공학석사학위논문

저수지 청소를 위한 무인수상정의 경로계획, 운항제어 및 구현

Planning, control and implementation of USV for lake cleaning operation

2023 년 8 월

서울대학교 대학원

기계공학부

정 석 진

저수지 청소를 위한 무인수상정의 경로계획, 운항제어 및 구현

Planning, control and implementation of USV for lake cleaning operation

지도교수 이 동 준

이 논문을 공학석사 학위논문으로 제출함

2023 년 4 월

서울대학교 대학원

기계공학부

정 석 진

정석진의 공학석사 학위논문을 인준함

2023 년 6 월

위 원 장 : 이 경 수 (인)

부위원장 : 이 동 준 (인)

위 원 : 차 석 원 (인)

Abstract

Planning, control and implementation of USV for lake cleaning operation

Seokjin Jeong
Mechanical Engineering
The Graduate School
Seoul National University

In this paper, we introduce planning, control and implementation of unmanned surface vehicle for lake cleaning operations. Cleaning path with minimum turns are generated on non convex polygons to ensure efficient coverage on complex target regions. Path following control based on vector field tracking is implemented for robust convergence to generated path. Proposed planning and control are implemented on custom USV capable of outdoor mapping and localization. Simulation and Experiment results are given to validate constructed system performance.

Keywords: Unmanned surface vehicle (USV), Autonomous cleaning, Path following, Optimal coverage
Student Number: *2021-21797*

Contents

List of Figures

Abbreviations

USV	**U**nmanned **S**urfacel **V**ehicle
UAV	**U**nmanned **A**erial **V**ehicle
SLAM	**S**imultaneous **L**ocalization **A**nd **M**apping
GPS	**G**lobal **P**ositioning **S**ystem
RTK GPS	**R**eal **T**ime **K**inematic **G**lobal **P**ositioning **S**ystem
LIDAR	**LI**ght **D**etection **A**nd **R**anging
IMU	**I**nertial **M**easurement **U**nit
ESC	**E**lectronic **S**peed **C**ontrollers
MCU	**M**icro **C**ontroller **U**nit
PWM	**P**ulse **W**idth **M**odulation
EKF	**E**xtended **K**alman **F**ilter
TSP	**T**raveling **S**alesman **P**roblem
DOF	**D**egree **O**f **F**reedom

Symbols

$\mathbf{d}$	Altitude Length
$\mathbf{k_i}$	Diagonal Length
θ	Coverage Path Angle
$\mathbf{P}$	Polygon Instance
$\mathbf{\Omega_P}$	Polygon Configuration represented as Set of Polygons
$S(\mathbf{\Omega_P})$	Minimum Altitude of Configuration $\mathbf{\Omega_P}$
$C(\mathbf{\Omega_P})$	Minimum Altitude of Merged Configuration of $\mathbf{\Omega_P}$
$S(\mathbf{P_1^i}), S(\mathbf{P_1^i})$	Minimum Altitude of i-th possible Two Splitted Configuration of $\mathbf{\Omega_P}$
$\mathcal{C}$	Desired Path
$\mathbf{x}(t)$	Current Position of Vehicle at Time t
$\mathbf{s}(t)$	Nearest Point on $\mathcal{C}$ from $\mathbf{x}(t)$
$\mathbf{\Psi}(\mathbf{x}(t))$	Vector Field Generated from $\mathbf{x}(t)$
$\mathbf{D}(\mathbf{x}(t))$	Convergent Component of $\mathbf{\Psi}(\mathbf{x}(t))$
$\mathbf{T}(\mathbf{x}(t))$	Convergent Component of $\mathbf{\Psi}(\mathbf{x}(t))$

k_f	Convergent Coefficient of $\mathbf{\Psi}(\mathbf{x}(t))$
$G(\mathbf{x}(t))$	Convergent Weight Function Defined using $D(\mathbf{x}(t))$
$H(\mathbf{x}(t))$	Traversal Weight Function Defined using $G(\mathbf{x}(t))$
η	Norm Coefficient of $\mathbf{\Psi}(\mathbf{x}(t))$
$x(t)$, $y(t)$	2-D Current Position of Vehicle at Time t
$\psi(t)$	Yaw Angle of Vehicle at Time t
u_x, u_y	Desired Linear Control Input
τ_z	Desired Angular Control Input
ψ_d	Desired Yaw Angle
$\mathbf{M}$	Mass of Vehicle
$\mathbf{I}$	Moment of Inertia of Vehicle
$\mathbf{R}_\psi$	Rotation Matrix by ψ
δ_u	Linear External Disturbance
δ_τ	Angular External Disturbance
$k_{p,i,d,p\psi,i\psi,d\psi}$	Linear and Angular PID Gains
$e_{v,\psi}$	Velocity and Angular Error
k_x, k_y, k_ψ	Drag Force Coefficients

Chapter 1

Introduction

1.1 Motivation

Contamination of lake is critical to human lives since they are essential to nearby residents as a main source of drinking water. It is usually caused by disposal of household or industrial waste and excessive growth of algae in enclosed water. Therefore, repetitive cleaning of pollutants is required to keep the water clean. Unmanned Surface Vehicles (USVs) are widely used in lake cleaning applications due to its ability to perform cleaning operations without human effort.

Although most industrial USVs are remote controlled, products with automated solutions such as clearbot [1] and MC120 [2] are being developed. Clearbot [1] developed by open ocean engineering provides automated waste cleaning using

onboard sensors. It generates cleaning path based on predefined waypoints and provides obstacle avoidance during cleaning operation. MC120 [2] devloped by oceanalpha provides cleaning through area coverage. It plans cleaning path on predefined polygon areas through back and forth motion. However these solutions are restricted to simple target area configuration and do not consider optimality in path planning. Clearbot does not consider position and sequence of waypoints to execute optimal cleaning on given area. MC120 produces complete coverage path for simple polygons but fails to provide efficient coverage. Examples of the products are shown in Fig 1.1 and Fig 1.2. Therefore, we develop a novel cleaning USV capable of optimal cleaning operations in complex polygon areas. Simulations and experiments on real world lake are also executed to verify the performance of the proposed system.

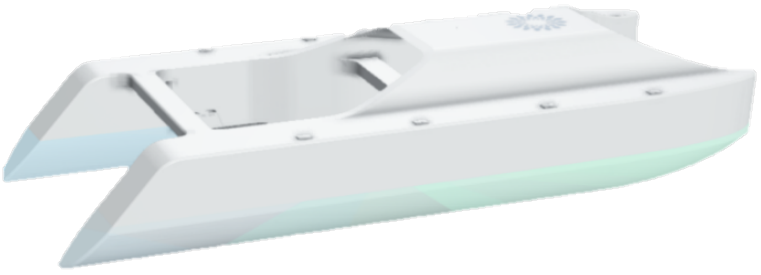


FIGURE 1.1: Clearbot

FIGURE 1.2: MC120

1.2 Related Works

There have been various researches developing USVs for cleaning applications. Luo et al [3] proposed spill coverage path planning for USVs based on image processing. They partitioned the spill area in square cells and formulated coverage into traveling salesman problem (TSP), calculating shortest path visiting all cells at least once. However their apporach was limited to approximate representation of the target region, incapable of exact representation of the target environment. Chang et al [4] developed a multi-function USV capable of obstacle avoidance, water-quality monitoring, and water surface cleaning. Since cleaning was implemented by simply moving to nearest garbage from current position, optimality was not considered. Zhu et al [5] designed a fully autonomous cleaning robot SMURF and proposed a novel coverage path planning based on triangular decomposition. Controller based on nonlinear model predictive control was also

implemented to consider changes in USV dynamics while cleaning. Zhu's method suceeded in defining optimal cleaning path for decomposed convex polygons, but failed to find optimal visiting sequence of each polygon.

Optimal coverage path planning has been widely researched for mobile robot applications. Mannadiar et al [6] introduced an optimal coverage algorithm based on boustrophedon cellular decomposition. Optimal euler tour visiting decomposed cells at least once with minimum cost was found, but their approach was limited to predefined sweep directions. Xu et al [7] adapted mannadiar's algorithm using an unmanned aerial vehicle UAV. Their method succeeded in executing coverage mission in real world terrains, but showed low coverage quality due to the effect of external disturbances and hardware actuation limitations. Huang [8] provided method to find optimal polygon decompositions by establishing optimal line sweep path for convex regions and then extending to non convex regions. However convex decomposition was done by simply lengthening all edges, which caused inefficient decomposition with excessive number of cells. Bochkarev et al [9] developed improved results by adding additional points to inner non convex points. They searched for optimal decomposition through itereation and formulated a generalized TSP visiting decomposed polygon coverage paths.

Cleaning path considering return home operation has also been researched. Shnaps et al [10] introduced battery powered coverage algorithm considering battery constraints. Equipotential contours composed of circular arcs was caculcated, and coverage was performed by sequentially following contours with lower potentials.

However, shnaps assumed battery usage to be only dependent on path length, ignoring energy consumption from velocity direction change during circular arcs.

1.3 Contribution

In this paper, we introduce complete USV system to solve problems in contemporary solutions. Cleaning path planning capable of cleaning complex non convex areas with optimal coverage based on minimum turns is proposed. Robust path following control is implemented to ensure convergence to generated path with external disturbances on water. Proposed path planning and path following control is integrated on a cleaning USV platform capable of outdoor mapping and localization. Results on simulation and real world environments are given to validate the performance of the proposed USV system.

The rest of the paper is organized as follows. In chapter 2, we introduce our cleaning path planning algorithm capable of covering general non convex polygons with optimality based on minimum turns. In chapter 3, we introduce path following control based on path convergent vector fields. In chapter 4, we propose novel cleaning USV platform capable of outdoor mapping and localization, mounted with planning and control from chapter 2 and 3. Hardware specifications are given with explanation of each function module. In chapter 5 simulation and real world experiment are conducted to verify the performance of the proposed

USV platform. Finally, conclusion and future works including reactive obstacle avoidance and multi robot coverage are given in chapter 6.

Chapter 2

Cleaning Path Planning

In this chapter, cleaning path planning for arbitrary polygon area with minimum number of turns is proposed. Target regions are given as manually designated polygons on lake. First, optimal coverage path direction is found for convex polygons based on minimum number of turns. Then minimal convex decomposition is executed to decompose non convex regions to convex regions. After decomposition, dynamic programming is implemented to find optimal polygon formulation. Lastly optimal TSP is generated to find tour visiting polygons with least cost. Overall process is illustrated in Fig 2.1

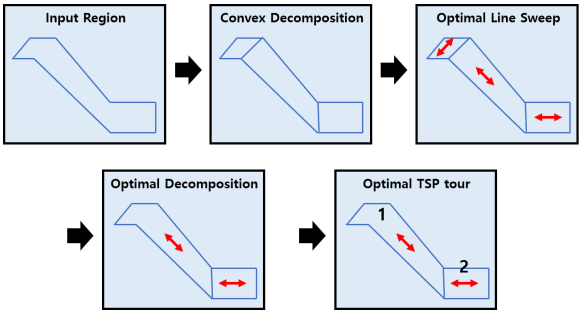


FIGURE 2.1: Cleaning path planning overview

2.1 Optimal Line Sweep in Convex Regions

In this section, we find optimal coverage path $\mathcal{P}$ with direction θ for convex polygons.

Definition 1. Given single convex region, we define back and forth coverage path $\mathcal{P}$ consisted of equally spaced parallel lines with interval $\mathbf{l}$. Coverage path direction θ is as angle of straight lines in $\mathcal{P}$. Visual representation of $\mathcal{P}$ and θ are shown in Fig 2.2. ■

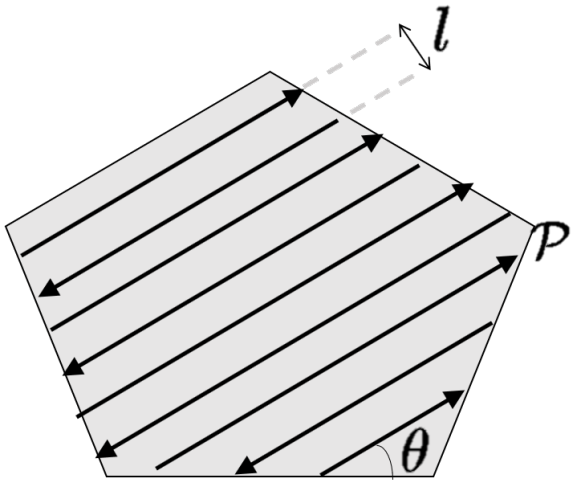


FIGURE 2.2: Coverage path

Coverage cost of path $\mathcal{P}$ can be expressed as path length and number of turns. If complete coverage is assumed, path length is determined by target region area and is identical for all coverage paths. Therefore, optimal coverage problem could be reformulated to minimization of number of turns, which is proportional to altitude length $\mathbf{d}$ defined by Huang [8].

Therefore, we search for coverage path angle θ_{min} minimizing altitude length $\mathbf{d}$ which is expressed as follows.

Definition 2. Altitude length $\mathbf{d}(\theta)$ refers to distance between first line and last line of coverage path. As shown in Fig 2.3, it can be expressed as distance between lowest and highest point of rolled polygon. Given coverage path angle θ, γ, and

i-th diagonal length $\mathbf{k_i}$ it is defined as follows.

$$\mathbf{d} = \mathbf{k_i} sin(\theta + \gamma) \tag{2.1}$$

■

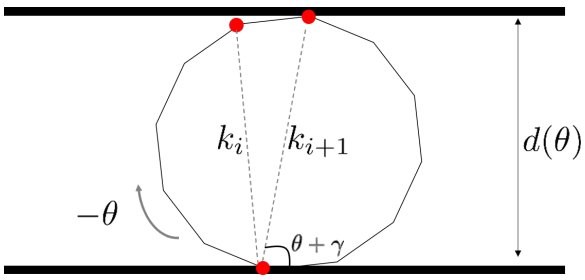


FIGURE 2.3: Minimum altitude diagram

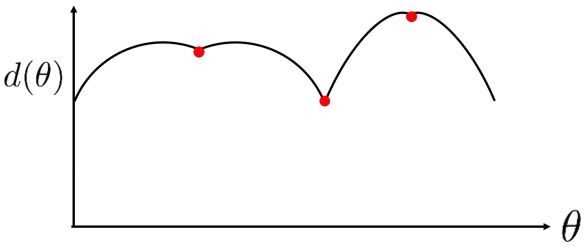


FIGURE 2.4: Minimium altitude graph

Local minima of proposed altitude function happens at discontinous points shown in Fig 2.4, which corresponds to line sweep direction perpendicular to polygon edges. Therefore, we iterate through every edge of given convex region and find

line sweep direction with shortest altitude length. Details are shown in Algorithm 1 as follows.

Algorithm 1 Minimum altitude

Input Convex polygon $\mathbf{P}$ with edges $\mathbf{E}$

Output Minimum altitude angle θ_{min}

$\theta_{min} = \mathbf{minaltitude}(\mathbf{P})$

1: **for** $\mathbf{e} \in \mathbf{E}$ **do**

2: θ_e : angle of coverage path parallel to $\mathbf{e}$

3: $\mathbf{d}_e$: altitude length of polygon with coverage angle θ_e

4: $\theta_{min} = \theta_e$ with shortest $\mathbf{d}_e$

2.2 Minimal Convex Decomposition

To extend the result of 2.1 to non convex polygons, they should be decomposed into convex sub polygons. Since larger number of polygons lead to more complex path with higher total transition cost between polygons, we use minimal convex decomposition by Keil [11] to decompose input polygon into minimum number of convex polygons. Keil's method is implemented by iterating through every concave reflex vertices inside given polygon, as illustrated in Fig 2.5. Additional diagonals are added between reflex points and other vertices to eliminate concavity, then decomposiiton with least number of added diagonals are selected. Details are shown in Algorithm 2 as follows.

Algorithm 2 Minimal Convex decomposition

Input Non convex polygon $\mathbf{P}$ with vertices $\mathbf{V}$ and relfex vertices $\mathbf{W}$

Output Addtional diagonals $\mathbf{M}$ corresponding to minimal convex decomposition

$\mathbf{M} = \mathbf{decomp}(\mathbf{P})$

```
for w ∈ W do
    for v ∈ V - w do
        if Cansee(v ,w) then
            P_left = Polygon v to w, P_right = Polygon w to v
            N = decomp(P_left).size + decomp(P_right).size
M = argmin_{v,w} N
```

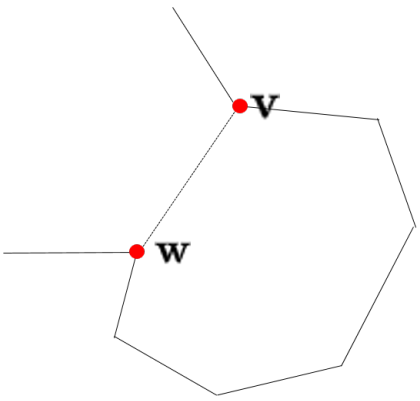


FIGURE 2.5: Concave reflex points

2.3 Dynamic Programming

Depending on the configuration of decomposed polygons, total altitude length can be reduced by merging adjacent polygons. For example at the case of Fig 2.6, total altitude length decreases while the opposite happens at Fig 2.7.

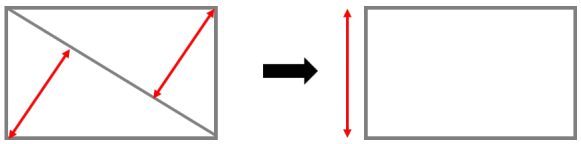


FIGURE 2.6: Altitude decrease

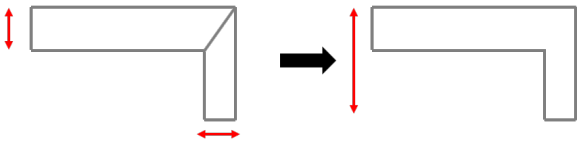


FIGURE 2.7: Altitude decrease

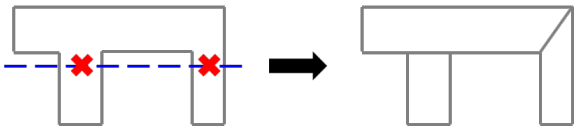


FIGURE 2.8: Non monotone polygon

Therefore, dynamics programming to find decomposed polygon configuration with least altitude length in 2.1 is performed to minimize total number ot turns in given non convex polygon. We use Huang's minimal sum of altitudes (MSA) decomposition. By iterating through all possible division of given polygon in two

sub polygons, optimal polygon formulation of given polygon with minimum sum of altitudes is produced. Essential rule for MSA decomposition is expressed as follows.

$$S(\mathbf{\Omega_P}) = min(C(\mathbf{\Omega_P}), min(S(\mathbf{P_1^i}) + S(\mathbf{P_2^i}))) \tag{2.2}$$

$\mathbf{\Omega_P}$ refers to the current polygon configuration, $S(\mathbf{\Omega_P})$ refers to the minimum altitude of $\mathbf{\Omega_P}$ and $C(\mathbf{\Omega_P})$ refers to the minimum altitude of merged configuration of $\mathbf{\Omega_P}$. $S(\mathbf{P_1^i})$ and $S(\mathbf{P_2^i})$ refers to minimum altitude of i-th possible two splitted configuration of $\mathbf{\Omega_P}$. In the case where sweep direction is not monotone to current graph as in Fig 2.8, infinite cost is assigned at $C(\mathbf{\Omega_P})$ to prevent generation of incontinuous coverage path. Details are shown in Fig 2.9 and Algorithm 3 as follows.

Algorithm 3 Dynamic Programming

Input Initial polygon formulation $\mathbf{\Omega_P}$

Output Optimal polygon formulation $\mathbf{\Omega_Q}$

$\mathbf{\Omega_Q} = \mathbf{dpprog}(\mathbf{\Omega_P})$

1: $\mathbf{\Omega_P^*} = \mathbf{TwoGraphSplit}(\mathbf{\Omega_P})$

2: **for** $\mathbf{P_1^i}, \mathbf{P_2^i} \in \mathbf{\Omega_P^*}$ **do**

3: $\quad \mathbf{\Sigma_P} = \mathbf{merge}(\mathbf{\Omega_P^*})$

4: $\quad$ **if** $\mathbf{monotone}(\mathbf{\Sigma_P}) == \mathbf{false}$ **then**

5: $\quad\quad C(\mathbf{\Omega_P}) = \mathbf{Inf}$

6: $\quad S(\mathbf{\Omega_P}) = min(C(\mathbf{\Omega_P}), min(S(\mathbf{P_1^i}) + S(\mathbf{P_2^i})))$

7: $\mathbf{\Omega_Q} = \mathbf{\Omega_P}$ with least $C(\mathbf{\Omega_P})$

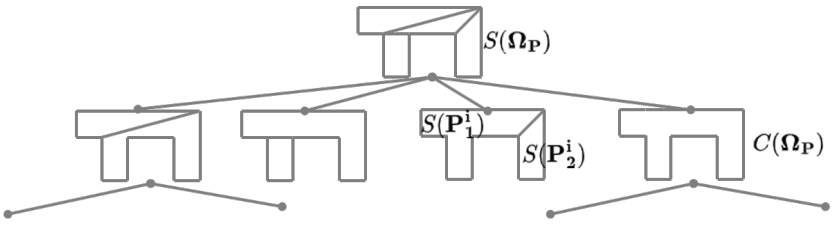


FIGURE 2.9: MSA decomposition diagram

2.4 TSP tour generation

To determine visiting sequence for the polygons generated at 2.3, we formulate TSP by constructing graph with coverage path in each node as vertex and length

between corners of adjacent coverage paths as cost. Note that resulting graph is an asymmetric directed graph, since cost between adjacent paths changes according to the previous visited nodes and corners. Graph generation is illustrated in Fig 2.10 and Algorithm 4 as follows.

Algorithm 4 Graph generation

Input Input polygon formulation $\mathbf{\Omega_P}$

Output Output graph $\mathbf{G}$ consisted of nodes $\mathbf{N}$ and edges $\mathbf{E}$

$\mathbf{G} = \mathbf{graphgen}(\mathbf{\Omega_P})$

1: **for** $\mathbf{P}_i \in \mathbf{\Omega_P}$ **do**

2: $\quad \mathbf{N} \leftarrow \mathbf{n}_i$

3: **for** Adjacent $\mathbf{P_i}, \mathbf{P_j} \in \mathbf{\Omega_P}$ **do**

4: $\quad \mathbf{E} \leftarrow \mathbf{e}_{ij}$

5: $\mathbf{G} \leftarrow \mathbf{N}, \mathbf{E}$

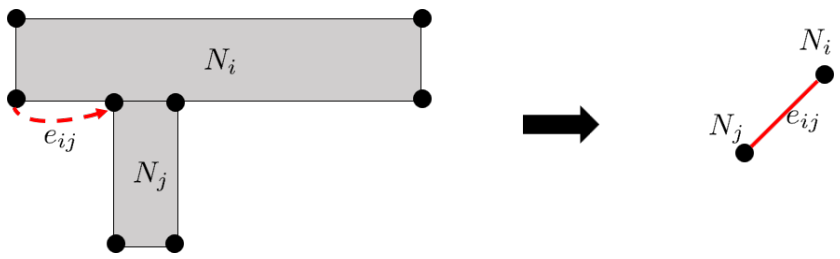


FIGURE 2.10: Graph generation diagram

After generating graph, exhaustive search concerning all possible sequence of nodes and corners is deployed to find optimal TSP tour visiting every node at

least once with minimum cost. Details are shown in Algorithm 5 as follows.

Algorithm 5 TSP tour generation

Input Input graph $\mathbf{G}$ consisted of nodes $\mathbf{N}$ and edges $\mathbf{E}$

Output Optimal node sequence $\mathbf{\Pi_{opt}}$

$\mathbf{\Pi_{opt}} = \mathbf{TSPtour}(\mathbf{G})$

```
while Π_tmp.size <= N.size do
    for n_i ∈ N do
        Π_tmp ← n_i
        for e_ij ∈ E starting form node n_i do
            Π_tmp ← n_j
            cost += |e_ij|
Π_opt = Π_tmp with least cost
```

Chapter 3

Path Following Control

In this chapter, path following control to track cleaning path generated in chapter to follow cleaning path in chapter 2 is introduced. Control is composed of vector field generation and velocity tracking control. First vector field converging to given path from current state is caculated using desired path, current USV state received from cleaning path planner and sensor measurements. Then path following control tracking vector field from current state is produced, Details are shown in Algorithm 6.

Algorithm 6 Minimal Convex decomposition

Input Desired cleaning path $\mathcal{C}$, Sensor measurement $\mathbf{X}$,

Output Path following control $\mathbf{F}$

$\mathbf{F} = \mathbf{pfcontrol}(\mathcal{C}, \mathbf{X})$

1: Position $\mathbf{x}$, Orientation θ, Velocity $(\dot{x}) \leftarrow (X)$

2: path converging vector field $\mathbf{\Psi} = \mathbf{VectorFieldGenerator}(\mathcal{C}, \mathbf{x}, \theta)$

3: path following control $\mathbf{F} = \mathbf{VelocityTrackingControl}(\mathcal{C}, (\dot{x}))$

3.1 Vector Field Generation

In this section, vector field guaranteeing robust convergence to desired path proposed by Rezende et al [12] is utilized. Fig 3.1 describes the vector field generation process. It is generated by setting nearest point on path to current position as the reference point and is constituted of convergent and traversal components.

Definition 3. Let us define $\mathbf{s}(t)$ as an nearest point on cleaning path $\mathcal{C}$ from current position $\mathbf{x}(t)$. Then convergent component $\mathbf{D}(\mathbf{x}(t))$ and traversal component $\mathbf{T}(\mathbf{x}(t))$ of generated vector field $\mathbf{\Psi}(\mathbf{x}(t))$ is constructed as follows.

$$\mathbf{D}(\mathbf{x}(t)) = \mathbf{x}(t) - \mathbf{s}(t) \tag{3.1}$$

$$\mathbf{T}(\mathbf{x}(t)) = \frac{d\mathcal{C}}{d\mathbf{s}(t)} \tag{3.2}$$

■

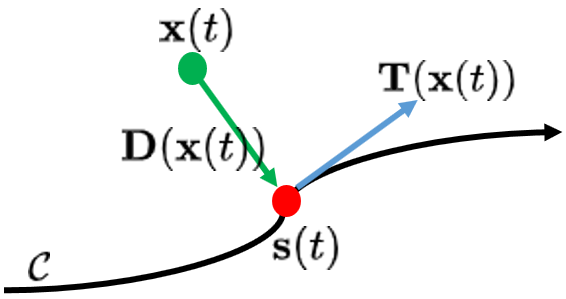


FIGURE 3.1: Vector field generation

Magnitude of each components are regulated by weight functions defined as follows.

Definition 4. Given convergent component $\mathbf{T}(\mathbf{x}(t))$, traversal component $\mathbf{D}(\mathbf{x}(t))$ and convergence coefficient k_f, convergent weight function $G(\mathbf{x}(t))$ and traversal weight function $H(\mathbf{x}(t))$ are derived as follows.

$$G(\mathbf{x}(t)) = \frac{2}{\pi}\arctan(k_f D(\mathbf{x}(t))) \tag{3.3}$$

$$H(\mathbf{x}(t)) = \sqrt{1 - G(\mathbf{x}(t))^2} \tag{3.4}$$

■

Using components, weight functions, and norm coefficient η, vector field $\mathbf{\Psi}(\mathbf{x}(t))$ is generated as follows.

$$\begin{aligned}\mathbf{\Psi}(\mathbf{x}(t)) = &- \eta(\mathbf{x}(t))G(\mathbf{x}(t))\frac{\mathbf{D}(\mathbf{x}(t))}{||\mathbf{D}(\mathbf{x}(t))||} \quad \text{(convergence)} \\ &+ \eta(\mathbf{x}(t))H(\mathbf{x}(t))\mathbf{T}(\mathbf{x}(t)) \quad \text{(traversal)}\end{aligned} \tag{3.5}$$

Proof of convenrgence of proposed vector field based on lyapunov formation could be found on Rezende's work. They cover the case of perturbed systems, ensuring robust convergence of system following proposed vector field in environment with bounded disturbances.

3.2 Velocity Tracking Control

In this section, velocity tracking control solving problem 1 is generated to track desired velocity generated in 3.1.

Problem 1. Given position $x(t)$, $y(t)$ and yaw $\psi(t)$ as shown in Fig 3.3, find linear control input u_x, u_y and angular control input τ_z tracking desired velocity $\mathbf{\Psi}(\mathbf{x}(t))$ and desired yaw angle ψ_d. Definition of ψ_d is illustrated in Fig 3.2 is expressed using $\mathbf{\Psi}$ as follows.

$$\psi_d = tan^{-1}(\frac{\mathbf{\Psi}_y}{\mathbf{\Psi}_x}) \tag{3.6}$$

■

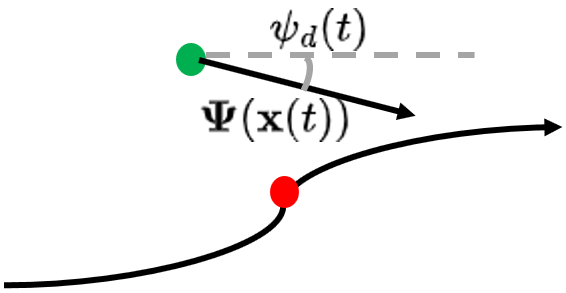


FIGURE 3.2: Desired vector field and angle

Dynamics of USV with mass $\mathbf{M}$ and moment of inertia $\mathbf{I}$ is expressed as follows. δ_u and δ_τ refers to external disturbance in linear and angular directions. R_ψ refers to rotation matrix by angle ψ in yaw direction.

$$\begin{pmatrix} u_x \\ u_y \end{pmatrix} = R_\psi^{-1} M \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} + \delta_u \tag{3.7}$$

$$\tau_z = I\ddot{\psi} + \delta_\tau \tag{3.8}$$

Based on dynamics illustrated on above, velocity tracking control constituted of feedforward and PID feedback control is constructed as follows.

$$\begin{pmatrix} u_x \\ u_y \end{pmatrix} = \mathbf{R}_\psi^{-1}(M\dot{\mathbf{\Psi}} + k_p\mathbf{e_v} + k_i\int \mathbf{e_v}dt + k_d\dot{\mathbf{e_v}}) \tag{3.9}$$

$$\tau_z = I\ddot{\psi}_d + k_{p\psi}\mathbf{e}_\psi + k_{i\psi}\int \mathbf{e}_\psi dt + k_{d\psi}\dot{\mathbf{e}}_\psi \tag{3.10}$$

$\mathbf{e_v} \in \Re^2$ refers to velocity error and $\mathbf{e}_\psi \in \Re$ refers to angular error. k_p, k_i, k_d refers to linear PID gains, and $k_{p\psi}$, $k_{i\psi}$, $k_{d\psi}$ refers to angular PID gains.

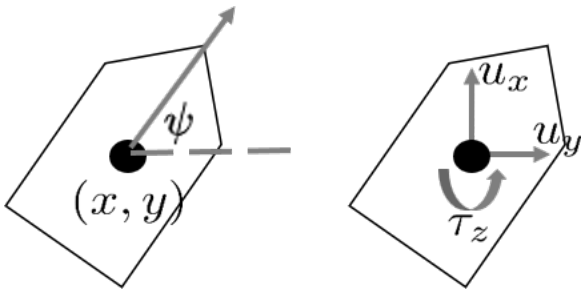


FIGURE 3.3: USV formulation

Chapter 4

System Implementation

4.1 Overview

Overall pipeline of cleaning USV is illustrated on Fig 4.3. In the beginning, map of the target area including target regions is constructed by offline SLAM [13] based on lidar and RTK GPS measurements. Based on given map, optimal cleaning path for target regions is generated by path planner in chapter 2. EKF using onboard IMU IMU and RTK GPS is implemented to find real time position and orientation of USV. Path following control is generated from EKF measurements to compute control input converging to desired cleaning path, which is distributed to low level thrusters at the end of the pipeline.

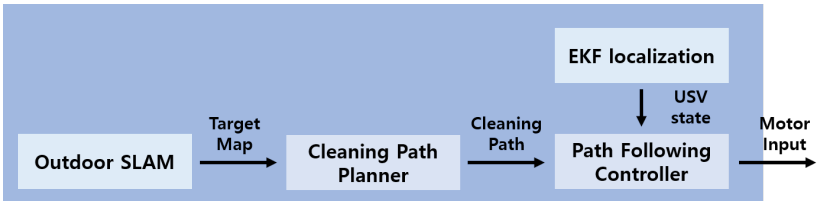


FIGURE 4.1: USV system pipeline

4.2 Custom USV

In this section, design of custom USV for lake cleaning implementation is introduced. Onboard sensors and computing units are placed on stably designed floating platforms, with actuators capable of full 3 DOF actuation.

4.2.1 USV Hull Design

Considering lake environments with external disturbances due to wind and current, USV should minimize perturbation in roll and pitch direction to perform cleaning with high quality. We select catamaran hull design which has high buoyancy on both ends to maximize ship stability in both directions, as could be seen in Fig 4.2.

Physical properties of USV are also selected to guarantee stability. We distribute mass of USV considering relative position of center of gravity and buoyancy, preventing toppling behavior from recovery torques. Dimensions are shown in Table 4.1.

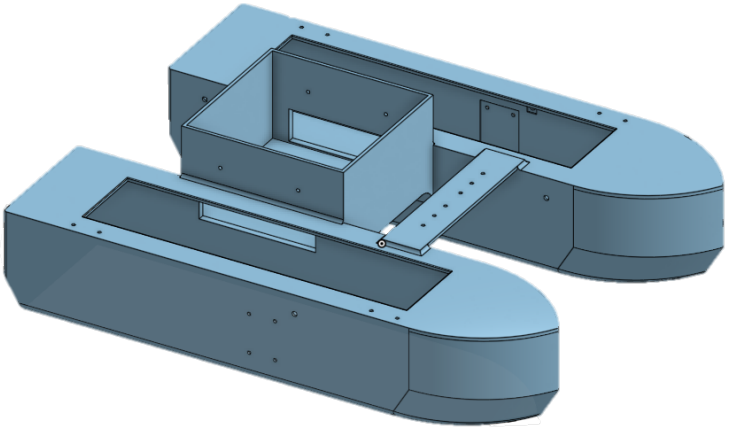


FIGURE 4.2: USV hull design

x[m]	0.8
y[m]	0.64
z[m]	0.22
Mass[kg]	16.66
Submerged Height[m]	0.12
Ix[kgm^2]	0.916
Iy[kgm^2]	0.883
Iz[kgm^2]	1.37

TABLE 4.1: Specs of custom USV

4.2.2 Hardware Architecture

Overall hardware architecture is shown in Fig 4.3. Measurement from onboard sensors LIDAR, RTK GPS, IMU are transferred to onboard computer, which

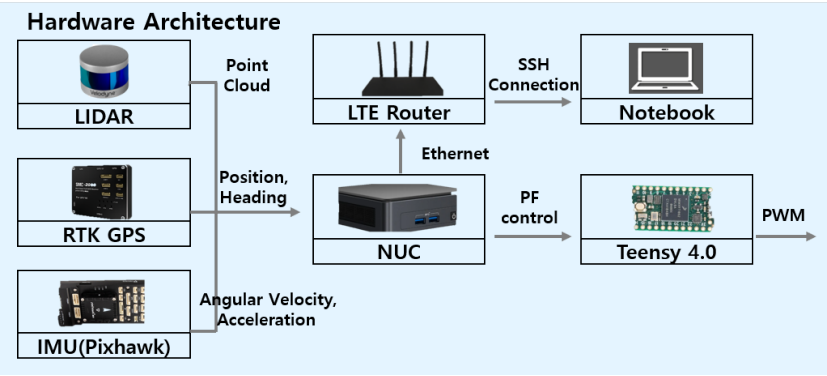


FIGURE 4.3: USV hardware architecture

computes and delivers path following control input to teensy board. Teensy converts received input to PWM signals and distributes them to ESCs, which actuates connected thrusters. Onboard computer is also connected to wireless routers embedded inside the USV. This enables real time monitoring of USV in outdoor lake environments. Remote controlled thrusters are also attached with seperate power sources for emergency recovery in case of system failure. Specifications of each elements are presented in Table 4.2, and complete assembly of USV is shown at Fig 4.4.

Onboard computer	Intel NUC8i7BEH
RTK GPS	Synerex SMC 2000, SMC +
IMU	Pixhawk 6x
LIDAR	Velodyne puck
ESC	Bluerobotics basic ESC
MCU	Teensy 4.0
Thruster	Hydromea diskdrive 50

TABLE 4.2: Custom USV hardware specifications

FIGURE 4.4: USV hardware assembly

4.2.3 Thrust Distribution

As described in 4.1, path following control should be distributed to low level thrusters. Thruster arrangement designed to generate fully actuated thrust by 4 tilted actuators is considered. Fig 4.5 illustrates selected thruster arrangement with tilt angle α.

Distributed low level thrusts f_1, f_2, f_3, f_4 computed from path following thrust u_x, u_y, τ_z in chapter 3 are expressed as follows.

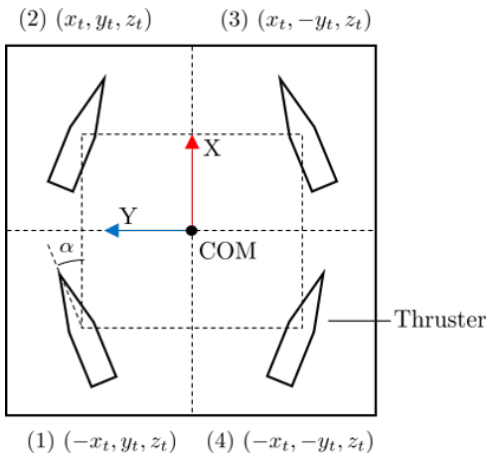


FIGURE 4.5: Thruster arrangement

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} = B^{\dagger} \begin{pmatrix} u_x \\ u_y \\ \tau_z \end{pmatrix} \tag{4.1}$$

$B^{\dagger}$ refers to pseudo inverse of shape matrix representing arrangement of 4 tilted thrusters, which is expressed as follows.

$$T_z = x_t sin\alpha + y_t cos\alpha \tag{4.2}$$

$$B = \begin{pmatrix} cos\alpha & cos\alpha & cos\alpha & cos\alpha \\ sin\alpha & -sin\alpha & sin\alpha & -sin\alpha \\ -T_z & -T_z & T_z & T_z \end{pmatrix} \tag{4.3}$$

Chapter 5

Result

In this chapter, simulation and experiment results to validate performance of proposed cleaning USV is given. Scenario visiting seperate non convex target regions, region 1,2,3 on 200m $\times$ 100m scale lake map is selected as shown in Fig 5.1. To refect real world cleaning operations, dead zones expressed in red lines are considered. Target regions avoiding dead zones and decomposed as in chapter 2 are shown in black polygons. Generated desired cleaning path with length of 594m and consisted of 30 turns is shown in blue lines. Controller gains and coefficients used in results are shown in Table 5.1a and Table 5.1b.

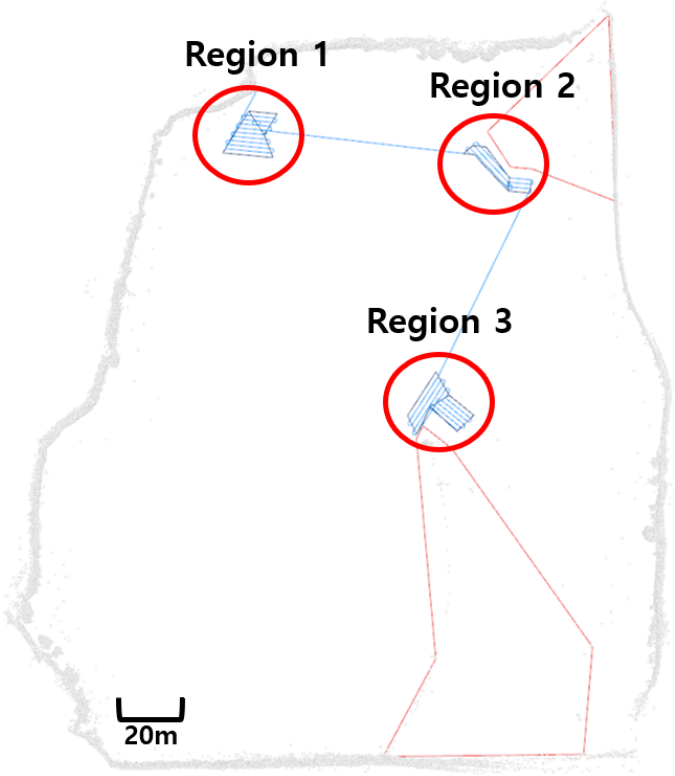


FIGURE 5.1: Cleaning scenario setting

η	0.5	k_f	8.0
k_p	20.0	$k_{p\psi}$	5.0
k_i	0.0	$k_{i\psi}$	0.0
k_d	0.0	$k_{d\psi}$	0.0

(a) Simulation

η	0.5	k_f	4.0
k_p	20.0	$k_{p\psi}$	2.0
k_i	1.0	$k_{i\psi}$	0.0
k_d	0.0	$k_{d\psi}$	1.2

(b) Experiment

TABLE 5.1: Controller gains and coefficients

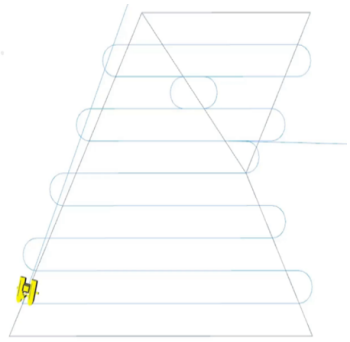
5.1 Simulation

Simulator by ROS based C++ environment is constructed. Drag force and bounded gaussian disturbances $\delta_x, \delta_y, \delta_\psi$ are considered as follows in USV dynamics to represent real lake environment. k_x, k_y, k_ψ are drag force coefficients.

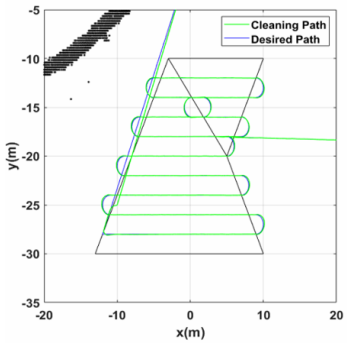
$$\begin{pmatrix} u_x \\ u_y \end{pmatrix} = R_\psi^{-1} M \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} - \begin{pmatrix} k_x|\dot{x}|\dot{x} & 0 \\ 0 & k_y|\dot{y}|\dot{y} \end{pmatrix} + \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix}, \tag{5.1}$$

$$\tau_z = I\ddot{\psi} - k_\psi|\dot{\psi}|\dot{\psi} + \delta_\psi \tag{5.2}$$

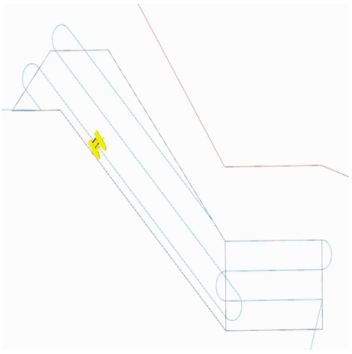
Visualization is done using RVIZ and plot is made using MATLAB. System showed 8.12cm path following RMSE error, showing high convergence performance to desired cleaning path. Simulation setting and results are shown in Fig 5.2.



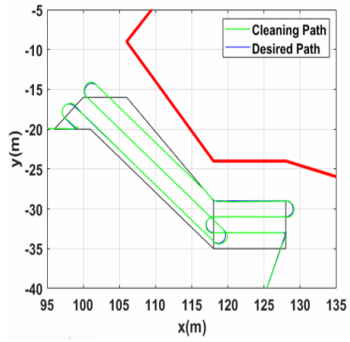
(a) Region1 setting



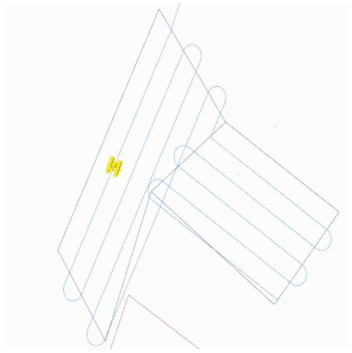
(b) Region1 plot



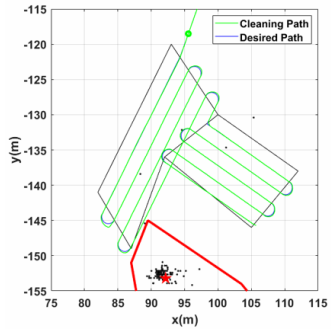
(c) Region2 setting



(d) Region2 plot



(e) Region3 setting



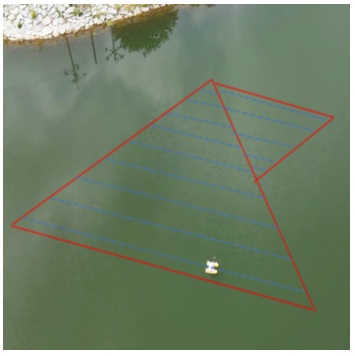
(f) Region3 plot

FIGURE 5.2: Simulation result

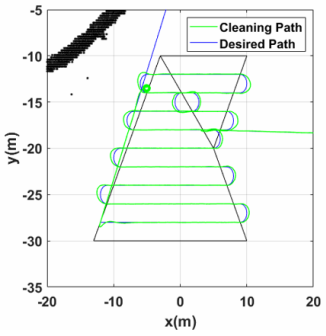
5.2 Outdoor Experiment

Experiment was conducted on lake map identical to simulation environment. constructed by offline SLAM [13]. Wind up to 5 m/s speed was present during cleaning operation.

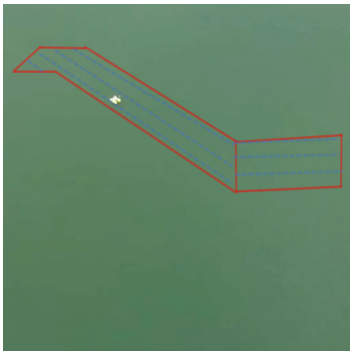
Our system showed 21cm path following RMSE error for total cleaning path, which is very accurate considering the scale of lake and USV. Total battery power consumed was 16 percent, sufficient to execute cleaning mission on total map without return home operation. Experiment setting and results are shown in Fig 5.3.



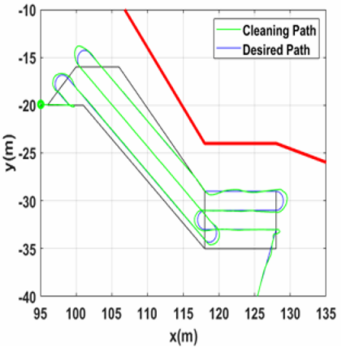
(a) Region1 setting



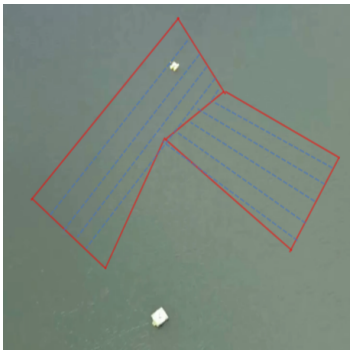
(b) Region1 plot



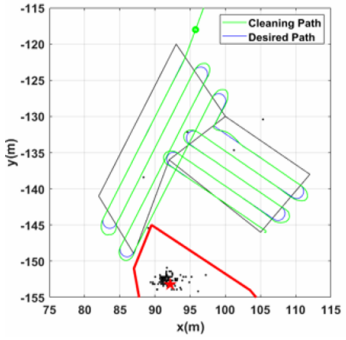
(c) Region2 setting



(d) Region2 plot



(e) Region3 setting



(f) Region3 plot

FIGURE 5.3: Experiment result

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this paper, we implemented complete automated lake cleaning USV system ranging from mapping to control. Path planning for non convex polygons based on minimum turns was developed. We decided optimal coverage direction for convex polygon, then searched for optimal decomposition of non convex polygon with minimum number of turns. Minimal decomposition was applied to decompose non convex region into minimum number of convex polygons, then adjacent polygons was merged through dynamic programming. Next, path following control using velocity field was utilized to follow planned path. Velocity field ensuring

robust convergence to given path was generated, then it was tracked by velocity tracking control.

We also proposed design process of custom USV used for lake cleaning implementation. Hull design and physical properties to ensure USV stablity was elaborated, followed by hardware specifications including onboard sensors, computing units, ESCs and thrusters. Simulation and experiment results on lake environment was conducted by constructed USV to verify performance of proposed system.

6.2 Future Work

In future works, we plan to develop online cleaning path planning of given lake, generating cleaning path while mapping target map simultaneously. Mapping of unknown environments would be implemented, and exhaustive search process in current path planning method will be enhanced to reduce computing time.

Moreover, we intend to improve current system by using multiple USVs. Coverage problem with multiple mobile robots has already been researched in various applications. By entailing optimization factors related to multiple USVs in current path planning formulation, we plan to extend current system to include multiple cleaning USVs.

Bibliography

[1] Clearbot. https://www.clearbot.org/. Accessed: 2023-06-22.

[2] oceanalpha - mc120. https://www.oceanalpha.com/product-item/mc120/. Accessed: 2023-06-22.

[3] Yogang Singh Shaocheng Luo, Jun Han Bae Hanyao Yang, Xiumin Diao J. Eric Dietz, and Byung Cheol Min. Image processing and model-based spill coverage path planning for unmanned surface vehicles. In *OCEANS MTS/IEEE SEATTLE*, pages 1–9, 2019.

[4] Yu Liang Hsu Hsing Cheng Chang, Jia Ron Wu San Shan Hung, Guan Ru Ou, and Chuan Hsu. Autonomous water quality monitoring and water surface cleaning for unmanned surface vehicle. *Sensors*, 2021.

[5] Yuwei Cheng Jiannan Zhu, Yixin Yang. Smurf: A fully autonomous water surface cleaning robot with a novel coverage path planning method. *Journal of Marine Science and Engineering*, 2022.

[6] Ioannis Rekleitis Raphael Mannadiar. Optimal coverage of a known arbitrary environment. In *IEEE International Conference on Robotics and Automation*, pages 5525–5530, 2010.

[7] Ioannis Rekleitis Anqi Xu, Chatavut Viriyasuthee. Optimal complete terrain coverage using an unmanned aerial vehicle. In *IEEE International Conference on Robotics and Automation*, pages 2513–2519, 2011.

[8] W.H. Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *IEEE International Conference on Robotics and Automation*, pages 27–32, 2001.

[9] Stephen L. Smith Stanislav Bochkarev. On minimizing turns in robot coverage path planning. In *IEEE International Conference on Automation Science and Engineering*, pages 1237–1242, 2016.

[10] Elon Rimon Iddo Shnaps. Online coverage of planar environments by a battery powered autonomous mobile robot. In *IEEE Transactions on Automation Science and Engineering*, pages 425–436, 2016.

[11] Jack Snoeyink Mark Keil. On the time bound for convex decomposition of simple polygons. *International Journal of Computational Geometry and Applications*, 2002.

[12] Adriano M. C. Rezende, Vinicius M. Goncalves, and Luciano C. A. Pimenta. Constructive time-varying vector fields for robot navigation. *IEEE Transactions on Robotics*, pages 852–867, 2022.

[13] Hdl graph slam. https://github.com/koide3/hdl_graph_slam. Accessed: 2023-06-22.

요약

본 논문에서는 호수 청소 작업을 위한 무인수상정의 경로 계획, 운항제어 및 구현을 시행한다. 우선 다각형으로 정의된 목표 청소 지역에서 최소 회전 수를 기반으로한 최적 청소경로를 생성하여 복잡한 구역에서 효율적인 청소 경로를 계획한다. 이후 벡터장 기반 경로추적 제어를 적용하여 외란 존재시에도 강건하게 청소 경로를 추종한다. 마지막으로 실외 지도형성과 위치추정이 가능하도록 독자적으로 개발한 무인수상정에 상기한 경로 생성기 및 제어기를 탑재하여 시뮬레이션 및 실제환경 실험을 진행, 본 논문에서 제안한 시스템의 성능을 검증하였다.

주요어: Unmanned surface vehicle (USV), Autonomous cleaning, Path following, Optimal coverage
학번: *2021-21797*

Acknowledgements

우선 2년간 제 학업 성취 및 진로 결정에 큰 도움을 주신 이동준 교수님께 감사의 말을 전합니다. 모든 학생들을 신경쓰시며 연구지도를 해주시고, 능력을 이끌어내 주시는 모습을 보며 존경하지 않을 수 없었습니다.

또한 2년간 연구실에서 생활을 함께한 동료들에게도 감사의 말을 전합니다. 우선 같은 그룹에서 같이 고생하며 연구과제를 진행한 석현이와 승욱이, 준택이에게 감사한다는 말을 전하고 싶습니다. 또한 같은 년도에 들어와 연구실 적응에 큰 도움을 준 민성이형 및 재우형, 다른 그룹임에도 불구하고 막힐때마다 도움을 준 지석이형, 상윤이형, 태균이형, 용혁이형, 현수형, 야외 실험에 도움을 준 도윤이와 기홍이에게도 감사한다는 말을 전하고 싶습니다. 이 외에도 함께 연구실 생활을 하며 많은 것을 배우고 깨닫게 해준 다른 연구실 구성원 분들과 연구에 집중할 수 있는 환경을 조성해주신 주남희 선생님에게도 감사의 말씀을 전합니다.

무엇보다도 항상 헌신적인 사랑으로 대학원 생활과 취업 활동에 힘이 되어 주셨던 부모님께 감사드리고 사랑한다는 말을 전합니다. 가족의 지원 없이는 석사학위와 취업의 결실을 맺지 못했을 것입니다.

2023년 6월

정석진