first build an optimal tracking model [17] that takes into account such conditions as load changes in dynamic fire suppression. We use the Gauss Pseudospectral Method to transform The continuous-time optimal control problem into a nonlinear programming problem. The control input can then be generated analytically by solving the QP sub-problem using Sequential Quadratic Program-

Weihao Chen is with Faculty of Mechanical Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, 200240, Shanghai, China Chen.WH@sjtu.edu.cn

Ziying Lin, Peng Peng, and Wei Dong are with the State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiaotong University, 200240, Shanghai, China. E-mails: lynnaero, pengpeng, dr.dongwei@sjtu.edu.cn.

ming (SQP) algorithm. In addition, the calculation efficiency is further improved by separating variables and prior gradient calculation. Comparative simulations are carried out to verify the targeting accuracy and computational efficiency. Results demonstrate that our method can compute the control action in approximately 0.5 ms (compared to the generic optimizers, the efficiency is improved by about 20 times). The constant-speed tracking error is less than 0.1m, and the error range is also acceptable in the case of variable-speed tracking.

The contributions of this paper include:

1) For the problem of continuously targeting for firefighting with the quadrotor, an efficient strategy is first proposed. Use GPM + SQP + variable separation and utilize gradient priors to speed up computation.
2) A simulation environment is established for the firefighting tasks with the quadrotor. Results demonstrate the excellent targeting performance and the high computational efficiency of the proposed efficient strategy.
3) The released code at https://github.com/Chen-WH/ARM2022, including an example application in ROS.

The remaining content is organized as follows: Section II presents the model and problem formulation. Section III presents the optimization method. The results of the simulation experiments are given in section IV. Section V concludes the paper.

II. Modeling and Problem Formulation

We first present the general dynamic model of the quadrotor in this section. On this basis, the corresponding mathematical optimization problem is established, and the solution details are fully excavated to improve the performance.

To clarify the formulation of the optimal control strategy designed in this paper, two assumptions are imposed as follows.

1) The pitch angle θ and the thrust f of the quadrotor can be set directly without dynamics and delay such that they can be regarded as the control inputs.
2) Considering that the yaw angle ψ and the roll angle ϕ of the quadrotor can be controlled separately without underactuated characteristics, it can be approximated as zero, i.e.,$\psi = 0, \phi = 0$.

In view of the developed model, the control objective can be discribed as follows. An optimal control input of the quadrotor is required to design in such a way the quadrotor can continuously and accurately target the fire source and keep in a safe distance. The targeting problem is illustrated in Fig. 1, where $\boldsymbol{p}_{fire}^{\mathcal{I}} = x_{fire}\boldsymbol{e}_x^{\mathcal{I}} + y_{fire}\boldsymbol{e}_y^{\mathcal{I}} + z_{fire}\boldsymbol{e}_z^{\mathcal{I}}$ represents the position of the desired point in inertial coordinate $\boldsymbol{e}^{\mathcal{I}}$.

Combining with Assumption 1, we can define the state vector $\boldsymbol{x}$ and the control input vector $\boldsymbol{u}$ of the system as:

$$\begin{aligned}\boldsymbol{x} &= \left[(\boldsymbol{p}_{drone}^{\mathcal{I}} - \boldsymbol{p}_{fire}^{\mathcal{I}})^T, (\boldsymbol{v}_{drone}^{\mathcal{I}} - \boldsymbol{v}_{fire}^{\mathcal{I}})^T, (\boldsymbol{v}_{drone}^{\mathcal{I}})^T, m\right]^T \\ \boldsymbol{u} &= \left[\frac{f}{m}, \theta\right]^T\end{aligned} \tag{1}$$

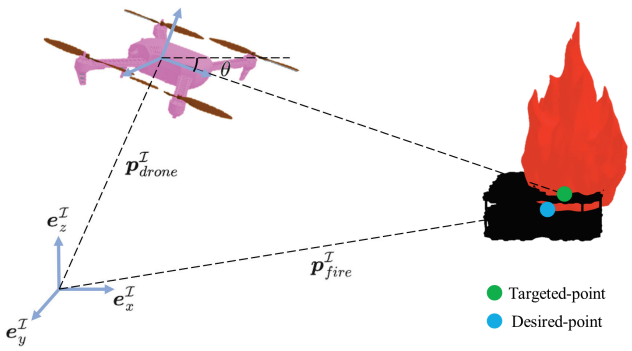


Fig. 1. A schematic diagram for the targeting problem. Blue point is the desired hit point of the fire extinguisher in the turtlrbot3; green point is the intersection of the fire extinguishing device's extension line and the targeted plane.

where $\boldsymbol{v}_{fire}^{\mathcal{I}} = v_{fire,x}\boldsymbol{e}_x^{\mathcal{I}} + v_{fire,y}\boldsymbol{e}_y^{\mathcal{I}} + v_{fire,z}\boldsymbol{e}_z^{\mathcal{I}}$ represents the velocity of a moving fire source in inertial coordinate $\boldsymbol{e}$. The quadrotor's velocity $\boldsymbol{v}_{drone}^{\mathcal{I}}$ is considered as separate states. It is separately introduced to reflect the air drag effect. In this way, the static case and the uniform motion case, where the turtlrbot3 and the quadrotor are both relatively stationary, can be distinguished. m is the mass of the quadrotor and the changing load, because the design of the control law also needs to consider the change of the device load during the fire extinguishing process.

Based on Assumption 2, the dynamics of the targeting process can then be reformulated as follows:

$$\begin{aligned}\dot{\boldsymbol{x}}(t) =& h\left[\boldsymbol{x}(t), \boldsymbol{u}(t)\right] \\ =& \left[(\boldsymbol{v}_{drone}^{\mathcal{I}} - \boldsymbol{v}_{fire}^{\mathcal{I}})^T, (\boldsymbol{a}_{drone}^{\mathcal{I}} - \boldsymbol{a}_{fire}^{\mathcal{I}})^T\right]^T \\ \boldsymbol{a}_{drone}^{\mathcal{I}} =& [\boldsymbol{u}_1 C_{\boldsymbol{u}_3} S_{\boldsymbol{u}_2} - c_1\boldsymbol{x}_7, -\boldsymbol{u}_1 S_{\boldsymbol{u}_3} - c_2\boldsymbol{x}_8, \\ & \boldsymbol{u}_1 C_{\boldsymbol{u}_3} C_{\boldsymbol{u}_2} - c_3\boldsymbol{x}_9 - \frac{m_0}{m}g\Big]^T\end{aligned} \tag{2}$$

where $S_x = \sin(x)$ and $C_x = \cos(x)$, $\boldsymbol{x}_i$ represents the i-th element of $\boldsymbol{x}$, $\boldsymbol{u}_i$ represents the i-th element of $\boldsymbol{u}$.

To facilitate the optimal control, define $d_x = x_{fire} - x_{drone}$ as the distance from the center of mass of the quadrotor to the targeted plane, and define $d_z = d_x \tan\theta - (z_{drone} - z_{fire})$ as the distance along $\boldsymbol{e}_z^{\mathcal{I}}$ between the targeted point and the desired point on the targeted plane. For this targeting task, the control objective is to design an optimal control input $\boldsymbol{u}$ such that d_x is as close to the safe distance(3 meters in this paper) as possible. At the same time, d_y, d_z and control input should be minimized within a finite time horizon($t \in [0, t_f]$). Therefore, the considered optimal control problem of the targeting problem can be formulated as follows:

$$\begin{aligned}
\min \quad & \mathcal{J} = \int_0^{t_f} f[\boldsymbol{x}(t), \boldsymbol{u}(t), t]\mathrm{d}t \\
\text{s.t.} \quad & f[\boldsymbol{x}(t), \boldsymbol{u}(t), t] = \frac{1}{2}\boldsymbol{u}^{\boldsymbol{T}}\boldsymbol{u} + k_1(d_x - 3)^2 + k_2 d_z^2 \\
& \dot{\boldsymbol{x}}(t) = h[\boldsymbol{x}(t), \boldsymbol{u}(t)] \\
& 0 \leq g[\boldsymbol{u}(t)] \\
& \boldsymbol{x}(0) = \boldsymbol{x_0}
\end{aligned} \tag{3}$$

where the terminal time t_f is fixed; $\boldsymbol{x_0} \in \mathbb{R}^9$ is the initial value of the state vector, which is known as a prior condition; $k_i, i = 1, 2$ are the positive weights.

From Eq.3, it can be inferred that the pitch angle $\boldsymbol{u}_2 = \theta$ is coupled with the translational position $\boldsymbol{x}_1$ in the performance objective $\mathcal{J}$. Additionally, there exist severe nonlinear characteristics in this optimal control problem, owing to the coupled kinematics with the turtlrbot3 and the inherent nonlinearities of the quadrotor. Therefore, it is very time-consuming to directly solve this nonlinear optimal control problem with generic optimizers, which cannot meet the real-time computing requirements of the real-time system.

III. Optimal Control

It is noted that the optimal control problem of Eq.3 can be transformed from the time interval $\tau \in [-1, 1]$ to the time interval $t \in [0, t_f]$ via the linear mapping.

$$t = \frac{t_f}{2}\tau + \frac{t_f}{2} \tag{4}$$

A. Gauss Pseudospectral Discretization

The direct approach to solving the continuous Bolza optimal control problem of Sec.II is to discretize and transform Eq.3 to an Nonlinear Programming(NLP) [18]. The Gauss pseudospectral method is based on approximating the state and control trajectories using interpolating polynomials. The state and control can be approximated using the basis of the $N+1$ Lagrange interpolation polynomial $L_i(\tau)$ and the N Lagrange interpolation polynomial $L_i^*(\tau)$, respectively.

$$\begin{aligned}
\boldsymbol{x}(\tau) &\approx \boldsymbol{X}(\tau) = \sum_{i=0}^{N} \boldsymbol{X}(\tau_i) L_i(\tau) \\
L_i(\tau) &= \prod_{j=0, j\neq i}^{N} \frac{\tau - \tau_j}{\tau_i - \tau_j}, i = 0, \ldots, N
\end{aligned} \tag{5}$$

$$\begin{aligned}
\boldsymbol{u}(\tau) &\approx \boldsymbol{U}(\tau) = \sum_{i=1}^{N} \boldsymbol{U}(\tau_i) L_i^*(\tau) \\
L_i^*(\tau) &= \prod_{j=1, j\neq i}^{N} \frac{\tau - \tau_j}{\tau_i - \tau_j}, i = 1, \ldots, N
\end{aligned} \tag{6}$$

Differentiating the expression in Eq.5, the derivative of each Lagrange polynomial at the Legendre-Gauss points can be represented in a differential approximation matrix, $\boldsymbol{D} \in \mathbb{R}^{N \times N+1}$. The elements of the differential approximation matrix are determined offline as follows:

$$\begin{aligned}
\dot{\boldsymbol{x}}(\tau) &\approx \dot{\boldsymbol{X}}(\tau) = \sum_{i=0}^{N} \boldsymbol{X}(\tau_i) \dot{L}_i(\tau) \\
\boldsymbol{D}_{ki} &= \dot{L}_i(\tau_k) = \sum_{l=0}^{N} \frac{\prod_{j=0, j\neq i, l}^{N} (\tau_k - \tau_j)}{\prod_{j=0, j\neq i}^{N} (\tau_i - \tau_j)}
\end{aligned} \tag{7}$$

where $k = 1, \ldots, N$ and $i = 0, \ldots, N$. The dynamic constraint is transformed into algebraic constraints via the differential approximation matrix as follows:

$$\sum_{i=0}^{N} \boldsymbol{D}_{ki} \boldsymbol{X}_i - \frac{t_f}{2} h(\boldsymbol{X}_k, \boldsymbol{U}_k, \tau_k) = \boldsymbol{0}, k = 1, \ldots, N \tag{8}$$

where $\boldsymbol{X}_k \equiv \boldsymbol{X}(\tau_k) \in \mathbb{R}^n$ and $\boldsymbol{U}_k \equiv \boldsymbol{U}(\tau_k) \in \mathbb{R}^m (k = 1, \ldots, N)$. Note that the dynamic constraint is collocated only at the Legendre-Gauss points and not at the boundary points. Additional variables in the discretization are defined as follows: $\boldsymbol{X}_0 \equiv \boldsymbol{X}(-1)$, and $\boldsymbol{X}_f$, where $\boldsymbol{X}_f$ is defined in terms of $\boldsymbol{X}_k, (k = 0, \ldots, N)$ and $\boldsymbol{U}_k, (k = 1, \ldots, N)$ via the Gauss quadrature

$$\boldsymbol{X}_f \equiv \boldsymbol{X}_0 + \frac{t_f}{2} \sum_{k=1}^{N} w_k h(\boldsymbol{X}_k, \boldsymbol{U}_k, \tau_k) \tag{9}$$

where w_k are the Gauss weights. The continuous cost function in Eq.3 is approximated using a Gauss quadrature as

$$\mathcal{J} = \frac{t_f}{2} \sum_{k=1}^{N} w_k f(\boldsymbol{X}_k, \boldsymbol{U}_k, \tau_k; 0, t_f) \tag{10}$$

Next, the control input constraint in Eq.3 is expressed as

$$g(\boldsymbol{X}_k, \boldsymbol{U}_k, \tau_k) \geq 0 \tag{11}$$

The cost function of Eq.11 and the algebraic constraints of Eq.9-11 define an NLP whose solution is an approximate solution to the continuous Bolza problem. Finally, it is noted that discontinuity in the state or control can be handled efficiently by dividing the trajectory into phases, where the dynamics are transformed within each phase and then connected by additional phase interface (also known as linkage) constraints. The complete expression of the model problem in this paper is

$$\begin{aligned}
\min \quad & J = \frac{t_f}{2} w^T \Big[\frac{1}{2}(U_1 - g) \odot (U_1 - g) + \frac{1}{2} U_2 \odot U_2 \\
& + k_1(X_1 + 3) \odot (X_1 + 3) \\
& + k_2(X_1 \odot U_2 + X_2) \odot (X_1 \odot U_2 + X_2) \Big] \\
s.t. \quad & 0 = \widetilde{D}\widetilde{X}_1 - \frac{t_f}{2} X_3 \\
& 0 = \widetilde{D}\widetilde{X}_2 - \frac{t_f}{2} X_4 \\
& 0 = \widetilde{D}\widetilde{X}_3 - \frac{t_f}{2}(U_1 U_2 - c_1 X_5) \\
& 0 = \widetilde{D}\widetilde{X}_4 - \frac{t_f}{2}(U_1 - g - c_2 X_5)
\end{aligned} \tag{12}$$

$$0 \leq U_1 \leq 30, \quad -1 \leq U_2 \leq 1$$

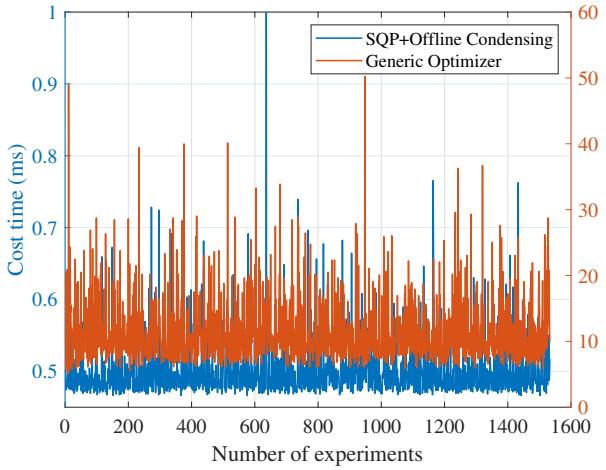


Fig. 2. Schematic diagram of computational efficiency comparison.(The maximum number of iterations is 2000) The SQP+Offline condensing (variable separation and prior gradient) method in this paper is about 20 times faster than the generic optimizer

B. Sequential Quadratic Programming

The above NLP problem Eq.12 can be simplified to the following form.

$$\begin{aligned} \min\ & f(\boldsymbol{Z}) \\ s.t.\ & h_i(\boldsymbol{Z}) = 0, \quad i \in E \\ & g_i(\boldsymbol{Z}) \leq 0, \quad i \in I \end{aligned} \tag{13}$$

where $\boldsymbol{Z} = [\boldsymbol{X}^T, \boldsymbol{U}^T]^T$, E is the set of indicators for equality constraints, I is the set of indicators for inequality constraints. The Lagrangian for this problem is

$$\mathcal{L}(\boldsymbol{Z}, \lambda, \sigma) = f(\boldsymbol{Z}) + \sum_{i \in E} \lambda_i h_i(\boldsymbol{Z}) + \sum_{i \in I} \sigma_i g_i(\boldsymbol{Z}) \tag{14}$$

where λ_i and σ_i are Lagrange multipliers. At the k-th iteration $\boldsymbol{Z}_k$, a basic sequential quadratic programming algorithm defines an appropriate search direction $\boldsymbol{d}_k$ as a solution to the quadratic programming subproblem.

$$\begin{aligned} \min\ & \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) + \nabla \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma)^T \boldsymbol{d}_k \\ & + \frac{1}{2} \boldsymbol{d}_k^T \nabla^2 \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) \boldsymbol{d}_k \\ s.t.\ & h_i(\boldsymbol{Z}_k) + \nabla h_i(\boldsymbol{Z}_k)^T \boldsymbol{d}_k = 0 \\ & g_i(\boldsymbol{Z}_k) + \nabla g_i(\boldsymbol{Z}_k)^T \boldsymbol{d}_k \leq 0 \end{aligned} \tag{15}$$

Inequality constraint $g_i(\boldsymbol{Z})$ is called active at $\boldsymbol{Z}$ if $g_i(\boldsymbol{Z}) = 0$, and inactive at $\boldsymbol{Z}$ if $g_i(\boldsymbol{Z}) < 0$ equality constraints are always active. The active set at $\boldsymbol{Z}$ is made up of those constraints $g_i(\boldsymbol{Z})$ that are active at the current point. In each iteration, the inequality constraints that satisfy the condition $g_i(\boldsymbol{Z}_k) > -\varepsilon$($\varepsilon$ is a small constant) are recorded as the active set g_i^*, solve the QP subproblem defined by the active set.

$$\begin{aligned} \min\ & \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) + \nabla \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma)^T \boldsymbol{d}_k \\ & + \frac{1}{2} \boldsymbol{d}_k^T \nabla^2 \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) \boldsymbol{d}_k \\ s.t.\ & h_i(\boldsymbol{Z}_k) + \nabla h_i(\boldsymbol{Z}_k)^T \boldsymbol{d}_k = 0 \\ & g_i^*(\boldsymbol{Z}_k) + \nabla g_i^*(\boldsymbol{Z}_k)^T \boldsymbol{d}_k = 0 \end{aligned} \tag{16}$$

$$\begin{cases} \nabla \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) + \nabla^2 \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma)^T \boldsymbol{d}_k = 0 \\ h_i(\boldsymbol{Z}_k) + \nabla h_i(\boldsymbol{Z}_k)^T \boldsymbol{d}_k = 0 \\ g_i^*(\boldsymbol{Z}_k) + \nabla g_i^*(\boldsymbol{Z}_k)^T \boldsymbol{d}_k = 0 \end{cases} \tag{17}$$

where $\nabla \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) = \nabla f(\boldsymbol{Z}_k) + \sum_{i \in E} \lambda_i \nabla h_i(\boldsymbol{Z}_k) + \sum_{i \in I} \sigma_i \nabla g_i^*(\boldsymbol{Z}_k)$, the matrix form is obtained as follows, so that it can be solved to obtain $\boldsymbol{d}_k$.

$$\begin{bmatrix} \nabla^2 \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma) & \nabla \boldsymbol{h}(\boldsymbol{Z}_k) & \nabla \boldsymbol{g}^*(\boldsymbol{Z}_k) \\ \nabla \boldsymbol{h}(\boldsymbol{Z}_k)^T & \boldsymbol{O} & \boldsymbol{O} \\ \nabla \boldsymbol{g}^*(\boldsymbol{Z}_k)^T & \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{d}_k \\ \boldsymbol{\lambda} \\ \boldsymbol{\sigma} \end{bmatrix} = \begin{bmatrix} -\nabla f(\boldsymbol{Z}_k) \\ -\boldsymbol{h}(\boldsymbol{Z}_k) \\ -\boldsymbol{g}^*(\boldsymbol{Z}_k) \end{bmatrix} \tag{18}$$

C. Offline Condensing

The online QP solution can be prepared by a condensing of the QP Eq.18: We divide $\boldsymbol{d}_k = \Delta \boldsymbol{Z}$ into its state and control components $\Delta \boldsymbol{X}$ and $\Delta \boldsymbol{U}$, and resolve the equality constraints to obtain $\Delta \boldsymbol{X}$ as a linear function of $\Delta \boldsymbol{U}$, such that we can substitute

$$\Delta \boldsymbol{X} = \boldsymbol{M} \Delta \boldsymbol{U} + \boldsymbol{N} \tag{19}$$

Note that the expression for matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ can be precomputed offline, we use the expression to substitute $\Delta \boldsymbol{X}$ wherever it appears in the QP, to generate the condensed QP.

The expression for all matrices in Eq.17, $\nabla^2 \mathcal{L}(\boldsymbol{Z}_k, \lambda, \sigma), \nabla \boldsymbol{h}(\boldsymbol{Z}_k), \nabla \boldsymbol{g}^*(\boldsymbol{Z}_k), \nabla f(\boldsymbol{Z}_k), \boldsymbol{h}(\boldsymbol{Z}_k), \boldsymbol{g}^*(\boldsymbol{Z}_k)$ of this QP sub-problem are precomputed offline as prior expressions.

Since the solution efficiency of SQP is affected by the selection of the initial point, the boundary conditions of the two adjacent control problems are similar under the control frequency of 50 Hz. Therefore, we use the previous calculation result to initialize (warm-starting) each time to further improve efficiency. [19]

IV. Simulation Results

A. Evaluation Environment Setup

The simulation is implemented on a desktop computer with an Intel Core i7-9750H CPU running at 2.60GHz, with 16GB of RAM. It runs on a Linux (Ubuntu 18.04) operating system and is constructed in a physical reality environment.

The implemented structure of this simulation environment is illustrated in Fig. 3. The quadrotor and turtlrbot3 dynamics are implemented in the Gazebo environment. The control node is implemented in C++ to calculate the optimal control input and then control the quadrotor at a frequency of 50 Hz. The control node is implemented in the Robot Operating System (ROS).

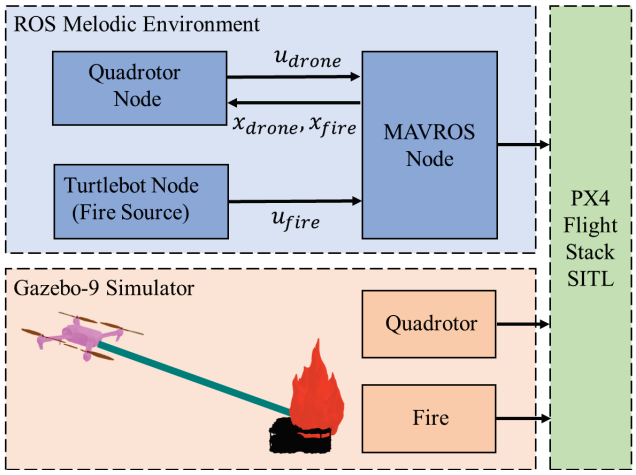


Fig. 3. The schematic diagram of system structure in simulation

In the following simulations and experiments, to further increase the computational efficiency, the position of the quadrotor along $\boldsymbol{e}_y^{\mathcal{I}}$ is separately controlled by PD controller as the proposed model does.

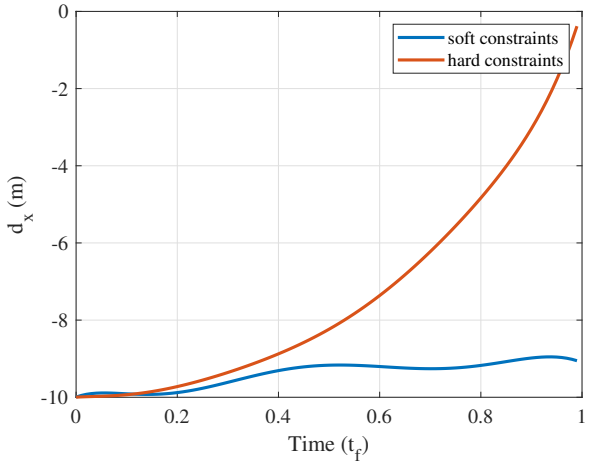


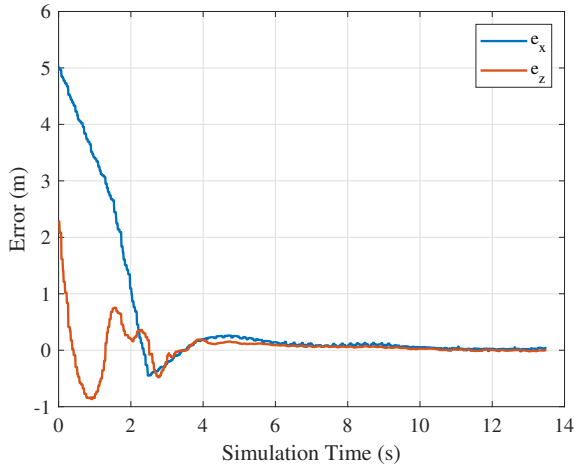
Fig. 4. Schematic diagram of the trajectory using GPM formulation. Different tracking strategies can be simulated by adjusting the weights of constraints to generate different tracking trajectories.(The fire source is at the $[-10, 0, 1]^T$ position)

B. Simulation Tests

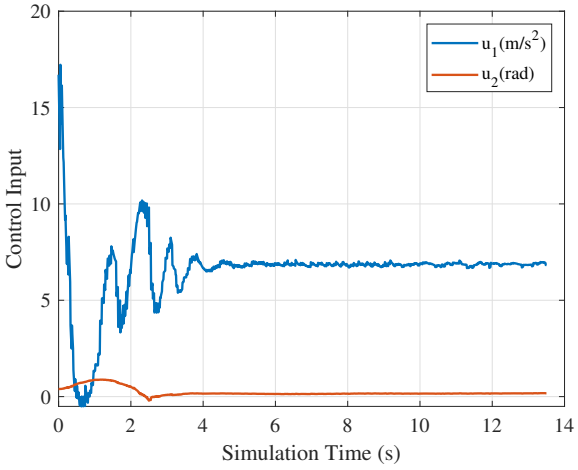
In this section, two cases are carried out on the desktop computer to investigate the targeting performance and computational efficiency of the designed method.

Case 1: The first test is conducted, where the turtlebot3 (fire source) is moving along $\boldsymbol{e}_x^{\mathcal{I}}$ at a constant speed, i.e., $v_{fire,x} = 3\mathbf{m/s}$, and the height of fire location is 0.15m. At the initial moment, the Turtlrbot3 Waffle Pi (fire source) is located at the origin, and the quadrotor is hovering at $[-10, 0, 0.15]^T$. The comparative targeting results are depicted in Fig. 5, where d_x, d_z represents the targeting error, u_1, u_2 represents the control inputs defined in Sec. II.

The average computation time in each controller updating timestep is 0.53 ms, which is better than the time (14.63ms) required to directly solve the original boundary value problem with $scipy.integrate.solve\_bvp$[1] in Python 3.6. Results demonstrate that the constant-speed tracking error is less than 0.1m, while the state curve and control inputs are smooth and stable.



(a) Targeting error

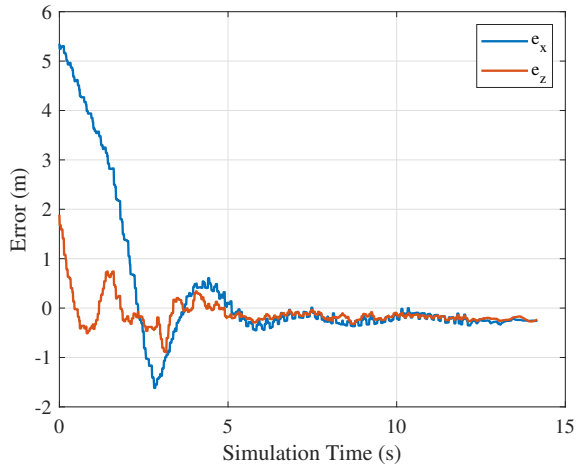


(b) Control inputs

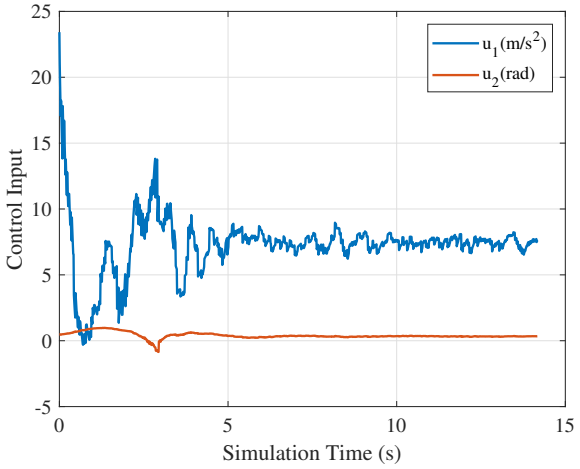
Fig. 5. Case 1 in simulation: the turtlebot3 (fire source) moves by constant speed

Case 2: The second test is then carried out when the velocity of the turtlebot3 (fire source) is time-varying (uniform acceleration motion). Similar to case 1, the quadrotor is hovering at $[-10, 0, 0.15]^T$ and the turtlebot3 is located at the origin initially. The targeting result is illustrated in Fig. 6. Thanks to the high computational efficiency of the proposed optimization method, it is reasonable to treat the turtlebot3's velocity as a constant at each controller updating timestep in the system modeling. Therefore, the method proposed in this paper still has good performance under the time-varying speed of turtlebot3.

[1]see https://github.com/scipy/scipy



(a) Targeting error



(b) Control inputs

Fig. 6. Case 2 in simulation: the turtlebot3 (fire source) moves by time-varying speed

V. Conclusion

This paper proposes a strategy that enables continuously targeting and extinguishing of moving fire sources. First, we use the Gauss Pseudospectral Method to transform The continuous-time optimal control problem into a nonlinear programming problem. The control input can then be generated analytically by solving the QP sub-problem using Sequential Quadratic Programming (SQP) algorithm. In addition, the calculation efficiency is further improved by separating variables and prior gradient calculation. Comparative simulations are carried out to verify the targeting accuracy and computational efficiency. Results demonstrate that our method can compute the control action in approximately 0.5 ms (compared to the generic optimizers, the efficiency is improved by about 20 times). The constant-speed tracking error is less than 0.1m, and the error range is also acceptable in the case of variable-speed tracking.

References

[1] V. Spurny, V. Pritzl, V. Walter, M. Petrlik, T. Baca, P. Stepan, D. Zaitlik, and M. Saska, "Autonomous firefighting inside buildings by an unmanned aerial vehicle," *IEEE Access*, vol. 9, pp. 15 872–15 890, 2021.

[2] C. Yuan, Z. Liu, and Y. Zhang, "Uav-based forest fire detection and tracking using image processing techniques," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2015, pp. 639–643.

[3] E. Ausonio, P. Bagnerini, and M. Ghio, "Drone swarms in fire suppression activities: A conceptual framework," *Drones*, vol. 5, no. 1, p. 17, 2021.

[4] B. Aydin, E. Selvi, J. Tao, and M. J. Starek, "Use of fire-extinguishing balls for a conceptual system of drone-assisted wildfire fighting," *Drones*, vol. 3, no. 1, p. 17, 2019.

[5] A. Farinha, R. Zufferey, P. Zheng, S. F. Armanini, and M. Kovac, "Unmanned aerial sensor placement for cluttered environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6623–6630, 2020.

[6] V. P. Tran, F. Santoso, M. A. Garratt, and I. R. Petersen, "Adaptive second-order strictly negative imaginary controllers based on the interval type-2 fuzzy self-tuning systems for a hovering quadrotor with uncertainties," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 1, pp. 11–20, 2019.

[7] G. Chen, W. Dong, X. Sheng, X. Zhu, and H. Ding, "An active sense and avoid system for flying robots in dynamic environments," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 2, pp. 668–678, 2021.

[8] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3480–3486.

[9] W. Dong, G.-Y. Gu, Y. Ding, X. Zhu, and H. Ding, "Ball juggling with an under-actuated flying robot," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 68–73.

[10] B. Hu and S. Mishra, "A time-optimal trajectory generation algorithm for quadrotor landing onto a moving platform," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 4183–4188.

[11] Hu, Botao and Mishra, Sandipan, "Time-optimal trajectory generation for landing a quadrotor onto a moving platform," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 2, pp. 585–596, 2019.

[12] R. Charles, A. Bry, and N. Roy, "Polynomial trajectory planning for quadrotor flight," in *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*, 2013.

[13] G. Hoffmann, S. Waslander, and C. Tomlin, "Quadrotor helicopter trajectory tracking control," in *AIAA guidance, navigation and control conference and exhibit*, 2008, p. 7410.

[14] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "A prototype of an autonomous controller for a quadrotor uav," in *2007 European Control Conference (ECC)*. IEEE, 2007, pp. 4001–4008.

[15] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter," in *2008 16th mediterranean conference on control and automation*. Ieee, 2008, pp. 1258–1263.

[16] M. Greeff and A. P. Schoellig, "Flatness-based model predictive control for quadrotor trajectory tracking," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 6740–6745.

[17] Z. Lin, W. Dong, S. Liu, X. Sheng, and X. Zhu, "An efficient egocentric regulator for continuous targeting problems of the underactuated quadrotor," *arXiv preprint arXiv:2108.02930*, 2021.

[18] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao, "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.

[19] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *IEEE Transactions on control systems technology*, vol. 18, no. 2, pp. 267–278, 2009.