MEDIAL AXIS TRANSFORMATION-BASED CONVEX MODEL PREDICTIVE CONTROL FOR MULTIROTOR INDOOR AUTONOMOUS FLIGHT WITH FIXED ALTITUDE

Hyungjoo Ahn[1] & Hyochoong Bang[1]

[1]KAIST, Korea Advanced Institute of Science and Technology

Abstract

This paper presents an advanced strategy for Medial Axis Transformation (MAT)-based Convex Model Predictive Control (CMPC) for multirotor indoor autonomous flight. The main idea of this proposed framework is the collaboration of Moptimizinggraph(CFG) generation with a set of circles, piecewisely-and-sequencially defined safe flight corridor(SFC), and CMPC encompassing constraints arising from surrounding obstacles, dynamic equations, and system saturation for the local planner. The proposed approach is capable of generating near-optimal trajectories that closely approximate the optimal solutions, thereby ensuring efficient path planning. Moreover, the computational efficiency of the proposed method is demonstrated through promising results in various aspects of trajectory planning tasks.

Keywords: Convex Model Predictive Control(CMPC), Safe Flight Corridor(SFC), Medial Axis Transformation(MAT), Global & Local Planner

1. Introduction

In the advancements seen in autonomous UAV flight indoors, obstacle avoidance methods are frequently used because indoor spaces tend to have more obstacles, aiming to avoid collisions. Scenarios where these methods are essential include exploring unknown areas like disaster sites or underground caves, situations where human pilots might struggle to control the multirotor due to communication issues, and tasks such as surveillance or reconnaissance in indoor settings.

The challenges associated with the mentioned instances can be grouped by the nature of obstacles encountered: moving obstacles such as humans, collaborating robots of similar or different species, and stationary obstacles like walls, pillars, or indoor surfaces. Notably, advancements in indoor collision avoidance and autonomous flight have diverged based on these categories of obstacles. Particularly concerning stationary obstacles, employing tools such as depth cameras or LiDAR on multirotors allows for early detection, generating a map represented as an occupancy grid map. This facilitates proactive planning based on pre-existing mapped data.

Indoor obstacle avoidance and autonomous flight strategies for multirotors, concerning stationary obstacles, commonly involve two main components: global and local planners. Firstly, global planners establish a rough path from the starting point to the intended destination, outlines feasible routes. These planners typically rely on methods such as RRT* that use random searches[13–15], A*, D*, or Dijkstra employs dynamic programming[5, 6, 10, 20, 22], and geometric techniques based on obstacles such as the Generalized-Voronoi-Diagram (GVD) or Medial-Axis Transformation (MAT) [3, 7, 8, 11]. MAT techniques, specifically, play a pivotal role in crafting a safe path from the current position to the target while maintaining a set distance from nearby obstacles. This process concurrently generates a Collision-Free-Graph (CFG)[1] or safe area[12].

Additionally, while the global planner generates a brief global path, the primary objective of the local planner is to perform trajectory planning that avoids obstacles and collisions while following this path, considering the platform's dynamic characteristics. For this purpose, Model Predictive Control (MPC)

is widely used, particularly due to its faster computational speed compared to non-convex or nonlinear solvers. Moreover, Convex Model Predictive Control (CMPC) techniques are widely employed, particularly to leverage the computational time advantage, as extensively explored in the literature, as evidenced by [1, 9, 18]. Previous research has constructed piece-wise convex Safe Flight Corridors (SFCs) by discretizing convex geometries into cylindrical/rectangular or convex polygonal/convex polyhedral elements. However, when formulating optimization problems using such SFCs, complex inequality conditions arise. In convex optimization, the simpler the constraints defining the problem, the faster the solver can solve it. Therefore, a simple formulation of convex problems is crucial.
Meanwhile, the MAT technique, introduced earlier, consists of trajectories that gather the centers of inscribed disks and include information about the radii of each inscribed disk. When applied to spatial graphs described by the Medial Axis (MA), a subset of graph node sequences renders the space piece-wise convex. Additionally, each space comprising the MA consists of circles or spheres, making the construction of inequality constraints extremely simple.
In this paper, inspired by these characteristics of MA, we propose the MAT-CMPC technique and framework. The primary contribution of this proposed framework lies in integrating MAT-based CFG/SFC generation with a set of circles for the global planner and CMPC formulation encompassing constraints arising from surrounding obstacles, dynamic equations, and system saturation for the local planner. This integration is significant as it associates MAT techniques with the creation of SFCs using piece-wise convex yet simple quadratic forms represented by circles, essential for formulating the local planner as a CMPC problem.
The remaining parts of this paper are structured as follows: Chapter 2.introduces the definition of the indoor autonomous flight problem addressed in this study. In Chapter 3. the methodology for constructing CFG and piece-wise convex SFC based on MAT is described. Chapter 4.presents the MAT-CMPC problem incorporating the previously constructed SFC, dynamics, and other constraints, along with an iterative solution approach. Finally, Chapters 5.and 6.analyze the performance of the proposed MAT-CMPC problem solution in terms of constraint appropriateness and optimal trajectory by comparing aspects such as path length and computation time, concluding the research.

2. Problem Definition

Before addressing the issue, two coordinate systems, denoted as $\mathfrak{I}$ and $\mathfrak{B}$, are introduced. These represent the inertial and body frames in 2D, respectively, as depicted in Fig. 1, assuming the multirotor maintains a constant altitude.

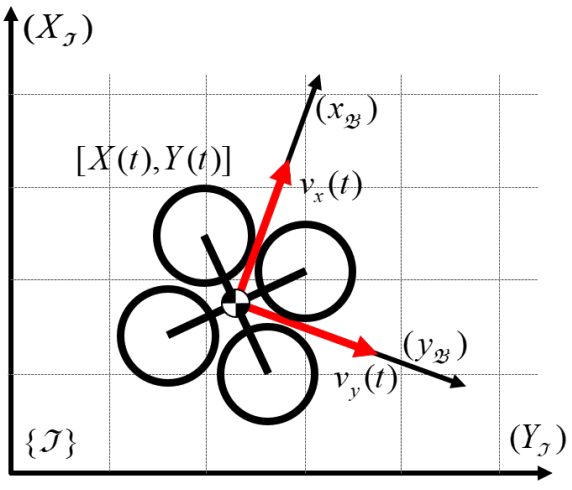

Figure 1 – Definition of coordinate system

Furthermore, the current state variable of the multirotor are denoted by $\mathbf{x} = \left[\mathbf{p}^T, \mathbf{v}^T\right]$, where $\mathbf{p}(t) = [X(t), Y(t)]$ represents the position in frame $\mathfrak{I}$, and $\mathbf{v}(t) = [v_x(t), v_y(t)]$ represents the velocity in frame $\mathfrak{B}$. The control input of the multirotor is defined as the acceleration $\mathbf{u}(t) = \mathbf{a}_{\text{cmd}}(t) = [a_{\text{cmd},x}, a_{\text{cmd},y}]$ in frame $\mathfrak{B}$. We assume that the velocity and acceleration of the multirotor are subject to the following

saturation conditions as:

$$\begin{aligned} \|\mathbf{v}(t)\|_2^2 &\leq v_{\max}^2, \\ \|\mathbf{u}(t)\|_2^2 &\leq a_{\max}^2. \end{aligned} \tag{1}$$

where $v_{\max}$ and $a_{\max}$ denote the maximum allowable velocity and acceleration, respectively. The dynamic equation for the multirotor concerning the previously defined state variables is defined as follows:

$$\dot{x} = f(x,u). \tag{2}$$

Multirotor operations in a globally known map environment assume the presence of only static obstacles, with all obstacle-free spaces represented by $\mathbf{S}_{\text{free}} \subset \mathbb{R}^2$, where $\mathbb{R}$ denotes the set of real numbers. Then, the proposed conditions for ensuring that the multirotor avoids collisions can be defined as follows:

$$\mathbf{p} \subset \mathbf{S}_{\text{free}} \tag{3}$$

The problem addressed in this paper is to enable the multirotor to reach the final destination $\mathbf{p}_{\text{goal}}$ in frame $\mathfrak{I}$ while considering given obstacle information, state variables, and the dynamics of the multirotor. This can be formulated as an iteratively solved optimal control problem, as expressed below:

$$\begin{aligned} &\underset{\mathbf{x}(t),\,\mathbf{u}(t)}{\text{Minimize}} \;\; J(\mathbf{x},\mathbf{u}), \\ &\text{subject to} \;\; \mathbf{x}(t_c) = \mathbf{x}_0, \text{ Eqs. } 1-3,\; t \in [t_c, t_c + T_{\text{pred}}], \end{aligned} \tag{4}$$

where t_c, $\mathbf{x}_0$, and T_{pred} denotes the current time, initial state of multirotor, and the prediction horizon range, respectively. To define the optimal control problem defined above as a convex optimization problem, the following chapters elaborate on deriving each constraint and defining objective functions for CMPC problem.

3. MAT-based Piece-wise convex Safe Flight Corridor

3.1 Medial Axis for Collision Free Graph and Global Path Corridor

In the preceding chapter, we defined $\mathbf{S}_{\text{free}}$ with respect to the given global map. Typically, obstacle information is provided in the form of an occupancy grid map or binary grid, denoted as $\mathbf{M}_{\text{grid}} \subset \mathbb{N}^{r\times s}$, where $\mathbb{N}$ represents the set of natural numbers, comprising all positive integers starting from 1, and $\mathbf{M}_{\text{grid}}$ represents a 2D grid of size $(r \times s)$. As briefly mentioned in Chapter 1,. previous attempts have been made to describe specific spaces based on discrete MAT. Particularly, techniques based on the Euclidean-Distance-Transformation (EDT), as introduced in [16, 17], have been proposed. In this paper, we employ the aforementioned techniques to construct the CFG, denoted as $\mathbf{G}_{\text{CFG}} \subset \mathbb{N}^{r\times s}$, using the discrete MA approach to represent spaces devoid of obstacles, akin to $\mathbf{S}_{\text{free}}$, within the given $\mathbf{M}_{\text{grid}}$. At this point, $\mathbf{G}_{\text{CFG}}$ is defined as follows: for an $r \times s$ grid, the parameters at the centers or skeleton positions of empty spaces have values equal to the distance to the nearest obstacle in that grid, while parameters corresponding to other positions have a value of 0. The Euclidean Distance Map (EDM), $\mathbf{M}_{\text{EDM}}$, and the CFG, $\mathbf{G}_{\text{CFG}}$, with respect to $\mathbf{M}_{\text{grid}}$ are represented as follows:

$$\mathbf{M}_{\text{EDM}} = EDT\left(\mathbf{M}_{\text{grid}}\right) \tag{5}$$

$$\mathbf{G}_{\text{CFG}} = MAT\left(\mathbf{M}_{\text{EDT}}\right) \tag{6}$$

where $EDT(\cdot)$ and $MAT(\cdot)$ denote the conversion processes using the EDT and the MAT, respectively. Furthermore, the global path corridor $\mathbf{C}_{\text{GP}}$ can be computed based on $\mathbf{p}(t_c)$, the current position, $\mathbf{p}_{\text{goal}}$, the global goal position, and $\mathbf{G}_{\text{CFG}}$. This is achieved by generating sequential nodes of the shortest path between $\mathbf{p}(t_c)$ and $\mathbf{p}_{\text{goal}}$ based on $\mathbf{G}_{\text{CFG}}$. As a brief explanation, we first identify the nodes $\mathbf{p}_{\text{CFG,start}}$ and $\mathbf{p}_{\text{CFG,goal}}$ in the graph $\mathbf{G}_{\text{CFG}}$ that are closest to $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{goal}}$, respectively. Subsequently, employing the A* algorithm, we compute the shortest sequence of corridor nodes $\mathbf{C}_{\text{GP},2}$ within $\mathbf{G}_{\text{CFG}}$ between $\mathbf{p}_{\text{CFG,start}}$ and $\mathbf{p}_{\text{CFG,goal}}$. Finally, we concatenate $\mathbf{C}_{\text{GP},2}$ with the shortest diagonal paths $\mathbf{C}_{\text{GP},1}$ between $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{CFG,start}}$, and $\mathbf{C}_{\text{GP},3}$ between $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{CFG,start}}$, which completes $\mathbf{C}_{\text{GP}}$. The algorithm for computing $\mathbf{C}_{\text{GP}}$ is as follows:

Algorithm 1 Global path corridor

Function CALCULATESHORTESTPATH($\mathbf{G}_{\text{CFG}}$, $\mathbf{p}_{\text{start}}$, $\mathbf{p}_{\text{goal}}$)

$\mathbf{p}_{\text{CFG,start}} \leftarrow$ FINDCLOSESTNODE($\mathbf{G}_{\text{CFG}}, \mathbf{p}_{\text{start}}$)
$\mathbf{p}_{\text{CFG,goal}} \leftarrow$ FINDCLOSESTNODE($\mathbf{G}_{\text{CFG}}, \mathbf{p}_{\text{goal}}$)
$\mathbf{C}_{\text{GP},1} \leftarrow$ SHORTESTDIAGONALPATH($\mathbf{M}_{\text{EDM}}, \mathbf{p}_{\text{start}}, \mathbf{p}_{\text{CFG,start}}$)
$\mathbf{C}_{\text{GP},2} \leftarrow$ A*($\mathbf{G}_{\text{CFG}}, \mathbf{p}_{\text{CFG,start}}, \mathbf{p}_{\text{CFG,goal}}$)
$\mathbf{C}_{\text{GP},3} \leftarrow$ SHORTESTDIAGONALPATH($\mathbf{M}_{\text{EDM}}, \mathbf{p}_{\text{start}}, \mathbf{p}_{\text{CFG,goal}}$)
$\mathbf{C}_{\text{GP}} \leftarrow$ CONCATENATE($\mathbf{C}_{\text{GP},1}, \mathbf{C}_{\text{GP},2}, \mathbf{C}_{\text{GP},3}$)
return $\mathbf{C}_{\text{GP}}$

The obtained $\mathbf{C}_{\text{GP}}$ includes not only the positional information of nodes representing the shortest path, but also the obstacle distance information corresponding to each grid position on $\mathbf{M}_{\text{EDM}}$. Thus, $\mathbf{C}_{\text{GP}}$ can be represented using $\mathbf{P}_{\text{GP}}$ and $\mathbf{R}_{\text{GP}}$ as follows:

$$\mathbf{C}_{\text{GP}} = \{[\mathbf{p}_{\text{GP},i},\ R_{\text{GP},i}] \mid i = 1,2,\ldots,n_{\text{GP}}\}. \tag{7}$$

Here, $\mathbf{P}_{\text{GP}}$ is the sequential vector of node positions:

$$\mathbf{P}_{\text{GP}} = \left\{\mathbf{p}_{\text{GP},i} \mid \mathbf{p}_{\text{GP},i} = [X_{\text{GP},i},\ Y_{\text{GP},i}] \in \mathbb{R}^2,\ \text{for } i = 1,2,\ldots,n_{\text{GP}}\right\}, \tag{8}$$

and $\mathbf{R}_{\text{GP}}$ represents the sequence of distances to the nearest obstacles for each node:

$$\mathbf{R}_{\text{GP}} = \left\{R_{\text{GP},i} \mid i = 1,2,\ldots,n_{\text{GP}}\right\}, \tag{9}$$

where n_{GP} represents the number of elements in $\mathbf{C}_{\text{GP}}$.
The examples of $\mathbf{M}_{\text{EDM}}$, $\mathbf{G}_{\text{CFG}}$, and $\mathbf{C}_{\text{GP}}$ with the given $\mathbf{M}_{\text{grid}}$ are illustrated in Figure 2, utilizing the occupancy grid map provided by [21].

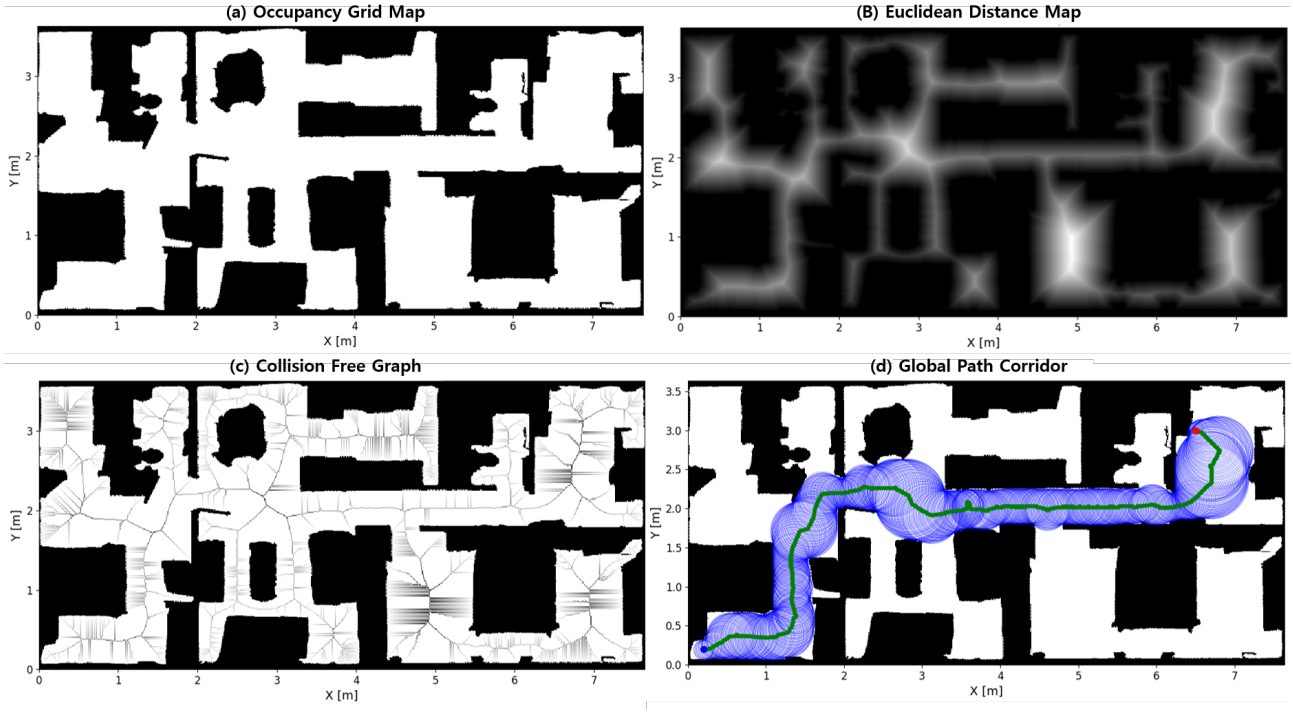

Figure 2 – The example of (a)given map, (b)euclidean distance map, (c)collision free graph, and (d)global path corridor

3.2 Piece-wise convex SFC with Prediction Horizon Update

With given t_c, $\mathbf{p}(t_c)$, $T_{\text{pred},0}$, and $\mathbf{C}_{\text{GP}}$, a SFC in range of $T_{\text{pred},0}$ can be defined as $\mathbf{C}_{\text{SFC}}(t_c)$. To obtain $\mathbf{C}_{\text{SFC}}(t_c)$, we first discretize the originally given prediction horizon $T_{\text{pred},0}$ to map $\mathbf{C}_{\text{SFC}}(t_c)$ with discretely defined $\mathbf{C}_{\text{GP}}$ as below:

$$T_{\text{pred},0} = n_{\text{pred},0}\Delta t, \tag{10}$$

where Δt and $n_{\text{pred},0}$ denotes the time step-size and the original number of discretized prediction horizon, respectively. Then, we find the node on $\mathbf{C}_{\text{GP}}$ closest to $\mathbf{p}(t_c)$ denoted as $\mathbf{p}_{\text{near}}(t_c)$ and the diagonal node sequence with the nearest distance with obstacles $\mathbf{C}_{\text{SFC},1}(t_c)$, adopting the same method as the "FINDCLOSESTNODE" and "SHORTESTDIAGONALPATH" in Algorithm 1. Additionally, by connecting $\mathbf{C}_{\text{SFC},1}(t_c)$ with the node sequence in $\mathbf{N}_{\text{GP}}$ following the index of $\mathbf{p}_{\text{near}}(t_c)$, a temporal path and distance sequence $\mathbf{C}_{\text{SFC,tmp}}(t_c)$ is created for the current multirotor position $\mathbf{p}(t_c)$. Finally, $\mathbf{C}_{\text{SFC}}(t_c)$ is defined as the augmented system of $\mathbf{p}_{\text{SFC}}$ and R_{SFC} by taking the initial elements of $\mathbf{C}_{\text{SFC,tmp}}(t_c)$ up to the number of the prediction horizon. However, as $\mathbf{p}(t_c)$ approaches $\mathbf{p}_{\text{goal}}$, there may be fewer elements in $\mathbf{C}_{\text{SFC,tmp}}(t_c)$ than the initially given number of time steps $n_{\text{pred},0}$. In such cases, the time step number n_{pred} is updated to the number of elements in $\mathbf{C}_{\text{SFC,tmp}}(t_c)$. Otherwise, the initially given value $n_{\text{pred},0}$ is used as n_{pred}. Obtained $\mathbf{C}_{\text{SFC}}$ can be represented using $\mathbf{P}_{\text{SFC}}$ and $\mathbf{R}_{\text{SFC}}$ as follows:

$$\mathbf{C}_{\text{SFC}} = \left\{ [\mathbf{p}_{\text{SFC},i},\, R_{\text{SFC},i}] \mid i = 1,2,\ldots,n_{\text{pred}} \right\}, \tag{11}$$

where $\mathbf{P}_{\text{SFC}}$ and $\mathbf{R}_{\text{SFC}}$ are the sequential vector of positions and radius as:

$$\mathbf{P}_{\text{SFC}} = \left\{ \mathbf{p}_{\text{SFC},i} \mid \mathbf{p}_{\text{SFC},i} = [X_{\text{SFC},i},\, Y_{\text{SFC},i}] \in \mathbb{R}^2, \text{ for } i = 1,2,\ldots,n_{\text{pred}} \right\}, \tag{12}$$

$$\mathbf{R}_{\text{SFC}} = \left\{ R_{\text{SFC},i} \mid i = 1,2,\ldots,n_{\text{pred}} \right\}. \tag{13}$$

The pseudo-code for this process is provided below:

Algorithm 2 Definition of safe flight corridor (SFC)

Function CALCULATESFC($\mathbf{C}_{\text{GP}}$, $\mathbf{M}_{\text{EDM}}$, $\mathbf{p}(t_c)$, $n_{\text{pred},0}$, Δt)

Initialization: $\mathbf{C}_{\text{SFC}}(t_c) \leftarrow []$
$\mathbf{p}_{\text{near}}(t_c) \leftarrow$ FINDCLOSESTNODE($\mathbf{C}_{\text{GP}}, \mathbf{p}(t_c)$)
$\mathbf{C}_{\text{SFC},1}(t_c) \leftarrow$ SHORTESTDIAGONALPATH($\mathbf{M}_{\text{EDM}}, \mathbf{p}(t_c), \mathbf{p}_{\text{near}}(t_c)$)
$\mathbf{C}_{\text{SFC,tmp}}(t_c) \leftarrow$ AUGMENTPATHCURRENT2GOAL($\mathbf{C}_{\text{SFC},1}(t_c), \mathbf{C}_{\text{GP}}, \mathbf{p}_{\text{near}}(t_c)$)
if length($\mathbf{C}_{\text{SFC,tmp}}(t_c)$) $< n_{\text{pred},0}$ **then**
    $n_{\text{pred}} \leftarrow$ length($\mathbf{C}_{\text{SFC,tmp}}(t_c)$)
    $r_{\text{flag}} \leftarrow 0$
else
    $n_{\text{pred}} \leftarrow n_{\text{pred},0}$
    $r_{\text{flag}} \leftarrow 1$
for $i = 1$ to n_{pred} **do**
    $\mathbf{C}_{\text{SFC}}(t_c) \leftarrow$ Append i^{th} element of $\mathbf{C}_{\text{SFC,tmp}}(t_c)$
return $\mathbf{C}_{\text{SFC}}(t_c)$, n_{pred}, r_{flag}

In the above algorithm flow, r_{flag} is a flag variable that indicates whether $\mathbf{C}_{\text{SFC}}$ includes $\mathbf{p}_{\text{goal}}$. The subspace comprising $\mathbf{C}_{\text{SFC}}(t_c)$ consists of circles, making it piece-wise convex, which is crucial for defining the subsequent CMPC problem. The example of $\mathbf{C}_{\text{SFC}}(t_c)$ for arbitrarily given $\mathbf{p}(t_c)$is illustrated in the Figure. 3.

4. MAT-based Convex MPC

4.1 Definition of Constraints

As a equality constraint, the discretized dynamic equations are first introduced. For convenience, nominal dynamics are utilized in a linearized form. By utilizing Eq. 10 for the state variables $\mathbf{x}$

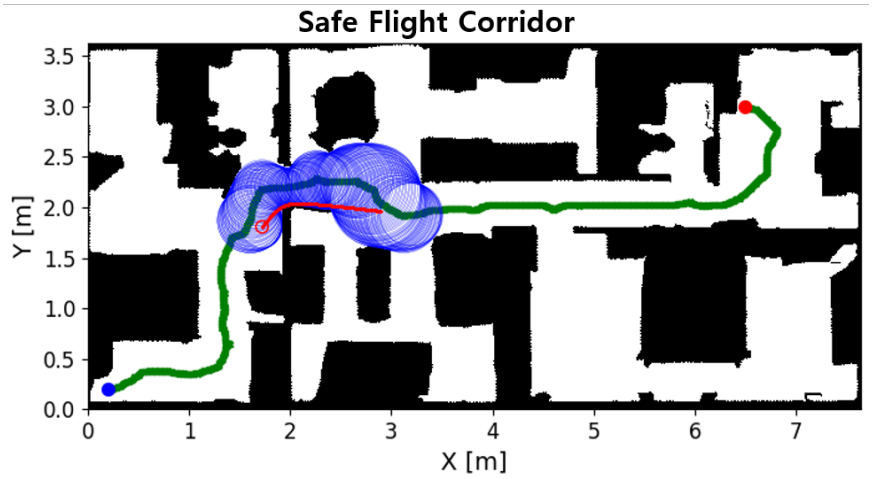

Figure 3 – Example of safe flight corridor

introduced in Chapter 2.the discretization can be represented as follows:

$$\mathbf{x}_k(t_c) = \mathbf{A}\mathbf{x}_{k-1}(t_c) + \mathbf{B}\mathbf{u}_{k-1}(t_c) = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k(t_c) + \begin{bmatrix} 0 & 0 \\ \Delta t/m & 0 \\ 0 & 0 \\ 0 & \Delta t/m \end{bmatrix} \mathbf{u}_k(t_c), \tag{14}$$

for

$$k \in [1, \ldots, n_{\text{pred}}] \tag{15}$$

where discretized state $\mathbf{x}_k(t_c)$ and control input $\mathbf{u}_k(t_c)$ are defined as:

$$\mathbf{x}_i(t_c) = [\mathbf{p}_i(t_c) \;\; \mathbf{v}_i(t_c))]^T = \mathbf{x}(t_c + i\Delta t), \; i \in [1, \ldots, n_{\text{pred}}], \tag{16}$$

$$\mathbf{u}_j(t_c) = \mathbf{u}(t_c + j\Delta t), \; j \in [0, \ldots, n_{\text{pred}-1}], \tag{17}$$

and m denotes the mass of multirotor. Furthermore, to ensure continuity in the state of the multirotor, the first element of the discretized state variable sequence must be equal to the current state of the multirotor. This can be expressed as follows:

$$\mathbf{x}_0(t_c) = \mathbf{x}(t_c). \tag{18}$$

Inequality constraints can be broadly divided into two categories: saturation constraints on state or control variables, and conditions regarding positions where the multirotor should not collide. Firstly, considerations for saturation can be defined simply using Eq. 1. Additionally, the conditions regarding positions, as expressed in Chapter 2.by Eq. 3, can be detailed for the SFC represented by Eq. 11 and discretized $\mathbf{x}_k$ over n_{pred} elements as follows:

$$||\mathbf{x}_k - \mathbf{p}_{\text{SFC},k}||_2^2 \leq R_{\text{SFC},k}^2, \quad \forall k \in \{1, 2, \ldots, n_{\text{pred}}\} \tag{19}$$

To ensure that the multirotor avoids collisions, each $\mathbf{x}_k$ must reside within the corridors of $\mathbf{C}_{\text{SFC}}$, which comprise individual convex-shaped regions (or circles) within.
Meanwhile, only when $r_{\text{flag}} = 1$, the following inequality constraint is additionally given to ensure that the multirotor approaches and stops near $\mathbf{p}_{\text{goal}}$:

$$\|\mathbf{p}_{n_{\text{pred}}} - \mathbf{p}_{\text{goal}}\|_2^2 \leq \varepsilon_{\text{pos}}^2 \tag{20}$$

$$\|\mathbf{v}_{n_{\text{pred}}}\|_2^2 \leq \varepsilon_{\text{vel}}^2, \tag{21}$$

where ε_{pos} and ε_{vel} denote the thresholds for the position and velocity of the multirotor's last state, respectively.

4.2 Iterative MAT-CMPC Problem Formulation

The constraints defined earlier can all be represented as affine or quadratic sets, allowing for the formulation of CMPC when the cost function is defined in a quadratic form. The cost function depending on r_{flag} is as follows:

$$J = \begin{cases} \sum_{k=0}^{n_{\text{pred}}-1} \|\mathbf{u}_k\|_2^2 + \sum_{k=1}^{n_{\text{pred}}} \|\mathbf{v}_k\|_2^2 + \|\mathbf{p}_{n_{\text{pred}}} - \mathbf{p}_{\text{SFC},n_{\text{pred}}}\|^2 & \text{, if } (r_{\text{flag}} = 0), \\ \sum_{k=0}^{n_{\text{pred}}-1} \|\mathbf{u}_k\|_2^2 + \sum_{k=1}^{n_{\text{pred}}} \|\mathbf{v}_k\|_2^2 & \text{, otherwise .} \end{cases} \tag{22}$$

Therefore, with the constraints and cost function defined above, we can formulate CMPC as follows:

$$\begin{aligned} &\underset{\mathbf{x}_k,\, \mathbf{u}_{k-1}}{\text{Minimize}} \;\; J(\mathbf{x},\mathbf{u}), \\ &\text{subject to} \;\; \mathbf{x}(t_c) = \mathbf{x}_0, \text{ Eqs. } 1,\ 14,\ 18-21,\ k \in [1,2,\ldots,n_{\text{pred}}], \end{aligned} \tag{23}$$

which is called MAT-CMPC. Furthermore, assuming iterative solving of the CMPC problem Eq. 23 would enable the multirotor to reach and stop near $\mathbf{p}_{\text{goal}}$. Representing this process in an algorithmic chart yields the following:

Algorithm 3 Iterative MAT-CMPC

Given : $\mathbf{C}_{\text{GP}}$, $\mathbf{M}_{\text{EDM}}$, $n_{\text{pred},0}$, $n_{\text{ctrl},0}$, $\mathbf{p}_{\text{goal}}$, Δt, ε_{pos}, ε_{vel}

Initialize: t_c, $\mathbf{x}(t_c)$, n_{pred}, n_{ctrl}
while $\|\mathbf{p}_0 - \mathbf{p}_{\text{goal}}\|_2 > \varepsilon_{\text{pos}}$ & $\|\mathbf{v}_0\|_2 > \varepsilon_{\text{vel}}$ **do**
    $\mathbf{C}_{\text{SFC}}(t_c)$, n_{pred}, $r_{\text{flag}} \leftarrow$ CALCULATESFC$(\mathbf{C}_{\text{GP}}, \mathbf{M}_{\text{EDM}}, \mathbf{p}(t_c), n_{\text{pred},0}, \Delta t)$
    $\mathbf{x}_k^*$, $\mathbf{u}_{k-1}^* \leftarrow$ Convex MPC from Eq. 23, $k \in [1,2,\ldots,n_{\text{pred}}]$
    if $n_{\text{ctrl},0} \leq n_{\text{pred}}$ **then**
        $n_{\text{ctrl}} \leftarrow n_{\text{ctrl},0}$
    else
        $n_{\text{ctrl}} \leftarrow n_{\text{pred}}$
    $\mathbf{x}_0(t_c) \leftarrow \mathbf{x}_{n_{\text{ctrl}}}^*$
    $t_c \leftarrow t_c + n_{\text{ctrl}}\Delta t$

5. Simulation Setup and Result

5.1 Environment Setup

For validation through simulation, the CPU environment utilized was Ryzen7 7700x, along with Python 3.8 and the "scikit-image" library [19]. The CMPC problem was solved using the MOSEK solver [2] with [4]. Moreover, the constants used to solve this problem are listed in Table 1 as follows:

Table 1 – Initial parameters for simulations

Parameter	Value
$n_{\text{pred},0}$, $n_{\text{ctrl},0}$	200, 10
ε_{pos}, ε_{vel}	0.001 [m], 0.2 [m/s]
Δt	0.1 [sec]
m	2 [kg]
g	9.807 [m/s^2]
v_{max}	1 [m/s]
a_{max}	$g\sin(\pi/6)$ [m/s^2]

The occupancy grid environment used in the simulation is the same as that utilized in Figure 2, and the set of $\mathbf{p}_{\text{start}}$ and $\mathbf{p}_{\text{goal}}$ for performance validation is identical to Table 2. Additionally, it is assumed that the multirotor is stationary at the beginning of the simulation.

Table 2 – Start and goal position for each scenario

Scenario #	$\mathbf{p}_{\text{start}}\ [m]$	$\mathbf{p}_{\text{goal}}\ [m]$
1	$[0.20 \quad 0.20]^T$	$[6.60 \quad 3.00]^T$
2	$[0.20 \quad 0.20]^T$	$[7.00 \quad 1.00]^T$
3	$[0.36 \quad 3.05]^T$	$[6.60 \quad 3.00]^T$
4	$[0.36 \quad 3.05]^T$	$[7.00 \quad 1.00]^T$

5.2 Simulation Result

The trajectory outcomes corresponding to the scenarios outlined in Table 2 are depicted in Figure 4. In this figure, we present the trajectory results obtained using the MAT-CMPC proposed in this paper, applied with the given $n_{\text{pred},0}$. Additionally, we include the optimal trajectory (OT) results calculated by assuming the global path corridor $\mathbf{C}_{\text{GP}}$ as $\mathbf{C}_{\text{SFC}}$, for the purpose of comparing trajectories and paths across each scenario. Additionally, the maximum and minimum values of control variables and

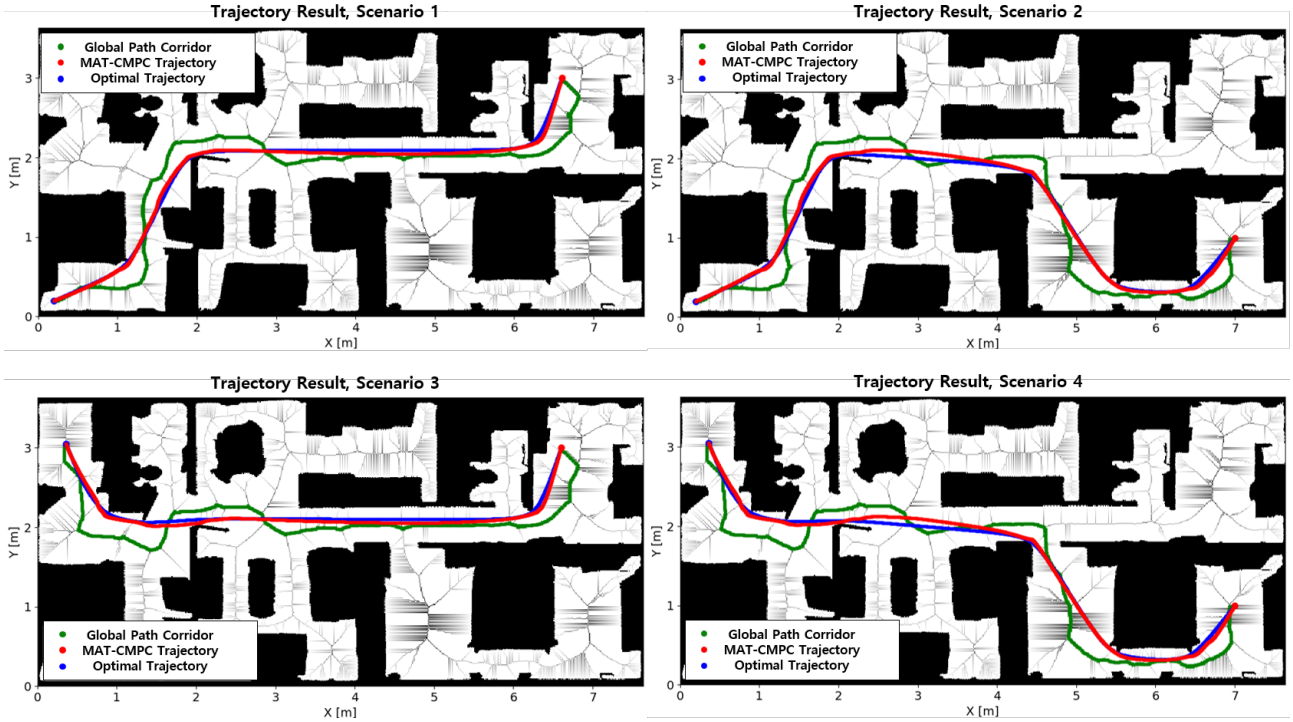

Figure 4 – MAT-CMPC and optimal trajectory result

velocity for each iteration are depicted in Figures 5-6 to ensure the satisfaction of constraints. Furthermore, in Table 3, it can be observed that the length of the trajectory traversed by the multirotor using the proposed MAT-CMPC is approximately 1% longer compared to the optimal trajectory. This indicates that there is not a significant difference in trajectory length compared to the optimal trajectory.

Table 3 – Comparison of trajectory length

Scenario #	MAT-CMPC Length(A) [m]	OT Length(B) [m]	Ratio(A/B-1) [%]
1	7.8972	7.8337	0.8114
2	8.8086	8.7147	1.0775
3	7.4224	7.3078	1.5686
4	8.2937	8.1853	1.3212

Below in Figure 7, we illustrate the time taken to compute $\mathbf{x}^*$ and $\mathbf{u}^*$ for each step of MAT-CMPC

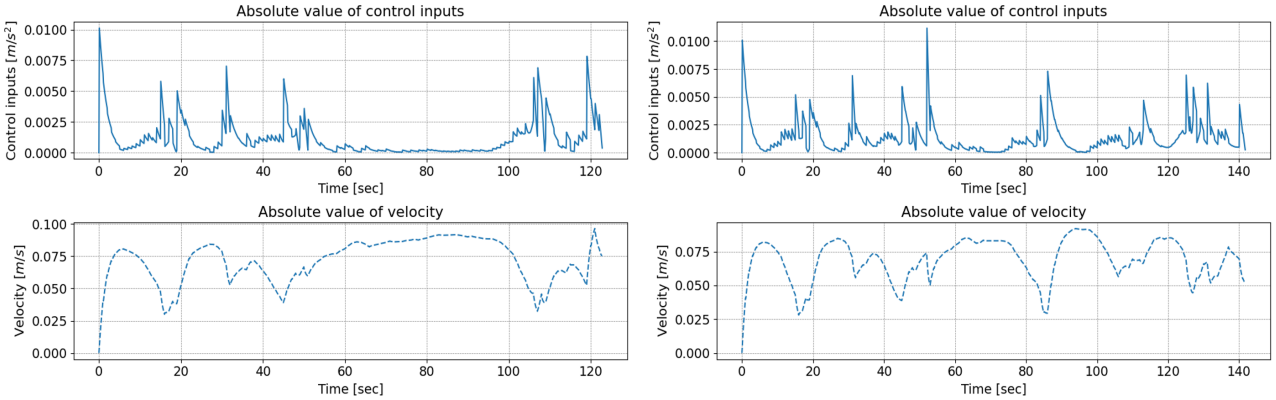

Figure 5 – Velocity and acceleration command in scenario 1(left) & 2(right)

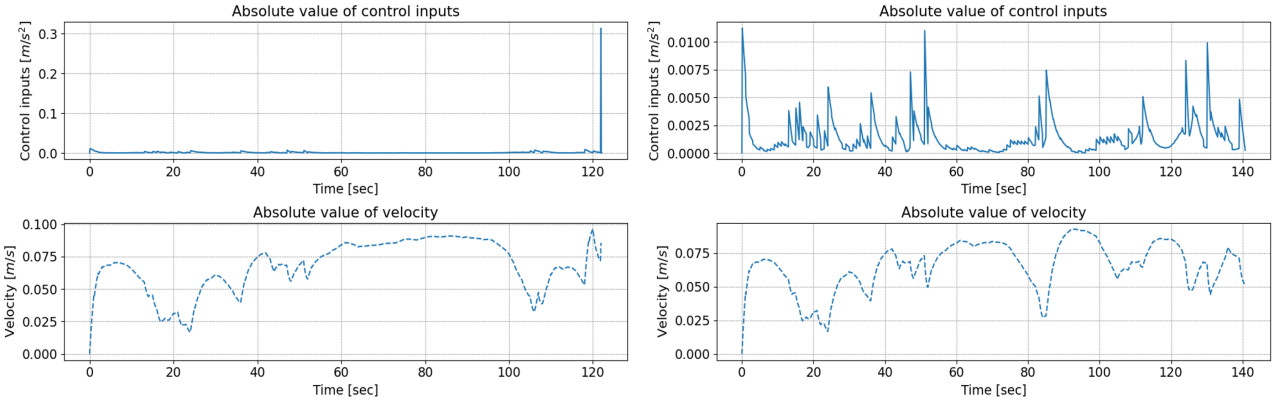

Figure 6 – Velocity and acceleration command in scenario 3(left) & 4(right)

and the optimal trajectory techniques. For MAT-CMPC, multiple calculation steps are necessary for the multirotor to reach $\mathbf{p}_{\text{goal}}$, hence, we display the average and the maximum/minimum computation times for each step within a scenario. This illustrates that the average computation time of MAT-CMPC is approximately five times faster.

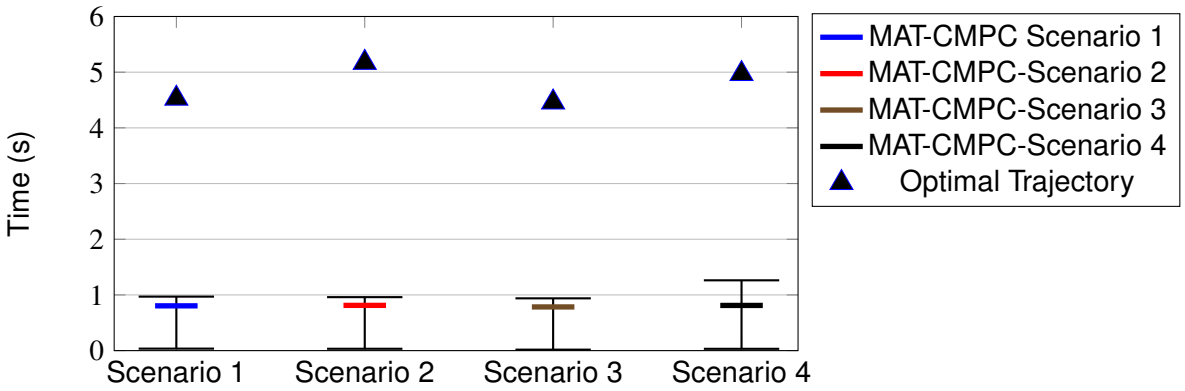

Figure 7 – Comparison of calculation time for each scenario with MAT-CMPC and optimal trajectory

6. Conclusion

The proposed MAT-CMPC technique is based on MAT to generate global path corridor and safe flight corridor, enabling the creation of CMPC. In this paper, validation of the proposed approach's performance is demonstrated through promising results in various aspects of trajectory planning tasks.

The trajectory obtained with MAT-CMPC showcase the effectiveness of the proposed approach in

generating feasible and smooth trajectories. By comparing these results with the optimal trajectory (OT) results, we can assess the performance of MAT-CMPC across various scenarios.
Furthermore, the analysis of trajectory lengths reveals that the trajectories produced by MAT-CMPC are only marginally longer (approximately 1%) than the optimal trajectories. This indicates that MAT-CMPC is capable of generating trajectories that closely approximate the optimal solutions, thereby ensuring efficient path planning.
Additionally, the computation time comparison highlights the computational efficiency of MAT-CMPC. Despite the need for multiple calculation steps to reach the goal, MAT-CMPC exhibits an average computation time approximately five times faster than the optimal trajectory technique. This emphasizes the practical viability of MAT-CMPC for real-time trajectory planning applications.
In conclusion, the MAT-CMPC approach, by leveraging MAT to define the Global Path Corridor/Safe Flight Corridor and formulating trajectory planning as a convex optimization problem, offers a promising solution for addressing complex trajectory planning tasks in various scenarios. Its ability to generate near-optimal trajectories efficiently makes it a valuable tool for autonomous navigation systems, particularly in dynamic environments where real-time decision-making is crucial.

7. Acknowledgement

This research was supported by the Challengeable Future Defense Technology Research and Development Program (No.912766601) of Agency for Defense Development in 2024.

8. Contact Author Email Address

If you have any questions, please send an email to:
'baoro1995@ kaist.ac.kr' (the first author) or 'hcbang@kaist.ac.kr' (the corresponding author).

9. Copyright Statement

References

[1] H. Ahn, J. Park, H. Bang, and Y. Kim. Model predictive control-based multirotor three-dimensional motion planning with point cloud obstacle. *Journal of Aerospace Information Systems*, 19(3):179–193, 2022.

[2] E. D. Andersen and K. D. Andersen. The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In *High performance optimization*, pages 197–232. Springer, 2000.

[3] D. Coeurjolly and A. Montanvert. Optimal separable algorithms to compute the reverse euclidean distance transformation and discrete medial axis in arbitrary dimension. *IEEE transactions on pattern analysis and machine intelligence*, 29(3):437–448, 2007.

[4] S. Diamond and S. Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[5] D. Ferguson, M. Likhachev, and A. Stentz. A guide to heuristic-based path planning. In *Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS)*, pages 9–18, 2005.

[6] A. K. Guruji, H. Agarwal, and D. Parsediya. Time-efficient a* algorithm for robot path planning. *Procedia Technology*, 23:144–149, 2016.

[7] L. Han, F. Gao, B. Zhou, and S. Shen. Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4423–4430. IEEE, 2019.

[8] Q. Hou, S. Zhang, S. Chen, Z. Nan, and N. Zheng. Straight skeleton based automatic generation of hierarchical topological map in indoor environment. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2229–2236. IEEE, 2021.

[9] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar. Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments. *IEEE Robotics and Automation Letters*, 2(3):1688–1695, 2017.

[10] M. Luo, X. Hou, and J. Yang. Surface optimal path planning using an extended dijkstra algorithm. *IEEE access*, 8:147827–147838, 2020.

[11] E. Masehian and M. Amin-Naseri. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *Journal of Robotic Systems*, 21(6):275–300, 2004.

[12] E. Masehian and M. Amin-Naseri. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *Journal of Robotic Systems*, 21(6):275–300, 2004.

[13] K. Naderi, J. Rajamäki, and P. Hämäläinen. RT-RRT* a real-time path planning algorithm based on RRT. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, pages 113–118, 2015.

[14] I. Noreen, A. Khan, and Z. Habib. A comparison of RRT, RRT* and RRT*-smart path planning algorithms. *International Journal of Computer Science and Network Security (IJCSNS)*, 16(10):20, 2016.

[15] I. Noreen, A. Khan, and Z. Habib. Optimal path planning using RRT* based approaches: a survey and future directions. *International Journal of Advanced Computer Science and Applications*, 7(11), 2016.

[16] E. Remy and E. Thiel. Exact medial axis with euclidean distance. *Image and Vision Computing*, 23(2): 167–175, 2005.

[17] A. V. Saúde, M. Couprie, and R. A. Lotufo. Discrete 2d and 3d euclidean medial axis in higher resolution. *Image and Vision Computing*, 27(4):354–363, 2009.

[18] J. Tordesillas and J. P. How. FASTER: Fast and safe trajectory planner for navigation in unknown environments. *IEEE Transactions on Robotics*, 2021.

[19] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

[20] C. W. Warren. Fast path planning using modified a* method. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 662–667. IEEE, 1993.

[21] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9068–9079, 2018.

[22] J. Yu, M. Yang, Z. Zhao, X. Wang, Y. Bai, J. Wu, and J. Xu. Path planning of unmanned surface vessel in an unknown environment based on improved d* lite algorithm. *Ocean Engineering*, 266:112873, 2022.